



Norme Di Progetto

Informazioni sul documento

Titolo documento	Norme Di Progetto
Versione attuale	v4.0.0
Data versione attuale	2012/04/04
Data creazione	2011/12/01
Redazione	Umberto Dall'Est Stefano Faoro Giacomo Lorigiola
Revisione	Antonio Pretto [v4.0.0] Luca Guerra [v3.0.0] Umberto Dall'Est [v2.0.0] Andrea Zironda [v1.0.0]
Approvazione	Andrea Zironda
Stato documento	Formale
Uso	Interno
Distribuito da	SevenFold
Destinato a	SevenFold

Sommario

Questo documento descrive le norme e le convenzioni che verranno seguite dal gruppo durante lo sviluppo del progetto.

Diario delle modifiche

Versione	Data	Autore	Modifiche
v4.0.0	2012/04/04	Andrea Zirona	Approvazione e rilascio quarta versione
v3.3.0	2012/03/24	Giacomo Lorigiola	Inserita sezione Test di unità e integrazione
v3.2.0	2012/03/23	Giacomo Lorigiola	Aggiornate norme di codifica e sezione Verifica
v3.1.0	2012/03/19	Giacomo Lorigiola	Creazione sezione Norme sui diagrammi UML
v3.0.0	2012/03/15	Andrea Zirona	Approvazione e rilascio terza versione
v2.1.0	2012/02/13	Stefano Faoro	Aggiornate sezioni "Verifica e Approvazione" e "Verifica e Validazione"
v2.0.0	2012/01/30	Giacomo Lorigiola	Approvazione e rilascio seconda versione
v1.2.0	2012/01/24	Stefano Faoro	Aggiunte convenzioni sui diagrammi di sequenza
v1.1.1	2012/01/23	Stefano Faoro	Correzioni varie
v1.1.0	2011/12/18	Stefano Faoro	Aggiornata sezione "Nomenclatura"
v1.0.0	2011/12/18	Antonio Pretto	Approvazione e rilascio prima versione
v0.8.0	2011/12/08	Stefano Faoro	Aggiunta sezione 7.4 "Norme sui test"
v0.7.0	2011/12/07	Stefano Faoro	Aggiornata la sezione "Ambiente di lavoro"
v0.6.3	2011/12/07	Umberto Dall'Est	Inserito cap7 "Norme di codifica"
v0.6.2	2011/12/06	Stefano Faoro	Aggiornata la sezione "Documentazione"
v0.6.1	2011/12/06	Umberto Dall'Est	Aggiornata la sezione "Norme sui requisiti"
v0.6.0	2011/12/03	Umberto Dall'Est	Completata sezione "Norme di sviluppo"
v0.5.0	2011/12/03	Umberto Dall'Est	Completata sezione "Norme sui requisiti"
v0.4.0	2011/12/03	Stefano Faoro	Completata sezione "Documentazione"
v0.3.0	2011/12/02	Umberto Dall'Est	Completata sezione "Ambiente di lavoro" e "Assegnazione dei lavori"
v0.2.0	2011/12/01	Stefano Faoro	Completate sezioni "Introduzione" e "Comunicazioni e convocazioni"
v0.1.0	2011/12/01	Umberto Dall'Est	Creata struttura documento

Indice

1	Introduzione	5
1.1	Scopo del documento	5
1.2	Glossario	5
2	Comunicazioni e Convocazioni	6
2.1	Comunicazioni	6
2.1.1	Interne	6
2.1.2	Esterne	6
2.2	Convocazioni	6
2.2.1	Interne	6
2.2.2	Esterne	6
3	Ambiente di Lavoro	7
3.1	Sistema Operativo	7
3.2	Controllo di versione	7
3.3	Interfaccia web per controllo di versione	7
3.4	Project manager	7
3.5	Documentazione condivisa	7
3.6	Tracker	7
3.7	Comunicazione remota	7
3.8	Documentazione	7
3.9	Diagrammi UML	8
3.10	Diagrammi di Gantt	8
3.11	Guide all'Ambiente di Lavoro	8
4	Assegnazione di Task e Ticket	9
4.1	Assegnazione Task	9
4.2	Ticketing Errori	9
5	Documentazione	10
5.1	Identificazione di versione	10
5.2	Nomenclatura	10
5.3	Informazioni sul Documento	10
5.4	Registro delle Modifiche	11
5.5	Norme di Stesura	11
5.5.1	Formato	11
5.5.2	Stato documento	11
5.5.3	Organizzazione repository	11
5.5.4	Denominazione Diagrammi e Tabelle	11
5.5.5	Convenzioni	11
5.5.6	Glossario	11
5.6	Verifica e Approvazione	12
5.6.1	Procedure	12
5.6.2	Strumenti	12
6	Norme sui Requisiti	13
6.1	Classificazione dei Requisiti	13
7	Norme sui diagrammi UML	14
7.1	Versione UML	14
7.2	Ambiente di lavoro	14
7.3	Diagrammi Use Case	14
7.3.1	Classificazione degli Use Case	14
7.3.2	Descrizione	14
7.4	Diagrammi di sequenza	15

7.5	Diagrammi delle attività	15
7.6	Diagrammi delle classi e dei package	15
8	Norme di Sviluppo	16
8.1	Norme sul Rendiconto Ore	16
8.2	Norme di Progettazione	16
8.3	Norme di Codifica	16
8.3.1	Template	16
8.3.2	Convenzioni	16
8.3.3	Lingua	16
8.3.4	Costrutti	16
8.3.5	Commenti	16
8.3.6	Intestazione file	17
8.3.7	Versionamento dei file	17
8.3.8	Tabella delle modifiche	17
9	Verifica e Validazione	17
9.1	RoR	17
9.2	Android	18
9.3	Procedure	18
10	Test di unità e integrazione	18
10.1	Stesura dei test	18

1 Introduzione

1.1 Scopo del documento

Lo scopo di questo documento è quello di delineare le norme e le convenzioni che verranno seguite dal gruppo SevenFold nel corso di tutte le attività di progetto quali comunicazioni, stesura della documentazione, codifica e uso degli strumenti.

Questo documento è disponibile a tutti i componenti di SevenFold che dovranno seguirne le convenzioni al fine di ottimizzare il lavoro di gruppo e l'utilizzo degli strumenti.

Durante lo svolgimento delle attività sarà il responsabile a verificare l'applicazione delle norme di seguito definite e ad approvare nuove convenzioni proposte dai componenti del gruppo.

1.2 Glossario

Per evitare ridondanze tutti i termini e gli acronimi presenti nel seguente documento che necessitano di definizione saranno seguiti da una "g" ad apice (E.g. User^g) alla loro prima occorrenza e saranno riportati in un documento esterno denominato *Glossario.pdf*.

Tale documento accompagna e completa il presente e consiste in un listato ordinato di termini e acronimi con le rispettive spiegazioni.

2 Comunicazioni e Convocazioni

2.1 Comunicazioni

2.1.1 Interne

I contatti di ogni membro del gruppo (e-mail, telefono, etc.) e le discussioni sono disponibili nella documentazione condivisa (nel project manager^g).

Gli incontri sul progetto avvengono o durante una convocazione interna orale o tramite conferenza remota.

2.1.2 Esterne

Tutte le comunicazioni con il proponente devono essere approvate dal responsabile di progetto. Ogni comunicazione deve essere salvata e pubblicata nella documentazione condivisa.

2.2 Convocazioni

2.2.1 Interne

Tutti i componenti del gruppo sono tenuti a partecipare agli incontri di gruppo effettuati durante tutti i giorni lavorativi presso i laboratori di via Paolotti a Padova, a meno di giustificazioni motivate.

Le convocazioni agli incontri obbligatori verranno pubblicate sul project manager e notificate informalmente tra i componenti.

2.2.2 Esterne

Sono previsti almeno due sessioni di brainstorm^g con il proponente durante lo sviluppo del progetto.

Se ritenuto necessario, prevediamo la possibilità che il responsabile indichi ulteriori incontri in una qualsiasi delle fasi di avanzamento del progetto.

Alla fine di ogni incontro il responsabile fissa un incontro obbligatorio durante il quale vengono formalizzate le user story^g registrate durante i brainstorm e programmato l'avanzamento del progetto.

3 Ambiente di Lavoro

3.1 Sistema Operativo

Il sistema operativo^g di riferimento è Linux.
Per evitare al più possibile le dipendenze dalle piattaforme sono state tuttavia utilizzate per la maggior parte applicazioni web e in mancanza di queste sono stati scelti gli applicativi supportati dal maggior numero di piattaforme.

3.2 Controllo di versione

È stato scelto git^g per la sua bassa dipendenza da connessione di rete, la semplicità d'uso e per l'interoperabilità tra i maggiori sistemi operativi.

3.3 Interfaccia web per controllo di versione

Si è preferito mantenere esterna la gestione delle repository^g Github^g per non incorrere nelle problematiche di installazione e mantenimento sistemistico presso il server del gruppo.
È stata riscontrata inoltre utilissima la disponibilità di un'interfaccia web per monitorare il lavoro del team. Tra le varie opzioni abbiamo scelto GitHub^g per la sua immediatezza.

3.4 Project manager

La maggior parte dei software commerciali in uso sono sovradimensionati per le necessità di progetto. Tra le varie opzioni si è scelto Apollo^g per la sua focalizzazione sul singolo progetto e per la semplicità e l'immediatezza dell'interfaccia.
Esso verrà utilizzato per l'esplosione delle attività in task assegnabili alle persone e organizzabili in gruppi.

3.5 Documentazione condivisa

Vista la quantità ridotta di informazioni condivise utili alle attività di gruppo, verranno utilizzate le funzionalità offerte dal project manager per quanto riguarda la documentazione testuale.
Per quella binaria invece deve essere aperto uno spazio web di libero accesso con relativi link ipertestuali nello spazio precedentemente menzionato.

3.6 Tracker

Durante le ultime due iterazioni si potrebbe avere la necessità di procedure di project management più snelle, in quanto i task saranno rapidi nel loro completamento.
Allora potrebbe essere più comodo l'utilizzo di un software meno potente e più libero di Apollo.
Ci si riserva la libertà di usare Pivotal Tracker^g, che rispetta le precedenti necessità, basandoci sull'esperienza accumulata durante le altre iterazioni.

3.7 Comunicazione remota

Si ritiene utile la possibilità di riunire il gruppo in discussione anche da casa.
Per rendere comoda questa soluzione è stato installato il server per conferenze remote Teamspeak^g per discutere senza avere la necessità di incontrarsi.
Il software permette rapide conversazioni di gruppo senza essere legati a servizi esterni ed è già conosciuto dalla maggior parte dei componenti.

3.8 Documentazione

La documentazione formale viene scritta in linguaggio LaTeX^g utilizzando l'IDE TexMaker^g e successivamente esportata in formato PDF usando il relativo compilatore pdfLaTeX^g.

3.9 Diagrammi UML

Per la realizzazione dei diagrammi UML si è scelto di utilizzare il servizio online LucidChart^g tramite il quale è possibile realizzare collaborativamente i diagrammi.

3.10 Diagrammi di Gantt

Per la realizzazione dei diagrammi di Gantt si è scelto di utilizzare il servizio online TeamGantt^g tramite il quale, come per il precedente, è possibile realizzare collaborativamente i diagrammi.

3.11 Guide all'Ambiente di Lavoro

Per chi avesse problemi ad installare, configurare o utilizzare uno degli strumenti dell'ambiente di sviluppo sono state redatte e inserite nella documentazione condivisa delle semplici ma esaustive guide step-to-step.

4 Assegnazione di Task e Ticket

4.1 Assegnazione Task

La creazione e l'assegnazione dei lavori tramite il project manager viene svolta da un responsabile. Sarà quest'ultimo a coordinare le attività all'interno del gruppo. Per facilitare l'avanzamento rapido dei progressi sarà tuttavia possibile, al completamento di tutti i lavori destinati ad un individuo, prendersi in carico di uno dei lavori non ancora assegnati. Sarà il project manager a tenere informati il responsabile e i restanti membri del gruppo dell'avanzamento delle attività svolte o in svolgimento.

4.2 Ticketing Errori

La segnalazione degli errori riscontrati nei documenti, testuali o di codice che siano, avviene tramite il sistema del ticketing⁹ mediante la creazione e l'assegnazione di task marcati come ticket sul project manager.

Ogni ticket deve fornire:

- Titolo significativo scritto nel formato "ChangeRequest: Titolo"
- Autore
- Oggetto della verifica, specificandone la sezione
- Motivazione della creazione del ticket
- Priorità
- Valutazione complessità

Anche se verosimilmente sono i verificatori ad occuparsi dei processi di verifica, la segnalazione degli errori può avvenire da parte di un qualsiasi membro del gruppo. Nell'eventualità di errori minori riscontrati nel codice si creano invece issue⁹ in GitHub che sono ricevuti direttamente dal redattore del documento considerato errato con una breve descrizione dell'errore trovato.

5 Documentazione

5.1 Identificazione di versione

La versione dei documenti viene identificata con la sigla:

vX.Y.Z

dove le seguenti sigle indicano:

- X:** numero di versione primario che viene incrementato solo alla consegna dei documenti in una delle revisioni previste durante lo svolgimento del progetto. L'incremento di x impone l'azzeramento degli indici minori.
- Y:** numero di versione secondario che viene incrementato quando si svolgono operazioni rilevanti quali aggiunte e rimozioni di paragrafi nel documento o modifiche sostanziose. L'incremento di y impone l'azzeramento di z.
- Z:** numero di versione terziario che viene incrementato in caso di piccole modifiche quali, per esempio, la correzione ortografica o l'indentazione del testo. L'indice z può non essere presente nel nome del file.

5.2 Nomenclatura

I nomi di tutti i documenti interni o esterni devono rispettare la seguente nomenclatura:

NomeDocumento_Versione.est

Il nome del file va scritto privo di spazi e con l'iniziale di ogni parola maiuscola.

La versione del documento deve essere conforme alle norme descritte nel paragrafo Identificazione di versione.

Per i documenti in consegna alle revisioni prevediamo di sostituire l'indicatore completo di versione con il solo numero di versione primario(e.g. NormeDiProgetto_v3.pdf).

5.3 Informazioni sul Documento

Nel frontespizio di ogni documento devono essere presenti le seguenti informazioni:

- Titolo documento
- Versione attuale
- Data versione attuale
- Data creazione
- Redazione: autori del documento
- Revisione: revisionatore del documento
- Approvazione: approvatore del documento
- Stato documento: instabile, stabile, verificato o formale
- Uso: interno o esterno
- Distribuito da
- Destinato a

5.4 Registro delle Modifiche

Ad ogni documento interno o esterno deve essere associato un registro delle modifiche, che verrà visualizzato sotto forma tabulare all'inizio del documento.

A seguito di una modifica al documento deve essere aggiornata la tabella delle modifiche specificando in ordine:

- La versione indicata secondo la convenzione del paragrafo 5.1
- La data della modifica
- L'autore della modifica
- Una rapida spiegazione verbale delle modifiche effettuate

5.5 Norme di Stesura

5.5.1 Formato

Per la stesura di tutti i documenti di progetto deve essere utilizzato il linguaggio LaTeX.

I file .tex verranno poi esportati in formato pdf attraverso il relativo compilatore.

Tutti i documenti devono essere redatti a partire dal template disponibile nella repository di gruppo. Il nome del file è `documentTemplate.tex` ed esso garantisce la struttura alla quale ogni documento deve essere conforme.

5.5.2 Stato documento

Durante la stesura di un documento questo viene contrassegnato come "instabile" attraverso lo stato del documento. Nel momento in cui il redattore riterrà il proprio lavoro completato e pronto per la fase di revisione dovrà contrassegnare il documento con lo stato "stabile".

5.5.3 Organizzazione repository

Ogni documento deve essere disponibile nella repository.

Ad ognuno di essi deve essere dedicata una cartella specifica denominata con la sigla del documento nella quale verranno riposti i relativi file .tex ed eventuali file interni quali immagini etc. dovranno essere suddivisi in rispettive sottocartelle specifiche riportanti nomi significativi.

5.5.4 Denominazione Diagrammi e Tabelle

Ad ogni diagramma o tabella presente all'interno dei documenti deve essere associata una didascalia che sarà composta da:

- numero progressivo per garantire una buona rintracciabilità del diagramma all'interno del documento
- codice del diagramma che ne identifica la tipologia (Use-Case etc.)

Tutte le tabelle o diagrammi vanno accompagnati dal titolo.

5.5.5 Convenzioni

Formato delle date Le date presenti nei documenti devono essere scritte nel formato: YYYY/MM/DD.

5.5.6 Glossario

Tutti i documenti faranno riferimento ad un unico glossario per la spiegazione dei termini tecnici.

All'interno di ogni documento i termini presenti nel glossario devono essere segnalati con la presenza di un apice ^g. Il glossario deve essere sintetico e contenere i termini in ordine alfabetico. Ogni volta che si incontra una nuova parola chiave nella stesura dei documenti deve essere definita la relativa voce.

5.6 Verifica e Approvazione

5.6.1 Procedure

Ogni documento redatto contrassegnato come "stabile" deve essere verificato da un verificatore che ne controlla correttezza e completezza.

Se il documento viene ritenuto valido il suo stato viene impostato a "verificato" e dovrà attendere l'approvazione del responsabile di progetto. In caso contrario il documento verrà rifiutato impostando il suo stato nuovamente a "instabile".

Al responsabile di progetto si richiede di approvare ogni documento che acquisisca lo stato "verificato". Nel caso che il documento venga approvato questo passa in stato "formale" altrimenti regredirà nuovamente allo stato "instabile".

Una volta che il documento riceve l'approvazione del responsabile ne viene incrementato l'indice primario di versione e il documento è pronto per essere distribuito.

La comunicazione tra i redattori, i verificatori e i responsabili verrà regolata tramite l'assegnazione dei task e il ticketing degli errori descritti nel paragrafo "Assegnazione di Task e Ticket" di questo documento.

5.6.2 Strumenti

Ogni documento deve essere controllato utilizzando lo strumento di "Verifica Ortografica" presente nel menu "Modifica" di TexMaker direttamente sul file .tex posto a verifica.

6 Norme sui Requisiti

6.1 Classificazione dei Requisiti

I requisiti vengono organizzati in forma gerarchica. Ogni requisito può quindi avere più sotto-requisiti.

Quando ognuno di questi sotto-requisiti viene soddisfatto, allora il requisito di livello superiore viene soddisfatto.

La notazione utilizzata è la seguente:

$\langle F|P|Q|V|I \rangle \langle o|d|z \rangle - X \langle Y \langle Z \rangle \rangle$

dove le seguenti sigle indicano:

F: requisito funzionale

P: requisito prestazionale

Q: requisito di qualità

V: requisito di vincolo

I: requisito di interfacciamento

o: obbligatorio

d: desiderabile

z: opzionale

X: requisito di primo livello

Y: sotto-requisito

Z: sotto-requisito di un sotto-requisito

I campi Y e Z possono essere assenti.

7 Norme sui diagrammi UML

Di seguito vengono indicate le norme relative alla stesura dei diagrammi UML.

7.1 Versione UML

Per la realizzazione di tutti i diagrammi UML si è scelto di adottare la versione UML 2.0.

7.2 Ambiente di lavoro

Per la realizzazione dei diagrammi UML viene utilizzato il servizio online LucidChart tramite il quale è possibile realizzare collaborativamente i diagrammi. In questo modo il materiale potrà essere facilmente condiviso e acceduto da tutti i membri del gruppo.

7.3 Diagrammi Use Case

7.3.1 Classificazione degli Use Case

Gli use case vengono organizzati in forma gerarchica.

La notazione utilizzata è la seguente:

Uc X<.Y<.Z> >

dove le seguenti sigle indicano:

X: use case di primo livello

Y: use case di secondo livello

Z: use case di terzo livello

I campi Y e Z possono essere assenti.

7.3.2 Descrizione

Per ogni scenario individuato dovrà essere indicata una precisa descrizione che aiuterà nella comprensione dello scenario.

La descrizione dovrà contenere le seguenti specifiche:



- Attori coinvolti
individua tutti gli attori coinvolti nello specifico scenario
- Precondizione
indica lo stato iniziale dello scenario
- Postcondizione
indica lo stato finale dello scenario
- Scenario principale
descrizione aggiuntiva che serve per spiegare in maniera discorsiva lo scenario descritto dal diagramma

7.4 Diagrammi di sequenza

Grazie all'utilizzo della versione UML 2.0 è possibile sviluppare attraverso i diagrammi di sequenza, concetti anche ad livello e non unicamente sequenze basate sull'interazione tra le istanze delle classi.

Per la stesura dei diagrammi di sequenza si dovranno utilizzare le seguenti convenzioni:

Convenzioni nell'uso delle tipologie di frecce:

- Messaggi sincroni : 
indica che il chiamante resta in attesa della risposta
- Messaggi asincroni: 
indica che il chiamante non rimane in attesa della risposta

Convenzione nel passaggio dei dati all'interno di metodi:
verrà utilizzata come convenzione per il passaggio dei parametri il "metodo classico" che prevede la scrittura dei parametri direttamente all'interno delle parentesi del metodo, scartando così il metodo "dei girini" ritenuto più verboso e meno immediato.

7.5 Diagrammi delle attività

Per la realizzazione dei diagrammi di attività inerenti al progetto non sono state rilevate particolari convenzioni da utilizzare in aggiunta alle norme dello standard UML 2.0 .

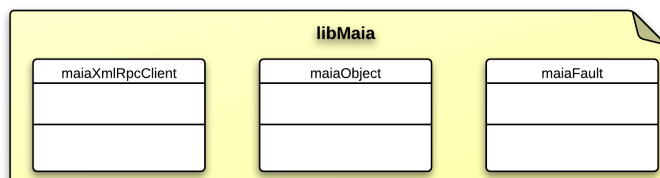
7.6 Diagrammi delle classi e dei package

Per i diagrammi dei package, dovrà essere indicato per ogni package il nome identificativo.

Sia per i diagrammi delle classi che dei package devono essere indicati i design patter utilizzati in modo da essere facilmente riconoscibili all'interno dei diagrammi:
nel caso di interi package che rappresentano componenti di design pattern (e.g. MVC), questi devono essere evidenziati con colori differenti e dovrà essere indicato il design pattern specifico;
nel caso di singole classi che rappresentano un design pattern (e.g. Singleton), deve essere indicato sopra il nome della classe il nome del rispettivo design pattern utilizzato.

Tutte le classi appartenenti a librerie utilizzate dovranno essere racchiuse all'interno di rettangolo di colore giallo indicanti il nome della libreria, come illustrato nell'immagine

Figura 1: Esempio convenzione librerie



8 Norme di Sviluppo

8.1 Norme sul Rendiconto Ore

Il project manager in uso prevede un sistema di rendicontazione ore e va obbligatoriamente utilizzato da tutti i componenti.

8.2 Norme di Progettazione

I progettisti dovranno seguire le seguenti indicazioni:

- Seguire, per ognuna delle fasi, le best practice generali e decise dal gruppo.
- Applicare, dove possibile, i Design Pattern più adatti.
- Utilizzare UML per la realizzazione dei diagrammi.
- Realizzare, dove possibile, classi di verifica del codice.

8.3 Norme di Codifica

Il codice va scritto rispettando scrupolosamente le linee guida del linguaggio in uso. Vanno rispettate obbligatoriamente le regole di denominazione presenti nella documentazione architetturale ed altre eventuali norme decise al momento di scelta di linguaggio.

8.3.1 Template

Per classi, interfacce ed altre strutture saranno resi disponibili template per la creazione di ogni tipologia di file necessaria alla compilazione.

Il programmatore dovrà rispettare ogni richiesta e regola dettata all'interno del template.

8.3.2 Convenzioni

- **Ruby**: per quanto riguarda il codice scritto in Ruby il gruppo è tenuto ad aderire alle convenzioni definite all'indirizzo <http://rails.nuvvo.com/lesson/5017-ruby-coding-convention>.
- **RoR**: per quanto riguarda il codice scritto con Rails il gruppo è tenuto ad aderire alle convenzioni definite all'indirizzo http://guides.rubyonrails.org/contributing_to_ruby_on_rails.html#follow-the-coding-conventions.
- **Android**⁹: per quanto riguarda la stesura del codice relativo all'applicazione mobile Android i programmatori sono tenuti ad aderire alle convenzioni definite nel seguente documento: <http://java.sun.com/docs/codeconv/CodeConventions.pdf>.

8.3.3 Lingua

La lingua in utilizzo per l'assegnazione di nomi è unicamente l'inglese.

8.3.4 Costrutti

Sono tassativamente vietati i costrutti più complessi del linguaggio, anche a spese prestazionali. Per la generalità dei casi preferiamo la leggibilità e la semplicità del codice per rendere quest'ultimo più facilmente verificabile e mantenibile.

8.3.5 Commenti

- **RoR**: Utilizzo della gemma annotate per descrivere in dettaglio attributi e metodi di una classe.
- **Android**: Il codice presente nel sottosistema Android verrà documentato come JavaDoc.

I commenti nel codice sono redatti in maniera minimale, ma pienamente espressiva. Vanno inseriti negli spazi previsti dal template in utilizzo.

8.3.6 Intestazione file

Ogni file derivante dalla codifica dovrà presentare un'intestazione che rispecchi la seguente struttura:

- Nome del file
- Autore del file
- Data della creazione del file
- Versione corrente del file
- Tabella delle modifiche del codice

8.3.7 Versionamento dei file

La versione dei file viene identificata con la sigla:

vX.Y

dove le seguenti sigle indicano:

- X:** numero di versione primario che viene incrementato solo quando un file viene definito completo di tutte le sue funzionalità, definite nel documento *DefinizioneDiProdotto_v1.pdf*. Una volta che un file è completo vengono effettuati i test che lo riguardano. L'incremento di x impone l'azzeramento degli indici minori.
- Y:** numero di versione secondario che viene incrementato quando vengono apportate modifiche minori che non comportano un cambio dell'indice primario.

8.3.8 Tabella delle modifiche

A seguito di una modifica al codice del file deve essere aggiornata la tabella delle modifiche specificando in ordine:

- La versione del file alla modifica
- La data della modifica
- Autore della modifica
- Spiegazione delle modifiche effettuate, specificando eventuali inserimenti di metodi o variabili

9 Verifica e Validazione

Questa sezione illustra le procedure e gli strumenti da utilizzare nella verifica del codice. Per maggiori dettagli sulle metriche controllate, sugli strumenti e sui test programmati si veda il documento *PianoDiQualifica_v3.pdf*.

9.1 RoR

- **Analisi statica:** saranno utilizzate le gemme "metric_fu" e "metrinal", dedicate alle misurazioni delle metriche (<http://metric-fu.rubyforge.org/>).
- **Analisi dinamica:** per quanto riguarda i test di unità nell'ambiente RoR per l'implementazione delle classi di test sarà utilizzata l'apposita gemma Rspec (<http://rspec.info/>). I test di sistema saranno effettuati con l'utilizzo di Selenium, un tool di "automazione del browser" per eseguire test automatizzati sulle pagine web nella maggior parte dei browser (<http://seleniumhq.org/>). Per la validazione delle pagine prodotte saranno utilizzati gli strumenti di validazione del W3C (<http://validator.w3.org/>).

9.2 Android

- **Analisi statica** : sarà utilizzato il plugin Metrics per Eclipse, strumento utilizzato per la misurazione delle metriche del codice Android.
- **Analisi dinamica**: per i test di unità in ambiente Android si prevede l'utilizzo di classi derivanti da `AndroidTestCase` o `AndroidActivityTestCase`, classi astratte fornite dall'SDK di Android che si appoggiano sul framework⁹ JUnit per i test sul codice Android (http://developer.android.com/guide/topics/testing/testing_android.html).
I test di sistema saranno effettuati con l'utilizzo dei tool monkeyrunner e UI/Application Exerciser Monkey, strumenti per il test automatizzato di applicazioni Android (http://developer.android.com/guide/developing/tools/monkeyrunner_concepts.html).

9.3 Procedure

Le varie sessioni di test devono essere effettuate al termine di ogni iterazione del ciclo di vita del progetto.

Come proposto dal dott. Gregorio Piccoli ogni sessione di test sarà effettuata da più gruppi in parallelo e ognuno di essi deve scrivere un breve documento contenente:

- Codice progressivo della sessione di test
- Data e ora del test
- Partecipanti al test
- Lista degli errori trovati, accompagnati dalla descrizione e dall'ora di rilevamento

10 Test di unità e integrazione

Potranno essere effettuate le attività di verifica attraverso i test di unità e integrazione pianificati in fase di progettazione.

10.1 Stesura dei test

I test di unità e integrazione dovranno essere suddivisi per sottosistema software sul quale vengono applicati (i.g. Server, Desktop, Mobile) e per ognuno dei sottosistemi dovrà essere creata un'apposita tabella nella quale per ogni test dovrà essere indicato:

- Componente
- Classe di test
- Classe testata
- Esito del test

Potrà essere fornita una breve descrizione dei test eseguiti indicando strumenti utilizzati e librerie aggiuntive apposite per la creazione dei test.