



responsabile@hourglabs.org

Norme di Progetto

Versione 4.0

Informazioni documento

Nome	<i>Norme_di_Progetto_v3.0.pdf</i>
Versione	4.0
Data creazione	2012-12-06
Data ultima modifica	2013-08-20
Stato del Documento	Formale ad uso interno
Redazione	Paolo Bustreo Sasa Ilievski
Verifica	Giovanni Morlin
Approvazione	Gioele Lorenzo Cresce
Distribuzione	hourglass Prof. Tullio Vardanega Prof. Riccardo Cardin

Registro delle modifiche

Data	Versione	Ruolo	Descrizione	Autore
2013-08-20	4.0	Responsabile	Approvazione	Gioele Lorenzo Cresce
2013-08-19	3.1	Verificatore	Miglioramento forma dopo segnalazione in RQ	Giovanni Morlin
2013-07-10	3.0	Responsabile	Approvazione	Gioele Lorenzo Cresce
2013-06-16	2.3	Verificatore	Correzione di errori grammaticali e miglioramento aspetto del documento	Giovanni Morlin
2013-04-21	2.2	Amministratore	Aggiornamento e modifica della sezione Ambiente di Verifica e Validazione	Thomas Rossetto
2013-02-09	2.1	Amministratore	Correzione errori rilevati dalla RP	Thomas Rossetto
2013-01-12	2.0	Responsabile	Approvazione	Gioele Lorenzo Cresce
2013-01-12	1.3	Verificatore	Correzione degli errori grammaticali e miglioramento della forma	Giovanni Morlin
2013-01-11	1.2	Analista	Stesura sottosezioni: Design Pattern, Specifica Tecnica, Convenzione di codifica, Framework. Ampliamento sezione Ambiente di Verifica e Validazione	Sasa Ilievski

Data	Versione	Ruolo	Descrizione	Autore
2013-01-11	1.1	Analista	Riorganizzazione generale del documento, correzione errori e aggiunta miglioramenti segnalati in uscita dalla RR	Paolo Bustreo
2012-12-10	1.0	Responsabile	Approvazione documento	Thomas Rossetto
2012-12-07	0.5	Verificatore	Correzione errori grammaticali, miglioramento forma	Gioele Lorenzo Cresce
2012/12/07	0.4	Analista	Stesura sezioni: Componenti grafici, Analisi dei Requisiti, Progettazione e Gestione milestones e ticketing	Paolo Bustreo
2012/12/06	0.3	Amministratore	Stesura sezioni: Incontri, Ambiente di lavoro	Giovanni Morlin
2012/12/05	0.2	Analista	Stesura sezioni: Tipi di documenti, Versionamento, Struttura del documento, Template, e Norme tipografiche	Riccardo Cesarotto
2012/12/05	0.1	Amministratore	Prima stesura del documento: creazione dello scheletro del documento, stesura sezioni: Introduzione, Comunicazioni, Documenti	Giovanni Morlin

Indice

1	Introduzione	6
1.1	Scopo del documento	6
1.2	Riferimenti	6
1.2.1	Riferimenti normativi	6
1.2.2	Riferimenti informativi	6
1.3	Glossario	6
2	Gestione comunicazioni	7
2.1	Comunicazioni interne	7
2.2	Comunicazioni esterne	7
3	Gestione incontri	8
3.1	Incontri interni	8
3.2	Incontri esterni	8
4	Documentazione	9
4.1	Lingua	9
4.2	Tipi di documenti	9
4.2.1	Documenti informali	9
4.2.2	Documenti formali	9
4.2.3	Verbali	9
4.2.4	Glossario	10
4.3	Versionamento	11
4.4	Struttura del documento	11
4.5	Template	12
4.6	Norme tipografiche	13
4.7	Elementi visivi	14
4.7.1	Immagini	14
4.7.2	Tabelle	14
5	Analisi dei Requisiti	15
5.1	Classificazione	15
5.2	Descrizione grafica	16
5.3	Descrizione didascalica	16
6	Progettazione	17
6.1	Classi di verifica	17
6.2	Diagrammi	17
6.3	Design Pattern	17
6.4	Stile di progettazione	18
6.5	Specifica Tecnica	18

7	Codifica	19
7.1	Convenzioni di codifica	20
7.1.1	Struttura delle classi	20
7.1.2	Convenzioni sui nomi	20
7.1.3	Struttura del codice Java	20
8	Ambiente di lavoro	21
8.1	Ambiente Generale	21
8.1.1	Sistema operativo	21
8.1.2	Repository	22
8.2	Ambiente dei documenti	22
8.2.1	Scrittura dei documenti	22
8.2.2	Verifica ortografica	23
8.2.3	Pianificazione	23
8.2.4	Diagrammi UML	23
8.3	Ambiente di sviluppo	24
8.3.1	IDE	24
8.3.2	Framework	24
8.3.3	Strumenti per il versionamento	24
8.3.4	Gestione milestones e ticketing	25
8.3.4.1	Creazione di milestones	25
8.3.4.2	Creazione ticket	26
8.3.4.3	Chiusura ticket	26
8.3.4.4	Creazione ticket di verifica	27
8.3.4.5	Esecuzione verifica e creazione ticket di correzione	27
8.4	Ambiente di Verifica e Validazione	27
8.4.1	Tracciamento dei requisiti	27
8.4.2	Analisi Statica	30
8.4.3	Analisi Dinamica	30

1 Introduzione

1.1 Scopo del documento

Con questo documento il gruppo **hourglass** si prefigge lo scopo di specificare delle norme che dovranno essere rispettate in tutto il processo di sviluppo del software.

Ogni singolo membro del gruppo è obbligato a visionare e prendere atto di tale documento, così da garantire coerenza, efficienza e correttezza nel proprio lavoro. Inoltre, seguendo le convenzioni qui esposte si semplificano attività quali la lettura, la verifica e la testabilità del prodotto sia da parte di membri interni che di terzi.

Il Verificatore in carica deve provvedere alla segnalazione di violazioni di tali norme durante la revisione dei documenti, in modo da salvaguardare il corretto proseguimento del lavoro attraverso procedure di correzione.

Se un membro ritiene di non essere in grado di aderire a tutte le convenzioni qui espresse, deve contattare l'Amministratore di progetto per chiederne una modifica oppure l'aggiunta di un'eccezione.

1.2 Riferimenti

1.2.1 Riferimenti normativi

- Regole di partecipazione alle revisioni di progetto

<http://www.math.unipd.it/tullio/IS-1/2012/>

1.2.2 Riferimenti informativi

- Guida introduttiva a \LaTeX : http://www.lorenzopantieri.net/LaTeX_files/ArteLaTeX.pdf/
- Git community book: <http://book.git-scm.com/>
- Documentazione TexMaker: <http://www.xmlmath.net/texmaker/doc.html/>

1.3 Glossario

Nel documento "*Glossario_v3.0.pdf*" sono elencati e descritti alcuni termini tecnici o acronimi che sono stati utilizzati nei documenti formali del gruppo, così da migliorarne la comprensione ed evitare ambiguità.

Tutti i termini presenti nel glossario compaiono sottolineati nei documenti, mentre per riferirsi a nomi di documenti ed enti o persone esterni si utilizzerà l'italico.

2 Gestione comunicazioni

2.1 Comunicazioni interne

Le comunicazioni inerenti al gruppo avvengono principalmente attraverso la piattaforma *Facebook*¹. È stato creato un gruppo privato denominato *hourglass-SWE*, per mezzo del quale i membri possono interagire. Partecipano al gruppo esclusivamente i sette individui che compongono **hourglass** e grazie a tale limitazione il Responsabile di Progetto può accertarsi che ogni componente abbia preso atto di nuove comunicazioni: *Facebook* consente di tener traccia di quante visualizzazioni possiede un'inserzione in tempo reale.

Se sorge la necessità di comunicazioni più frequenti ed interattive, il Responsabile decide se aprire un *Google Hangout*² con il quale si possono effettuare videoconferenze tra più utenti. I partecipanti sono obbligati a fornire un resoconto scritto al termine dell'incontro, redatto secondo i canoni delle norme sui Verbali disponibili nel paragrafo 5.3 - *Verbali*.

2.2 Comunicazioni esterne

L'Amministratore di Progetto ha il compito di relazionarsi con enti esterni attraverso l'apposita mail: amministratore@hourglass.org.

Una volta ricevute comunicazioni esterne, l'Amministratore deve informare il resto del gruppo dell'avvenuto contatto attraverso le procedure delineate al paragrafo 2.1.

¹<http://www.facebook.com/>

²<http://www.google.com/+learnmore/hangouts/>

3 Gestione incontri

3.1 Incontri interni

Il Responsabile di Progetto stabilisce l'ora, il luogo e la durata di un incontro tra i componenti del gruppo. Una volta prese queste decisioni, ha il dovere di informare i partecipanti con le metodologie del paragrafo 2.1, indicando anche le motivazioni e il richiedente. Vi sono due restrizioni: i membri del gruppo devono essere avvisati almeno quattro giorni prima per potersi organizzare con i propri impegni ed il giorno scelto non deve essere né il sabato né la domenica.

Nella scelta della data e dell'orario devono essere prese in considerazione le preferenze espresse nei commenti dell'inserzione, pubblicata dal responsabile nella piattaforma di comunicazione. In questo modo si vogliono evitare assenze di alcuni componenti del gruppo, ma se ciò non è possibile, colui che è impossibilitato a recarsi nel luogo e all'ora stabilita deve fornire al Responsabile una valida motivazione scritta in un commento all'inserzione precedente.

Alla fine di ogni incontro, il Responsabile delega la stesura di un resoconto ad uno dei partecipanti.

3.2 Incontri esterni

Anche in caso di incontri esterni è il Responsabile di Progetto a stabilirne le modalità con il Committente o con i Proponenti.

Un membro può richiedere un incontro con il Committente al Responsabile, il quale ne valuterà l'effettiva necessità, per poi dare esito negativo o positivo alla richiesta. Se l'esito sarà positivo, contatterà il Committente per decidere luogo, ora e data, poi avviserà gli altri membri. Una volta avvenuto tale incontro, il Responsabile è tenuto a redigerne un verbale.

4 Documentazione

Ogni documento prodotto da **hourglass** durante il processo di sviluppo del software sarà caratterizzato dagli standard riportati nella sezione sottostante.

4.1 Lingua

La lingua utilizzata per la descrizione dei documenti, per la comunicazione tra i membri del gruppo e con il Committente è quella italiana.

4.2 Tipi di documenti

4.2.1 Documenti informali

I documenti informali saranno creati solamente per uso interno al gruppo di lavoro. Potranno diventare formali e quindi essere successivamente distribuiti dopo aver ricevuto una revisione e l'approvazione del Responsabile di Progetto. I documenti informali saranno memorizzati nel repository come specificato a *14.3 – Repository*. I file andranno denominati con lo stesso nome del documento, gli spazi andranno indicati come *underscore* e andrà scritta in maiuscolo ogni lettera iniziale di una parola. Dopo il nome va indicata la versione del documento nel formato *vX.Y*, preceduta da uno spazio. Esempio: *Analisi_Dei_Requisiti_v1.1.pdf*.

4.2.2 Documenti formali

I documenti formali sono approvati dal Responsabile di Progetto e saranno pronti per essere divulgati alla loro lista di distribuzione. Ciascun documento formale sarà inserito nel repository dal Responsabile dopo averlo fatto verificare dai Verificatori, per poi approvarlo e aver accertato che rispetti tutti i requisiti descritti nelle Norme di Progetto. La nomenclatura del file seguirà la stessa regola descritta per i documenti informali.

4.2.3 Verballi

I verballi documenteranno tutte le riunioni tra i membri del gruppo **hourglass** e gli incontri con il Committente. Tutti i verballi, come gli altri documenti, seguiranno le fasi di stesura, verifica e approvazione da parte del Responsabile e inseriti nella cartella apposita denominata *Verballi* nel repository. Il file che conterrà il verbale dovrà essere denominato nel seguente modo:

Verbale_Numero_Data.pdf
dove:

- **Numero** indicherà un intero che andrà incrementato a ogni nuovo incontro di ogni tipo, partirà da uno.
- **Data** indicherà la data dell'incontro che il Verbale documenta, espressa secondo lo standard adottato dal gruppo (si veda sezione 9 - *Norme Tipografiche*).

I tipi di incontri sono:

- **Incontro reale tra i membri del gruppo**
- **Incontro tra il gruppo e il committente**
- **Incontro virtuale tra i membri del gruppo**

La prima pagina dei verbali sarà strutturata nella stessa maniera degli altri documenti (si veda sezione 4.4 – *Struttura del documento*). A seguire andranno elencati in ordine le seguenti informazioni:

- **Data** del ritrovo, espressa nel formato normato dal gruppo.
- **Ora** del ritrovo, espressa dal formalismo:
 - HH:MM con: HH l'ora e MM i minuti. Nel caso l'ora e/o i minuti siano composti da una sola cifra, si aggiunge lo zero davanti.
- **Luogo** dell'incontro dove:
 - se reale viene indicata la provincia e la città del ritrovo con la seguente dicitura: Provincia, Città.
 - se virtuale viene specificato il software utilizzato per la conversazione (es. *Hangout* di *Google*).
- **Lista dei partecipanti del gruppo** con i nomi e cognomi dei membri del gruppo che hanno partecipato all'incontro.
- **Lista dei partecipanti esterni** con i nomi e cognomi dei partecipanti esterni che possono essere i Proponenti o i Committenti.

Dopo aver specificato i campi appena descritti, si farà un riassunto sull'incontro indicando tutti i suoi punti importanti e le decisioni prese.

4.2.4 Glossario

I termini tecnici o gli acronimi relativi allo sviluppo del progetto saranno contenuti in un glossario in modo tale da renderli di facile comprensione al lettore. I termini scritti nel glossario saranno sottolineati nel documento ad ogni loro prima apparizione nel testo. Il glossario sarà chiamato *Glossario.pdf* e ciascun membro, a seguito del bisogno dell'inserimento di un termine, verificherà che esso non compaia già all'interno del glossario stesso e procederà con l'inserimento vero e proprio. Come tutti gli altri documenti anche il glossario dovrà essere sottoposto alle operazioni di stesura, verifica e validazione prima di una qualsiasi divulgazione.

4.3 Versionamento

Il redattore ha il compito di specificare la versione del suo documento secondo il seguente formalismo:

v. X.Y

con il seguente significato:

- **X** è un numero intero compreso tra 0 e 99, ad ogni revisione formale del documento viene incrementato di un'unità. Ogni suo incremento comporta l'azzeramento di Y
- **Y** è un numero intero compreso tra 0 e 99 ed ogni qual volta il documento è soggetto ad una modifica rilevante³ viene incrementato di un'unità.

4.4 Struttura del documento

Ciascun documento formale sarà composto rispettivamente dalle seguenti parti:

- **Pagina iniziale:** contiene le informazioni del documento ovvero:
 - *Logo del gruppo* con l'indirizzo e-mail del gruppo scritto a seguito.
 - *Nome del documento.*
 - *Versione del documento:* definito con il formalismo specificato nella sezione 4.3 – Versionamento.
- **Informazioni documento**
 - *Nome del documento.*
 - *Versione del documento:* definito con il formalismo specificato nella sezione 4.3 – Versionamento.
 - *Data di creazione:* indicante il giorno in cui è stata redatta la prima versione del documento.
 - *Data dell'ultima modifica:* indicante il giorno in cui è stato fatto l'ultimo cambiamento sul documento.
 - *Stato del documento:* che indica se il documento ha visibilità interna o esterna e se è di carattere formale o meno.
 - *Redazione:* lista dei membri del gruppo che hanno curato e scritto il documento.
 - *Verifica:* lista dei membri del gruppo che hanno verificato il documento.
 - *Approvazione:* indicante il nome del Responsabile che ha approvato la versione formale del documento.
 - *Distribuzione:* elencante chi sono i destinatari del documento.

³Con modifiche rilevanti non si intendono correzioni ortografiche oppure modifiche nella punteggiatura.

- **Registro delle modifiche:** contiene le modifiche effettuate durante lo sviluppo del progetto sul documento. È una tabella ordinata in ordine decrescente per versione (e data), partendo quindi dalla prima riga relativa all'ultima modifica effettuata e proseguendo sino alle modifiche più vecchie e alla creazione. Ciascuna riga è caratterizzata dalle seguenti colonne:
 - *Data:* che indica quando è stata apportata la modifica.
 - *Versione:* che indica su quale versione del documento è stata eseguita la modifica.
 - *Ruolo:* riportante il ruolo di chi ha operato la modifica in questione.
 - *Descrizione:* riporta in modo sintetico quali aggiunte ed operazioni siano state compiute sul testo.
 - *Autore:* riportante il nominativo di chi ha operato la modifica in questione.
- **Indice**
- Nelle **pagine** del documento saranno inserite le seguenti componenti:
 - *Logo del gruppo:* inserito in alto a sinistra.
 - *Nome del Documento e relativa versione:* posizionati in alto a destra.
 - *Nome del Documento:* posizionato a piè pagina a sinistra.
 - *Numero di pagina:* posizionato a piè pagina a destra nel seguente formato:
`n_di_pagina di numero_pagine_totale.`

4.5 Template

Il template realizzato dal gruppo per la scrittura dei documenti dovrà essere rispettato da tutti i membri di lavoro.

E' creato in L^AT_EX ed è presente all'indirizzo:

<https://github.com/Hourglass-Labs/MyTalk>

Il file *template.tex* ha all'interno codice per l'impaginazione dei documenti. Questo file funzionerà da base stilistica per la creazione degli altri documenti: basterà effettuare una sua copia e rinominarlo con un altro nome (es. *Analisi_Dei_Requisiti.tex*). Può essere modificato accertandosi prima di modificare le costanti che sono presenti all'inizio del documento (Tipo, Versione, Data) e, una volta compilato, le ricorrenze degli elementi relativi alle costanti verranno modificate automaticamente nelle varie parti del documento. Se si vogliono apportare delle modifiche al *template.tex* bisognerà fare richiesta all'Amministratore e, se attuate, si dovranno informare tutti i membri del gruppo.

4.6 Norme tipografiche

Nei documenti creati si rispettano delle regole tipografiche di seguito elencate.

- **Date:** ciascuna data riportata nei documenti dovrà essere scritta in un formato aderente all'*ISO 8601*⁴, ossia nella forma: *AAAA-MM-GG*, con *AAAA* indicante l'anno, *MM* il mese, *GG* il giorno.
- **Grassetto:** utilizzato negli elenchi per evidenziare ciascun elemento che viene specificato e descritto oppure per sottolineare concetti o termini importanti presenti all'interno del testo del documento in modo da porvi particolare attenzione. Il grassetto andrà utilizzato anche per il nome del progetto ogni talvolta verrà nominato (**MyTalk**), nonché per il nome del gruppo.
- **Corsivo:** utilizzato per evidenziare nomi di enti e documenti, oppure riferimenti a sezioni del documento, nomi di documenti, termini tecnici, o a software utilizzati. Riguardo a questi ultimi, ad ogni prima occorrenza nel documento del nome del software inserito, sarà necessario associare una nota a piè di pagina con dei riferimenti utili a documentazione online ad esso relativa.
- **Sottolineato:** utilizzato per segnalare che il termine si può cercare sul glossario. Ogni termine che si trova nel glossario va sottolineato in tutte le sue occorrenze all'interno dei documenti (escluso sui titoli delle sezioni, sulle definizioni e sui percorsi). Nel caso il medesimo termine compaia più volte in una stessa sezione, non sarà necessario sottolinearlo ad ogni occorrenza ma solo alla prima nella relativa sezione (incluso quindi anche le sotto-sezioni).
- **URL:** vanno inseriti utilizzando gli appropriati strumenti d'interazione messi a disposizione dall'ambiente L^AT_EX.
- **Maiuscolo:** utilizzato per la prima lettera di una parola che viene scritta dopo un punto oppure per gli acronimi. Viene usato anche sulla prima lettera di ogni parola del nome di un ruolo che non sia una preposizione.

⁴<http://www.w3.org/TR/NOTE-datetime>

4.7 Elementi visivi

4.7.1 Immagini

Ciascuna immagine che verrà inserita all'interno di un documento sarà accompagnata nella parte sottostante da una didascalia formata da una breve testo che la descriva. Tutte le immagini che sono nei documenti saranno nel formato grafico vettoriale oppure in PNG. È preferibile utilizzare il primo dei due in quanto il ridimensionamento dell'immagine con tale formato non influenza la qualità e la risoluzione. Per la conversione a *PNG* (nel caso l'immagine abbia un'estensione diversa) verrà utilizzato l'editor grafico *Gimp*⁵.

4.7.2 Tabelle

Ciascuna tabella dovrà essere accompagnata da un identificativo e da una breve descrizione. La tabella inoltre deve essere ordinata, coerente e di facile comprensione al lettore.

⁵<http://gimp.linux.it/>

5 Analisi dei Requisiti

L'Analisi dei Requisiti è fondamentale per la buona riuscita del progetto perciò in questa fase chiarezza, completezza e precisione sono di estrema importanza. Gli analisti dovranno elaborare un documento nel quale saranno presenti tutti i requisiti ricavati dal capitolato oppure da incontri con il committente.

Ogni requisito deve essere ben definito e completo, in modo tale da evitare fraintendimenti e ambiguità.

5.1 Classificazione

Per individuare univocamente un requisito e allo stesso tempo aggiungere informazioni di specifica o di gerarchia, si utilizza il seguente schema di classificazione:



Figura 1: Schema di classificazione

dove:

- **R** è una semplice sigla di requisito.
- **SCOPE** indica l'ambito di un requisito e può assumere i seguenti valori:
 - U (ambito utente)
 - A (ambito amministratore)
 - G (ambito generale)
- **TYPE** esprime il tipo di requisito:
 - FU (funzionale)
 - V (vincolo)
 - Q (qualità)
 - P (prestazionale)
- **PRIORITY** rappresenta la priorità del requisito, in ordine crescente nel seguente elenco:
 - F (facoltativo)
 - D (desiderabile)
 - O (obbligatorio)
- **ID** è un valore numerico incrementale che può essere separato da più punti, mostra la posizione di un requisito all'interno della propria gerarchia.

Esempio di Requisito: RUFUO 2.2 .

5.2 Descrizione grafica

Per una descrizione grafica si utilizza il formalismo UML 2.0, creando diagrammi degli Use Case. Anche in questo caso devono essere identificati da una segnatura qui riportata:

UC ID-NOME

dove:

- UC sta per Use Case
- ID è un valore numerico incrementale di grandezza arbitraria che mostra i livelli di gerarchia
- NOME è un nome univoco per identificare immediatamente un requisito

Esempio: UC 2.1- Invio Stream

5.3 Descrizione didascalica

E' possibile anche una descrizione didascalica con l'aggiunta ad ogni diagramma di Use Case di una descrizione narrativa sintetica, dove non devono mancare questi elementi:

- Fonte
- Attori coinvolti
- Scopo
- Breve descrizione
- Precondizione
- Flusso base degli eventi
- Postcondizione
- Flussi alternativi

6 Progettazione

Durante la fase di progettazione, i progettisti dovranno rispettare le seguenti norme:

6.1 Classi di verifica

Dovranno essere progettate delle classi fittizie ogni qual volta sia possibile ed obbligatoriamente per le classi di maggior peso sul corretto funzionamento dell'architettura. La loro funzione sarà quella di verificare il corretto funzionamento della classe da utilizzare nella Fase di Verifica.

6.2 Diagrammi

Nella Fase di Progettazione, il Progettista dovrà necessariamente, quando ne è richiesto l'utilizzo, utilizzare il linguaggio *UML 2.0* per definire quanto segue:

- **Diagrammi delle classi:** dovrà essere sviluppato un diagramma per ogni classe ritenuta necessaria dal progettista.
- **Diagrammi di attività:** dovrà essere sviluppato un diagramma di flusso ogni qualvolta lo sviluppo logico di un'azione permessa dal sistema in fase di codifica possa lasciare al programmatore una qualsiasi libertà decisionale, la quale potrebbe compromettere il corretto funzionamento del sistema.
- **Diagramma dei package:** dovranno essere sviluppati diagrammi di package per definire, in fase di codifica, la divisione dei moduli da assegnare ad ogni Programmatore, in modo chiaro e ben definito per eliminare il rischio di incomprensioni e sovrapposizione di due moduli.

6.3 Design Pattern

Nella descrizione di un design pattern si dovrà seguire il seguente schema:

- **Descrizione generale:** descrivere brevemente la struttura generale del design pattern
- **Motivazione:** descrivere il motivo della scelta di tale design pattern
- **Contesto applicativo:** elencare i contesti dove il design pattern è stato applicato

6.4 Stile di progettazione

Per rendere uniforme lo stile di progettazione si dovranno seguire le seguenti norme:

- **Annidamento di chiamata:** per evitare problemi di controllo del flusso, l'annidamento di chiamate di metodi dovrà essere limitato: il loro numero dovrà essere minore o uguale a otto.
- **Concorrenza:** nel caso sia ritenuto opportuno l'utilizzo di flussi di esecuzione concorrenti, il Progettista dovrà motivare i benefici portati dall'utilizzo di tale flusso. Egli dovrà quindi valutare se i benefici, dati dalla creazione di un nuovo flusso, siano superiori al costo (di memoria utilizzata, tempo di creazione, possibilità rallentamento del sistema ecc.) necessario alla creazione dello stesso.
- **Ricorsione:** la ricorsione andrà evitata. Qualora si ritenga opportuno l'utilizzo di tale tecnica, il Progettista dovrà spiegare, accuratamente, i benefici portati dal suo utilizzo e dovrà fornire una dimostrazione sulla terminazione.

6.5 Specifica Tecnica

Il documento *Specifica Tecnica* dovrà obbligatoriamente contenere:

- **Design Pattern:** Trattazione dei vari design pattern utilizzati seguendo le modalità del paragrafo 6.3
- **Architettura Generale:** Deve essere rappresentata mediante un grafico ed opportunamente descritta testualmente.
- **Descrizione delle singole componenti:** Ad ogni singola componente sarà assegnata una sezione dove verranno descritte testualmente e rappresentate graficamente. All'interno di ogni sezione dovranno esserci descrizioni testuali per le componenti interne rappresentate attraverso diagrammi di classe. Nelle descrizioni delle classi, dovranno comparire:
 - “Tipo, obiettivo e funzione del componente” che contiene una descrizione generica della classe.
 - “Relazioni con altre componenti” che contiene il contesto d'uso della classe e quindi le relazioni della classe con altre classi dell'architettura.
 - “Attività svolte e dati trattati” che indica i dati soggetti alla classe e le attività che svolge.
- **Tracciamento:** tabelle riportanti il tracciamento tra componenti e requisiti e viceversa.

7 Codifica

Per facilitare la comprensione del codice sorgente tra i membri del team è stato deciso dall'Amministratore che sarà obbligatorio applicare le regole sintattiche contenute nel documento *Java Code Convention* del 1999-04-20⁶. Ogni file sorgente sarà intestato con la seguente parte di codice per identificarne univocamente l'origine, la provenienza così che sia possibile tracciare in modo sicuro ogni modifica effettuata sul suddetto file, descrivendone ciascuna in ordine cronologico.

```
// Name: {Nome del file corrente}  
// Package: {Package di appartenenza}  
// Author: {Autore del file corrente}  
// Date: {Data di creazione del file}  
// Changes:  
// 1) Date : Author : Type  
// ... ..  
// 2) Date : Author : Type  
// ... ..  
//
```

I campi devono assumere i seguenti significati:

- **Name:** indica il nome del file in questione, compresa l'estensione.
- **Package:** è il package del file, compresa gerarchia dei package.
- **Author:** rappresenta l'autore originario del file, nella forma "Nome Cognome"
- **Date:** indica la data di creazione del file. Il formato deve essere *AAAA-MM-GG* con *AAAA* indicante l'anno, *MM* il mese, *GG* il giorno.
- **Changes:** E' lo storico delle modifiche effettuate nel file.
 - *Date* è la data di modifica secondo il formato riportato precedentemente.
 - *Author* è il nome del programmatore che ha effettuato le modifiche, anche in questo caso si usa la forma "Nome Cognome".
 - *Type* rappresenta il tipo di modifica effettuata, tra i quali: **[+]** per aggiunta di attributi o metodi, **[-]** per rimozione di attributi o metodi, **[#]** per la modifica di ottimizzazione o correzione.
In caso di aggiunte o cancellazioni, deve essere presente il nome degli attributi o dei metodi inseriti o eliminati, mentre se si tratta di modifiche deve essere presente il nome dei metodi soggetti a variazioni, seguiti dalla descrizione della modifica e della sua motivazione.

⁶<http://www.oracle.com/technetwork/java/codeconv-138413.html>

7.1 Convenzioni di codifica

Si chiede ai membri del gruppo di rispettare le seguenti regole.

7.1.1 Struttura delle classi

Nella struttura delle classi dovrà essere rispettato il seguente ordine di elementi:

- attributi statici
- attributi di istanza della classe
- costruttori
- metodi

7.1.2 Convenzioni sui nomi

- *Classi*: i nomi delle classi devono essere sostantivi o acronimi significativi, formati solamente da lettere. L'iniziale deve essere maiuscola e se il nome è composto da più parole, anche queste devono avere l'iniziale maiuscola.
- *Interfacce*: regole uguali a quelle per le classi.
- *Metodi*: devono essere dei predicati formati da sole lettere. Se composti da più parole la prima deve avere l'iniziale minuscola e le seguenti iniziale maiuscola.
- *Variabili locali e di istanza*: devono avere nomi brevi e significativi, possibilmente un solo termine. Nel caso siano formati da più parole, quest'ultime devono avere l'iniziale in maiuscolo (ad accezione della prima parola).
- *Costanti e variabili di classe*: devono essere completamente in maiuscolo e se costituite da più parole queste devono essere divise dal carattere '_'.

7.1.3 Struttura del codice Java

Al fine di aumentare l'uniformità dello stile di programmazione e di migliorare la qualità generale del software, dovranno essere rispettate le seguenti norme nel codice delle classi:

- Ogni riga di codice dovrà contenere al massimo un'istruzione.
- I nomi delle variabile dovranno essere in lingua inglese.
- Le variabili dovranno essere dichiarate all'inizio del blocco di appartenenza fatta eccezione per gli indici dei cicli for.
- I commenti in linea saranno utilizzabili solo se il commento non supera gli 80 caratteri, in ogni altro caso dovranno essere usati i commenti multilinea.

- I commenti multilinea devono essere allineati con il resto delle istruzioni del blocco in cui sono inserite.
- Il numero di caratteri per riga dovrà essere pari o inferiore a 80.
- Non si dovranno sfruttare le regole di scoping delle variabili per usare in blocchi interni variabili con il medesimo nome presenti in blocchi esterni.
- Le tabulazioni dovranno essere espanse in 2 caratteri di spazio.
- Non sono ammessi operatori sintetici di assegnamento come `+=`, `-=`...
- Gli operatori di incremento e decremento prefissi e postfissi possono essere utilizzati solo come istruzioni, non come membri di un'espressione.
- Le espressioni booleane non possono essere costituite da più di due condizioni booleane semplici cioè non composte al loro interno da espressioni combinate da operatori booleani.
- Non sono ammesse istruzioni di interruzione del flusso come *break* (ammissibile solo se presente in un costrutto *switch* o *continue*).
- Le parentesi graffe aperte di un blocco di codice relativo ad un metodo od un costrutto condizionale (*if*, *while*, ecc...) dovranno essere collocate alla stessa riga, le parentesi graffe chiuse dovranno essere alla stessa colonna dell'inizio del metodo o costrutto condizionale.

8 Ambiente di lavoro

Con ambiente di lavoro si vuole indicare tutte le infrastrutture hardware e software usate per creare, documentare e testare il prodotto. La lista delle componenti software non è da intendersi completa per tutta la durata del progetto, secondo quanto previsto dal *Piano di Progetto*.

8.1 Ambiente Generale

8.1.1 Sistema operativo

I sistemi operativi utilizzati dai membri del gruppo saranno eterogenei, in quanto l'applicazione che si andrà a creare avrà come caratteristica l'essere multiplatforma. Tale fattore potrebbe essere una facilitazione a trovare nel software prodotto inconsistenze sui differenti sistemi operativi.

Ci si appoggerà principalmente alla distribuzione *Ubuntu 12.10*⁷ per l'esperienza maturata in diversi anni dai membri del gruppo con software basati su quest'ultimo.

⁷<http://www.ubuntu.com/>

8.1.2 Repository

Il repository è fornito di hosting [GitHub](https://github.com)⁸ ed è attivo all'indirizzo

<https://github.com/Hourglass-Labs/MyTalk>

Lo scopo del repository sarà quello di mantenere per ogni revisione una cartella della documentazione aggiornata e una del codice del software presentato. La documentazione in via di redazione sarà presente esclusivamente in file *tex*, mentre quella approvata e definitiva sarà presente in formato *pdf*. Il servizio di hosting *GitHub* fornisce anche strumenti utili per la collaborazione alla progettazione come uno strumento di ticketing, una pagina wiki per avere della documentazione online sempre disponibile ai membri del team e separata da quello che è il repository del progetto.

Albero delle cartelle nel repository

Il repository al primo livello è organizzato in cartelle che corrispondono alle revisioni con il Committente. Al livello inferiore si divide la parte di documentazione del software da quella del sorgente in modo da avere per ogni revisione la documentazione e il software nella stessa cartella. Per quando riguarda la documentazione, le cartelle più annidate dovranno essere quelle che identificano la documentazione approvata e quella in via di redazione mentre per il codice sarà presente una cartella denominata *images* per le immagini da utilizzare nell'applicazione ed altre cartelle che conterranno i sorgenti del codice. Documentazione e codice possono essere rilasciati solo se precedentemente approvati dal Responsabile.

8.2 Ambiente dei documenti

Gli ambienti di scrittura sono a libera scelta per i membri del gruppo, ma sono soggetti alla convenzione di produrre file in formato *tex*⁹ usando un template predefinito all'interno del repository del team (si veda sezione 8.1.2 - *Repository*, per informazioni sullo stesso). La verifica deve essere necessariamente fatta da membri del team estranei ad ogni redazione del documento in questione. Si deve perciò prestare attenzione nella fase di pianificazione del lavoro per non incorrere in incongruenze nella fase di redazione e verifica.

8.2.1 Scrittura dei documenti

La prassi utilizzata dai componenti del gruppo sarà quella di redigere una bozza di ciascun documento in via di sviluppo utilizzando gli strumenti messi a disposizione dall'editor interattivo di *Google Drive*¹⁰, la cui accessibilità sarà assicurata solo ed esclusivamente ai membri del gruppo stesso. *Google Drive* consentirà di effettuare modifiche visionabili in tempo reale da tutti che andranno documentate accuratamente mediante l'utilizzo del tool di note su testo fornito dall'applicazione stessa. Il medesimo tool sarà anche d'ausilio al

⁸<https://github.com/>

⁹Vedi L^AT_EX nel *Glossario*

¹⁰<https://drive.google.com/start>

successivo Revisore per annotare modifiche necessarie ed indicazioni.

Tale versione dei documenti rappresenterà comunque esclusivamente una bozza per una più efficiente e simultanea collaborazione tra i membri del gruppo. Ogni versione del documento che raggiunga uno stato rilevante andrà realizzata in formato *tex* ed opportunamente versionata facendo ausilio del repository.

8.2.2 Verifica ortografica

Si utilizzerà Aspell (0.60.6.1)¹¹ per verificare la correttezza ortografica dei documenti.

8.2.3 Pianificazione

Per creare e aggiornare i diagrammi di Gantt è stato scelto il software Microsoft Project¹² in quanto i membri del gruppo l'hanno preferito ad altre soluzioni (in particolare a *Gantt Project*¹³, considerato tra le possibilità, ma ritenuto inferiore a *Microsoft Project* in quanto presentante un numero minore di funzionalità). Da segnalare che il suddetto software è presente nella versione 2010 su macchina virtuale dotata di sistema operativo Windows 7¹⁴, così da permetterne la portabilità tra i membri del gruppo.

8.2.4 Diagrammi UML

Il software utilizzato per creare diagrammi UML sarà quello fornito online da *LucidChart*¹⁵. *LucidChart* permetterà di lavorare simultaneamente sullo stesso file in modo da velocizzare la codifica dei *diagrammi UML* scelti. È stato molto apprezzato dai membri del team per la sua semplicità d' utilizzo e per la sua immediatezza, oltre alla possibilità di esportare diagrammi in formato png.

¹¹<http://aspell.net/>

¹²<http://www.microsoft.com/project/en-us/project-management.aspx>

¹³<http://www.ganttproject.biz/>

¹⁴<http://windows.microsoft.com/en-US/windows7/>

¹⁵<https://www.lucidchart.com/>

8.3 Ambiente di sviluppo

La memorizzazione persistente di tutto il materiale prodotto dal team risiederà in un repository che userà *Git*¹⁶ per il versionamento e sarà accessibile esclusivamente ai membri del team all'indirizzo <https://github.com/Hourglass-Labs/MyTalk>. Il repository sarà reso pubblico una volta terminato il progetto come suggerito nel paragrafo 6 del Capitolato d'Appalto in questione.

8.3.1 IDE

L'ambiente di sviluppo integrato sarà *Eclipse 4.2 Juno*¹⁷, con l'ausilio di un plug-in **Egit**¹⁸ necessario per interfacciare il repository remoto con l'ambiente di sviluppo *Eclipse*.

8.3.2 Framework

Il framework, entro il quale verrà sviluppato il progetto, sarà GWT. Esso fornisce una serie di strumenti pensati per realizzare web application complesse in grado di girare su un qualsiasi browser moderno, tra le quali può figurare anche l'applicazione che il gruppo si impegna a realizzare. Il Capitolato d'Appalto prevede che venga utilizzato il linguaggio di programmazione Javascript, noto per essere debolmente tipizzato, non ad oggetti ed interpretato. Ciò non garantisce i controlli in fase di compilazione e l'espressività propria di un linguaggio fortemente tipizzato e compilato. GWT risolve queste problematiche dato che consente di compilare codice sorgente scritto in Java in codice Javascript ottimizzato ed equivalente, mantenendo sempre la possibilità di scrivere codice Javascript nativo. In questo modo è possibile usare strumenti dedicati al linguaggio Java, ma allo stesso tempo di sfruttare le librerie Javascript.

8.3.3 Strumenti per il versionamento

Per il versionamento è stato scelto il software distribuito *Git* fornito con il servizio di hosting *GitHub*¹⁹. *Git* fornisce diversi vantaggi rispetto ad altri software per il versionamento:

- Le maggior parte delle operazioni con *Git* non necessitano di informazioni presenti in remoto.
- La cronologia del progetto è sempre disponibile nella piattaforma di lavoro in locale, senza necessità di collegamenti esterni.

¹⁶<http://git-scm.com/>

¹⁷<http://www.eclipse.org/>

¹⁸<http://www.eclipse.org/egit/>

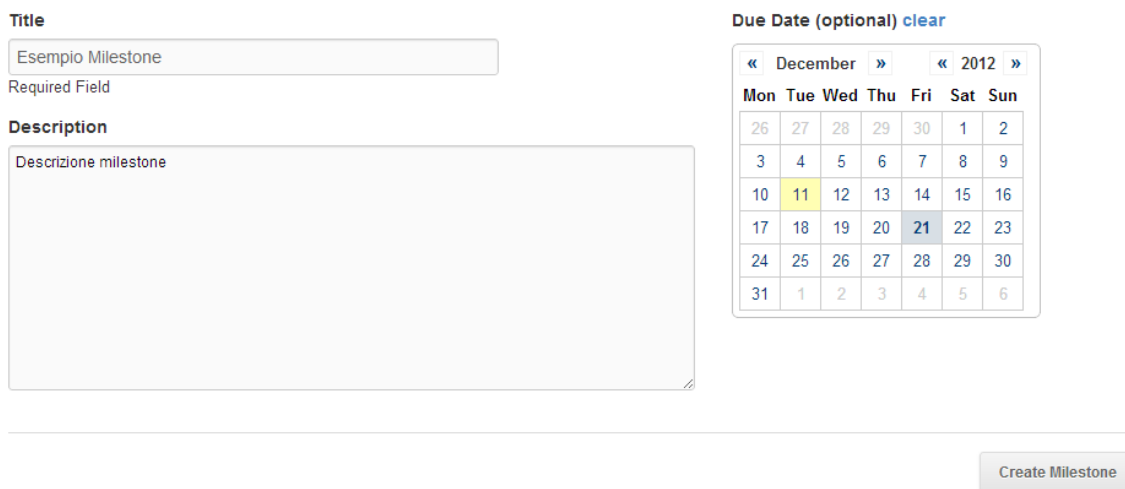
¹⁹<https://github.com/>

8.3.4 Gestione milestones e ticketing

Per la gestione di milestones, ticketing e tracking il gruppo utilizzerà il servizio fornito da *GitHub*. La gestione andrà eseguita nel seguente modo:

8.3.4.1 Creazione di milestones Il Responsabile si occuperà di creare, e gestire, le milestones del progetto. I dettagli sulle milestone sono presenti nel *Piano_di_Progetto.pdf*. Il Responsabile dovrà creare una milestone come in figura 2. I campi da definire saranno:

1. Titolo della milestone.
2. Descrizione della milestone.
3. Data di completamento.



Title

Esempio Milestone

Required Field

Description

Descrizione milestone

Due Date (optional) clear

« December » « 2012 »

Mon	Tue	Wed	Thu	Fri	Sat	Sun
26	27	28	29	30	1	2
3	4	5	6	7	8	9
10	11	12	13	14	15	16
17	18	19	20	21	22	23
24	25	26	27	28	29	30
31	1	2	3	4	5	6

Create Milestone

Figura 2: Esempio definizione di una milestone

8.3.4.2 Creazione ticket Una volta creata una milestone, il Responsabile dovrà creare i ticket, per ogni compito, contenuti nella milestone stessa. La creazione di un ticket andrà eseguita come illustrato nella figura 3. La creazione del ticket dovrà seguire i seguenti passi:

1. Selezione milestone.
2. Assegnamento titolo del ticket.
3. Aggiunta tipo di ticket (mediante le labels).
4. Assegnamento al membro del gruppo.
5. Aggiunta commento.

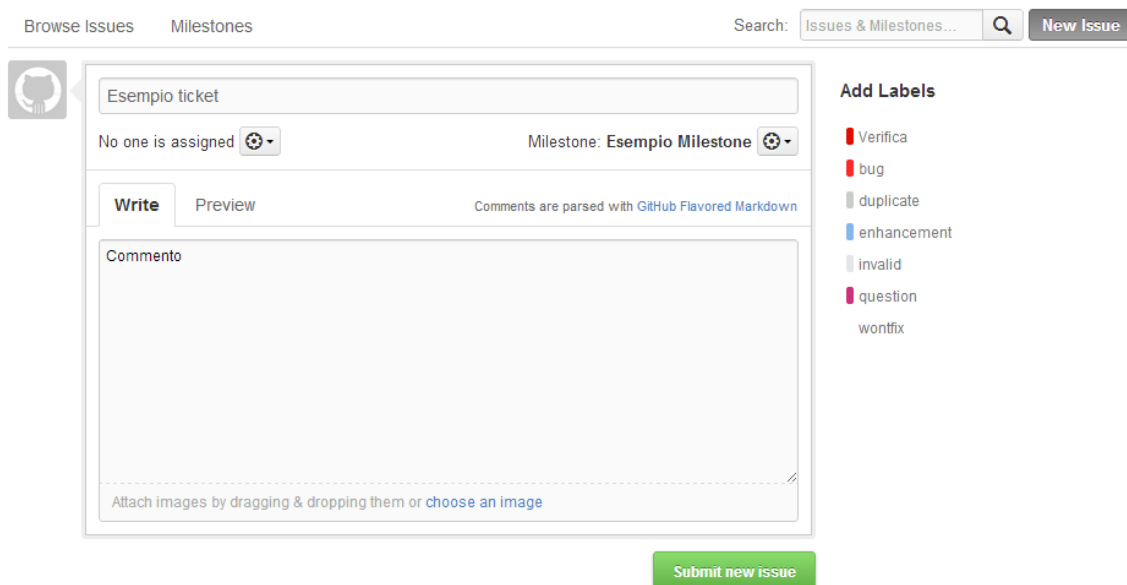


Figura 3: Esempio creazione ticket

8.3.4.3 Chiusura ticket Ogni volta che un membro completerà il ticket a lui assegnato, esso si dovrà prendere il compito di chiuderlo.

8.3.4.4 Creazione ticket di verifica Una volta che un ticket sarà stato completato, il Responsabile si dovrà occupare della creazione del ticket di verifica dello stesso. Il metodo di creazione del ticket di verifica è lo stesso di quello descritto in precedenza, tranne per il fatto che il tipo di ticket dovrà per forza essere di tipo Verifica.

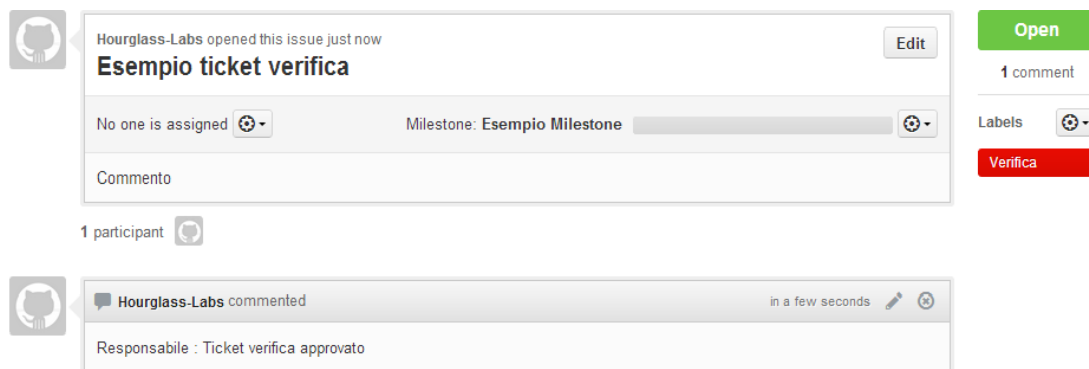


Figura 4: Esempio ticket di verifica

8.3.4.5 Esecuzione verifica e creazione ticket di correzione Qualora il Verificatore riscontri un grave errore nel compito che sta verificando, se lo ritiene necessario, potrà creare un ticket di correzione dell'errore. Tale ticket dovrà essere risolto dal membro del gruppo inizialmente assegnato al compito che il Verificatore sta verificando. Una volta che il Responsabile approva il ticket, tramite l'aggiunta di una risposta al ticket (vedi figura 4), il membro del gruppo che ha ricevuto il ticket dovrà eseguirlo. Si continuerà la procedura dal punto 8.3.4.3 *Chiusura ticket*.

8.4 Ambiente di Verifica e Validazione

Al fine di evitare conflitti di interessi nello svolgimento delle attività di verifica e di validazione, a fronte dell'obbligo di rotazione dei ruoli di lavoro all'interno del gruppo, i ruoli di Responsabile, Verificatore e Redattore devono essere impersonati da tre persone distinte ad ogni rilascio di un documento.

8.4.1 Tracciamento dei requisiti

Il Tracciamento dei Requisiti viene effettuato mediante un database *MySQL*²⁰, installato nel server remoto del team e amministrato tramite *phpMyAdmin*²¹. Per mezzo di questo strumento, il Tracciamento dei Requisiti e relativi *UseCase*, è realizzabile in modo flessibile ed efficiente, data la possibilità di modellare il database in base alle esigenze del team. Inoltre alla pagina

²⁰<http://www.mysql.com/>

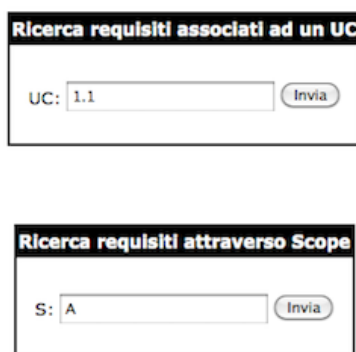
²¹http://www.phpmyadmin.net/home_page/index.php

<http://hourglabs.org/login.php>

dopo aver effettuato l'autenticazione con delle credenziali²² fornite dall'Amministratore di progetto, si possono tracciare i requisiti tramite un'interfaccia web che si relaziona con il database citato in precedenza: oltre ad un visualizzazione di una tabella per il tracciamento, ne viene generato il codice per un eventuale file \LaTeX .

hourglass

Requirements Tool



Ricerca requisiti associati ad un UC

UC:

Ricerca requisiti attraverso Scope

S:

Figura 5: Interfaccia per il Tracciamento dei Requisiti

²²Account per il committente:
Username: committente
Password: tracciamento

hourglass

Requirements Tool

	CODICE REQUISITO	DESCRIZIONE	FONTE
UC 1.1	RUFUD 1.3	L'Utente deve poter decidere se condividere o no l'audio ricevuto dal microfono.	Interno
	RUFUD 1.4	L'Utente deve poter decidere se condividere o no il video inviato dalla webcam.	Interno
	RUFUD 1.5	L'Utente deve poter condividere una chat con un altro Utente.	Capitolato
	RUFUD 1.6	L'Utente deve poter instaurare una chiamata con più Utenti, collegate al sistema, contemporaneamente.	Capitolato
	RUFUD 1.6.1	L'Utente deve poter invitare un nuovo per Utente in una chiamata già in atto.	Interno
	RUFUD 1.7	Un Utente deve poter ricevere una chiamata.	Capitolato
	RUFUD 1.8	L'Utente deve poter vedere le statistiche della connessione.	Capitolato

Codice file tex

```
\begin{longtable}[m]{2.5cm}{2.5cm} \hline \rowcolor{grigio}
{\bf USECASE} & {\bf REQUISITO}
\endhead
1.1 & RUFUD 1.3
RUFUD 1.4
RUFUD 1.5
RUFUD 1.6
RUFUD 1.6.1
RUFUD 1.7
RUFUD 1.8
\
\end{longtable}
```

Figura 6: Risultato del tracciamento dello UC 1.1

hourglass

Requirements Tool

CODICE REQUISITO	DESCRIZIONE	FONTE
RAFUD 19	L'Amministratore deve avere accesso alle funzionalità di gestione del Sistema.	Interno
RAFUD 19.1	L'Amministratore deve potersi autenticare per avere accesso alle funzioni riservate.	Interno
RAFUD 20	L'Amministratore può accedere alla lista degli Utenti registrati nel sistema.	Interno
RAFUD 21	L'Amministratore può selezionare un Utente particolare dalla lista degli Utenti registrati.	Interno
RAFUD 21.1	L'Amministratore può modificare le informazioni registrate dell'Utente selezionato.	Interno
RAFUD 21.2	L'Amministratore può controllare le azioni eseguite dall'Utente selezionato.	Interno

Codice file tex

```
\begin{center}
\begin{tabular}{c}{0.5\textwidth} c)
\rowcolor{grigio}
\hline
\HourStyle \textbf{CODICE REQUISITO} & \HourStyle \textbf{DESCRIZIONE} & \HourStyle \textbf{FONTE} \\
\rowcolor{white}
\hline
RAFUD 19 & L'Amministratore deve avere accesso alle funzionalità di gestione del Sistema. & Interno \\
\rowcolor{grigio}
\hline
RAFUD 19.1 & L'Amministratore deve potersi autenticare per avere accesso alle funzioni riservate. & Interno \\
\hline
RAFUD 20 & L'Amministratore può accedere alla lista degli Utenti registrati nel sistema. & Interno \\
\hline
RAFUD 21 & L'Amministratore può selezionare un Utente particolare dalla lista degli Utenti registrati. & Interno \\
\hline
RAFUD 21.1 & L'Amministratore può modificare le informazioni registrate dell'Utente selezionato. & Interno \\
\hline
RAFUD 21.2 & L'Amministratore può controllare le azioni eseguite dall'Utente selezionato. & Interno \\
\end{tabular}
\end{center}
```

Figura 7: Risultato ricerca dei requisiti attraverso l'ambito Amministratore

8.4.2 Analisi Statica

Gli strumenti stabiliti dal team per effettuare l'analisi statica sono i seguenti:

- **FindBugs²³ (2:0):** Strumento in grado di determinare staticamente i più comuni errori nella scrittura di codice Java.
- **Metrics²⁴ (1.3.6):** Strumento in grado di calcolare delle metriche software statiche su codice Java.

8.4.3 Analisi Dinamica

Gli strumenti che verranno utilizzati per l'analisi dinamica, sono:

- **JUnit²⁵ (4.11):** Strumento per eseguire test di unità su codice scritto in Java.
- **EclEmma²⁶ (2.2.1):** Strumento per per analizzare e constatare la copertura del codice nella normale esecuzione e durante l'esecuzione di test.
- **Selenium IDE ²⁷ (2.0.0):** Strumento per eseguire test di sistema, si tratta di un estensione per *Mozilla Firefox* che permette l'automazione del browser per eseguire azioni in pagine web.

²³<http://findbugs.sourceforge.net/>

²⁴<http://metrics.sourceforge.net/>

²⁵<http://www.junit.org/>

²⁶<http://www.eclemma.org/>

²⁷<http://docs.seleniumhq.org/>