wheelsoft

Norme di Progetto

10 febbraio 2007

Sommario

Norme interne per il progetto SIAGAS, necessarie per regolamentare lo svolgersi delle operazioni interne.

Informazioni documento

Produzione	WheelSoft - wheelsoft@gmail.com
Redazione	Giulio Favotto - gfavotto@studenti.math.unipd.it
Approvazione	Silvio Daminato - sdaminat@studenti.math.unipd.it
	Michele Volpato - mvolpato@studenti.math.unipd.it
File	Norme_di_Progetto_v1.9.pdf
Versione	1.9
Stato	Formale
Uso	Esterno
Distribuzione	Wheelsoft
	prof. Alessandro Sperduti
	prof. Renato Conte
	prof. Tullio Vardanega

Diario delle modifiche

1.9	10/02/07 - Aggiornamento *Norme di Codifica* con l'inserimento di norme specifiche per i 3 livelli dell'architettura del sistema SIAGAS, incluse norme per la *Compilazione*; inserimento delle sezioni *Norme per le attività di verifica* e *Tecniche e strumenti* in cui vengono rispettivamente descritte le norme per procedere alle attività di verifica, in particolare del codice, e gli strumenti da adottare (S. Ceschi Berrini e G. Favotto);
1.8	08/02/07 - Aggiornamento norme per *Versionamento* e *Programmazione*, inserimento norme per *Codifica* (M. Volpato);
1.7	03/02/07 - Aggiornamento norme per *Modalità di distribuzione* e correzioni varie (A. De Bortoli);
1.6	30/01/07 - Aggiornamento norme per *Modalità di distribuzione*, *Redazione Documenti* e *Versionamento* (M. Borgato);
1.5	25/01/07 - Inserimento norme per *Sviluppo e verifica Progettazione*, modificata modalità di distribuzione documenti (M. Borgato);
1.4	09/12/06 - Correzioni varie, sottolineatura parole da inserire nel glossario di progetto ed approvazione
1.3	08/12/06 - Aggiornamento a template di documento 1.2, inserimento sezioni *Nominazione dei documenti/Esterna, Redazione Documenti, Versionamento/In remoto* e *Sviluppo e verifica del glossario*, aggiornameto sezione 3.4 con modello di template 1.2;
1.2	29/11/06 - Correzioni varie, inserimento norme per *Sviluppo e verifica Analisi Requisiti*;
1.1	27/11/06 - Inserimento norme per *Comunicazioni*, *Convocazioni ed incontri*, *Gestione documenti ufficiali*, *Versionamento*;

Indice

1 Comunicazioni

1.1 Interne

1.1.1 Comunicazioni ufficiali

Fattibili da ogni membro del gruppo, devono avvenire tramite l'invio di una email al sistema di mailing list con indirizzo

wheelsoft@googlegroups.com

con oggetto Comunicazione Ufficiale e con la richiesta di ricevuta di ritorno.

1.1.2 Comunicazioni generiche

Possono avvenire tramite la mailing list senza alcuna specifica richiesta formale.

1.2 Esterne

Ogni comunicazione con soggetti esterni deve essere concordata con il Responsabile e documentata in maniera ufficiale.

2 Convocazioni ed incontri

Convocazioni di carattere interno, come riunioni o sessioni di sviluppo e di verifica collettive, o di carattere esterno, come incontri con committente e revisioni, vengono fissate dal Responsabile attraverso l'inserimento dell'appuntamento nel Calendario di Gruppo con due giorni d'anticipo e specificando data, orario, ordine del giorno. Nel caso il preavviso sia inferiore ai due giorni, la convocazione deve essere effettuata come comunicazione interna ufficiale.

Un membro del gruppo che ritiene necessario un incontro, interno o esterno, può manifestare l'esigenza contattando il Responsabile e suggerendo validi motivi per la convocazione. Spetta poi al responsabile valutare la richiesta ed agire di conseguenza.

Ad ogni convocazione concordata l'Amministratore di Progetto deve rendere disponibile copia cartacea della documentazione di interesse per tale convocazione. Deve inoltre redarre un documento che ne sintetizzi i temi trattati e le modalità dell'incontro secondo gli standard descritti in *Gestione documenti ufficiali*.

3 Gestione documenti ufficiali

3.1 Nominazione Documenti

3.2 Interna

La documentazione ufficiale interna deve essere nominata secondo i seguenti criteri:

{sigla}{contatore documento}_{data o versione}

dove:

- **sigla** indica il tipo di documento facendo riferimento alle attuali sigle esistenti:

  RN Riunione Interna;

  IN Incontro con committente;

  L'introduzione di una nuova sigla va concordata con l'Amministratore di progetto.

- **contatore documento** indica il contatore del rispettivo evento nella forma numerica xy, incrementato di uno rispetto all'ultimo documento ufficiale rilasciato;

- **data o versione** indicano la data dell'evento o per documenti di prolungata elaborazione la versione;

3.3 Esterna

La documentazione ufficiale esterna deve essere nominata secondo i seguenti criteri:

{titolo}_{versione}

dove:

- **titolo** indica il titolo assegnato internamente al documento;

- **versione** indica l'avanzamento di elaborazione raggiunto dal documento sotto forma numerica (vX.Y);

3.4 Template Documenti

Nel creare ogni tipo di documento [1] è necessario adattarsi al template di documento concordato e sviluppato in LaTeX. Pertanto ogni documento dev'essere redatto utilizzando LaTeX. Il template prevede un intestazione con, in testa, logo e nome del gruppo e, al piè di pagina, nome del documento e pagina. Il template è disponibile all'indirizzo:

[1] Eccezion fatta per accordi straordinari concordati con Responsabile ed Amministratore

http://www.wheelsoft.org/listing/files/TEMPLATE.zip

Adattandosi allo standard ciascun documento dovrà includere:

- **Titolo**
- **Data di creazione** della prima versione del documento;
- **Produzione** soggetto che ha prodotto il documento;
- **Redazione** soggetto/i che hanno redatto il documento;
- **Approvazione** soggetto/i che hanno approvato il documento;
- **Versione** attuale;
- **Stato** del documento:
  - Preliminare (solo a scopo informativo);
  - Formale (soggetto a revisione);
- **Uso**:
  - Interno;
  - Esterno;
- **File** nome del file del documento;
- **Lista di Distribuzione** a cui sarà fatto pervenire il documento;
- **Diario delle modifiche** in ordine di inverso versione, dalla più recente alla più antica, nel formato

  {Versione:}{Data}{Modifica effettuata}{Autore}[2]

  dove:
  - **Versione** dev'essere un numero nel formato X.Y con valore iniziale non inferiore a 1.1[3];
  - **Data** dev'essere la data in cui la modifica è stata effettuata;
  - **Modifica effettuata** deve specificare ciò che è variato dalla versione precedente;
- **Appunti sulla qualità**
  - del processo, nel caso il documento documenti un processo;
  - del prodotto inteso come il documento stesso;
- **Glossario** contenente la spiegazione dei termini specialistici usati nel documento stesso;

[2]Solo in caso l'autore sia diverso rispetto a quanto indicato nell'intestazione del documento;
[3]Standard di SVN;

3.5 Redazione documenti

Sono da considerare inoltre le seguenti norme di carattere tipografico:

- **Grassetto** da utlizzare con parsimonia per evidenziare parole chiave;
- *Corsivo* da utilizzare per citazioni o per termini significativi all'interno del documento;
- Sottolineato da utilizzare esclusivamene per i termini inseriti nel Glossario;

Per agevolare l'organicità, l'ordine e la comprensione del documento, ogni capitolo deve iniziare a pagina nuova.
Eventuali modifiche a queste norme vanno specificate all'intero del documento redatto.

3.6 Approvazione documenti

Una volta creato un documento, per passare dall'iniziale stato Preliminare a quello Formale, dev'essere approvato dal Responsabile, il quale può anche richiederne una revisione comportando quindi il rilascio di una nuova versione.

3.7 Distribuzione documenti

Una volta creato un documento che si trova in uno stato preliminare, può essere distribuito solo internamente[4] tramite il suo inserimento nella sezione **Documenti preliminari** del sito d'appoggio[5] in modo da poter essere visionato da tutti gli interessati. All'interno della suddetta sezione dev'essere presente, per ogni documento, una sottosezione denominata come il documento, che contiene al suo interno le varie versioni. Facilitando, in questo modo, la visione delle varie versioni e degli aggiornamenti apportati.

Un documento che invece è già passato al vaglio del Responsabile e si trova quindi in uno stato Formale può essere distribuito sia internamente, tramite il suo inserimento nella sezione **Documenti ufficiali** del sito d'appoggio, che esternamente, secondo i canali concordati con i soggetti esterni presenti nella lista di distribuzione. In questa sezione è presente l'ultima versione del documento visionato e verificato. Le precedenti versioni restano depositate nella sezione Documenti preliminari.

3.7.1 Modalità di distribuzione

- **Distribuzione interna:** l'inserimento di un documento nelle sezioni predisposte del sito d'appoggio deve avvenire in modo da permettere un'identificazione rapida e non ambigua dello stesso; si richiede quindi l'inserimento del nominativo assegnatogli, nonchè della sua versione nel formato:

{Nome documento} - {vX.Y}

[4] All'interno cioè del gruppo WheelSoft;
[5] http://www.wheelsoft.org;

Il documento deve essere inoltre disponibile in formato pdf ed in formato sorgente [6]. Nel caso un documento contenga al suo interno materiale multimediale (immagini o quant'altro), queste devono essere allegate al sorgente all'interno di un archivio in formato .zip. I documenti devono essere resi disponibili all'interno del sito nelle apposite directory di indirizzo base "http://www.wheelsoft.org/listing/" facendo riferimento alla gerarchia seguente.

- **archivio/**: per i documenti archiviati interni, che non sono stati consegnati per nessuna revisione
- **files/**: per i files generali
- **preliminari/**: per i documenti preliminari, modificati e in attesa di approvazione

Pertanto la gerarchia generale del sito per la distribuizione dei documenti (indirizzo base "http://www.wheelsoft.org/listing/") risulta essere la seguente:

- **RR/**: documenti consegnati per la Revisione dei Requisiti
  - **pdf/**: versioni PDF
  - **latex/**: sorgenti documento in LaTeX
- **RPP/**: documenti consegnati per la Revisione del Piano di Progetto
  - **pdf/**: versioni PDF
  - **latex/**: sorgenti documento in LaTeX
- **archivio/**: documenti archiviati interni, che non sono stati consegnati per nessuna revisione
- **files/**: files generali
  - **orari/**: orari settimanali dei singoli membri del gruppo
- **preliminari/**: documenti preliminari, modificati e in attesa di approvazione
  - **pdf/**: versioni PDF
  - **latex/**: sorgenti documento in LaTeX
- **zuml e rmp/**: per i files sviluppati con tools di supporto come *Poseidon* e *OpenWorkBench*.

- **Distribuzione esterna:** da definirsi;

[6] funzione allo stato dell'arte in via d'implementazione;

4 Versionamento

Ogni documento o file sviluppato e distributio deve avere una versione che ne identifichi lo stato d'avanzamento.

È stato reso disponibile un Server SVN raggiungibile all'indirizzo **svn://wheelsoft.no-ip.org:993/**. Il versionamento può essere effettuato in remoto per i documenti e materiale soggetto a modifiche da parte di tutto il gruppo.

4.1 Ambiente Microsoft Windows

Il Software consigliato per la gestione del Versionamento in ambiente *Microsoft Windows* è **Tortoise SVN**, reperibile all'indirizzo

http://tortoisesvn.net/downloads

Il Software gestisce il versionamento di ogni tipo di file, in codifica testuale o binaria.

Dal menù associato ad ogni file inserito nell'archivio sarà possibile accedere alle varie funzioni di revisione delle versioni, di cronologia, di annotazione etc...

4.1.1 Modalità in remoto

Ogni componente del gruppo possiede una username ed una password per connettersi al repository e quindi leggere, scrivere e modificare i file e i documenti. Una volta installato il programma, si suggerisce di procede come segue:

1. Impostazione dei dati per la connessione al server remoto, premendo con il pulsante destro del mouse in uno spazio vuoto;
2. selezionare TortoiseSVN, Repo-browser. Questo navigatore permette di visualizzare tutti i file presenti nel repository remoto;
3. inserire ora l'URL nell'apposito spazio: svn://wheelsoft.no-ip.org, quindi procedere;
4. inserire username e password personali e quindi nagivare nella cartella /SIAGAS/

Per quanto riguarda il versionamento in remoto:

1. creo una cartella in locale (working copy, WC) che farà da copia locale del materiale da versionare;
2. clicco con il tasto destro del mouse su uno spazio vuoto all'interno di tale cartella, TortoiseSVN, Import...
3. dopo aver modificato un file, premere con il pulsante destro del mouse sul file, TortoiseSVN, Commit... La finestra di dialogo di commit mostrerà tutti i cambiamenti apportati al file e il versionamento. Procedendo verrà quindi inviato il file al repository. Nel caso si prema con il pulsante destro del mouse sopra una cartella, tutto il suo interno verrà esaminato automaticamente.

Ogni qualvolta si desidera visionare gli eventuali aggiornamenti dei file apportati da altri, basterà premere con il pulsante destro del mouse in uno spazio vuoto all'interno della suddetta cartella e selezionare: TortoiseSVN, Update. La finestra di dialogo che verrà visualizzata adotta diversi colori per rappresentare i seguenti casi:

Colore	Azione
Verde	cambiamenti da WC a repository avvenuti con successo;
Porpora	aggiunto nuovo componente nel tuo WC;
Rosso scuro	componente ridondante eliminato dal tuo WC, oppure oggetto mancante aggiunto nel tuo WC;
Rosso chiaro	cambiamenti da WC a repository in conflitto;
Nero	nessun aggiornamento apportato;

4.2 Ambiente Linux

In ambiente *Linux* si consiglia l'uso del pacchetto **subversion**, reperibile all'indirizzo

http://subversion.tigris.org/project_packages.html

4.2.1 Modalità in remoto

L'utilizzo di *subversion* tramite console è molto semplice:
Innanzitutto è necessario scaricare in locale una copia del repository, con il comando che segue.

```
svn co svn://wheelsoft.no-ip.org:993/
```

Se si possiede già una copia locale del repository è sufficiente sincronizzare la copia del repository con quella locale, con il comando:

```
svn up
```

A questo punto è possibile modificare l'ultima versione del file desiderato, le modifiche vanno rispedite al server con il comando:

```
svn ci -m "commento di log"
```

5 Sviluppo e verifica Analisi Requisiti

5.1 Sviluppo Analisi dei Requisiti (AR)

L'AR si configura come il documento che mappa le esigenze fornite dal committente in requisiti descritti nella forma più specifica e non ambigua possibile. Pertanto, è richiesto che la formulazione di un requisito, sia esso

- di prodotto o funzionale;
- di processo o non-funzionale;
- implicito o proprio del dominio;
- esplicito;

venga fatta in maniera decidibile e non ambigua in modo da essere facilmente validata.

Quindi, oltre alle richieste formali di stesura di un documento ufficiale (sezione 3) è necessario:

Fonti: fornire precisa documentazione sulle fonti da cui i requisiti sono tratti e sulle modalità di acquisizione in modo da permetterne il tracciamento;

Classificazione: suddividere i requisiti in sottoinsiemi coesi ed identificare ogni requisito in maniera univoca con riferimenti numerici sequenziali;

Conflitti: identificare e risolvere eventuali contraddizioni o conflitti tra requisiti;

Priorità: assegnare ad ogni requisito o sottoinsieme un grado di priorità mediante interazione con gli stakeholder;

Verifica: verificare completezza e consistenza finale dei requisiti;

Descrizione grafica: fornire rappresentazione semi-formale o grafica dei casi d'uso rappresentati dai requisiti tramite gli opportuni strumenti di modellazione, sempre curandosi di rendere tali rappresentazione tracciabili;

Descrizione narrativa: fornire una descrizione narrativa che evidenzi:

- Attori Coinvolti;
- Scopo e descrizione del requisito;
- Flusso base degli eventi;
- Flussi alternativi;
- Pre e Post condizioni;

La struttura del documento dovrebbe quindi essere riconducibile a questa forma

1. **Introduzione**
   (a) Scopo del documento
   (b) Scopo del prodotto

(c) Glossario
- Definizioni
- Acronimi
- Abbreviazioni

(d) Riferimenti
- Normativi
- Informativi

2. **Descrizione generale**

(a) Contesto d'uso del prodotto
- Processi produttivi e modalità d'uso
- Piattaforma d'esecuzione e interfacciamento

(b) Funzioni del prodotto

(c) Caratteristiche degli utenti

(d) Vincoli generali

(e) Assunzioni e dipendeze

3. **Tabella dei requisiti**

(a) Requisiti funzionali

(b) Requisiti di qualità

(c) Requisiti di interfacciamento
- Con l'ambiente di installazione ed uso
- Con i processi produttivi e le modalità d'uso presso l'utente
- Con l'operatore

4. **Use Case**

Il documento finale dovrebbe quindi risultare:

- Completo
- Ben organizzato
- Privo di inconsistenze
- Privo di ambiguità
- Privo di ridondanze
- Privo di imprecisioni terminologiche
- Privo di dettagli tecnici

5.2 Verifica AR

La verifica deve accertare che ci siano i presupposti affinchè il prodotto venga costruito nel modo giusto (*Did i build the system right?*) valutando inoltre l'aderenza del documento alle norme sopra specificate nonchè la chiarezza espressiva, la chiarezza strutturale, l'atomicità, l'aggregazione, la necessarietà e la sufficienza dei requisiti.

Al fine di semplificare la validazione finale è inoltre necessario verificare che il tracciamento fonte-requisito possa essere fatto in maniera univoca.

6 Sviluppo e verifica Progettazione

6.1 Revisione di Progetto Preliminare (RPP)

6.1.1 Sviluppo Specifica Tecnica (ST)

La ST è la descrizione di una soluzione che soddisfi tutti i portatori di interesse del progetto. I prodotti di questa fase sono l'architettura e il modello logico. Definire l'architettura del prodotto impiegando componenti con specifica chiara e coesa, realizzabili con risorse date e costi fissati, utilizzando una struttura che faciliti la manutenibilità, in maniera decidibile e non ambigua in modo da essere facilmente validata. Viene richiesto di adottare un approccio sintetico, conciso ed essenziale, con valutazione delle possibili alternative.

Sulla base dell'analisi dei requisiti (AR), la progettazione definisce come tali requisiti saranno soddisfatti, entrando nel merito della struttura che dovrà essere data al sistema software da realizzare. Si sottolinea che deve comunque rimanere una fase distinta dalla programmazione e codifica. A questo livello di progettazione (altissimo livello) si definisce solo la struttura complessiva del sistema in termini di principali moduli di cui esso è composto. A questo livello si fa uso di particolari architetture hardware, sistemi operativi, protocolli di rete, framework e pattern. L'architettura non è un fine, ma uno strumento importante per il raggiungimento degli obiettivi di progetto. Obiettivo importante della progettazione è di soddisfare i requisiti di qualità fissati dal committente e dal fornitore.

Come accordato con il prof. R. Conte, dovranno essere utilizzati almeno tre pattern visti e trattati a lezione. Classificabili in tre principali categorie:

- Creazionale: tratta la configurazione ed inizializzazione di classi ed oggetti.
- Strutturale: tratta il disaccoppiamento tra interfacce ed implementazione delle classi e degli oggetti della loro composizione.
- Comportamentale: tratta le interazioni dinamiche tra gruppi di classi ed oggetti.

La struttura del documento ST dovrebbe essere riconducibile a questa forma:

1. **Introduzione**
   (a) Scopo del documento
   (b) Scopo del prodotto
   (c) Glossario
      - Definizioni
      - Acronimi
      - Abbreviazioni
   (d) Riferimenti
      - Normativi
      - Informativi
2. **Definizione di prodotto**
   (a) Metodo e formalismo di specifica

   (b) Presentazione dell'architettura generale del sistema e identificazione dei componenti architetturali di alto livello

3. **Descrizione dei singoli componenti**
   (a) Tipo, obiettivo e funzione del componente
   (b) Relazioni d'uso di altre componenti
   (c) Interfacce con e relazioni di uso da altre componenti
   (d) Attività svolte e dati trattati
4. **Stime di fattibilità e di bisogno di risorse**
5. **Tracciamento della relazione componenti - requisiti**

Il documento finale dovrebbe quindi risultare:

- Completo
- Ben organizzato
- In accordo con l'Analisi dei Requisiti
- Privo di inconsistenze
- Privo di ambiguità
- Privo di ridondanze
- Privo di imprecisioni terminologiche

6.1.2 Verifica ST

La verifica, come per l'Analisi dei Requisiti, deve accertare che ci siano i presupposti affinchè il prodotto venga costruito nel modo giusto, valutando l'aderenza del documento alle norme sopra specificate. Intende appurare che non vengano introdotti errori e che la chiarezza espressiva e strutturale, l'atomicità e l'aggregazione siano rispettate.

6.2 Revisione di Progetto Definitivo (RPD)

6.2.1 Sviluppo Definizione del Prodotto (DP)

La progettazione di dettaglio rappresenta una descrizione del sistema molto vicina alla codifica, ovvero che la vincola in maniera sostanziale (per esempio, descrivendo non solo le classi in astratto ma anche i loro attributi e metodi, con relativi tipi). A causa della natura impalpabile del software, e a seconda degli strumenti che si utilizzano nel processo, il confine fra progettazione e codifica può essere anche praticamente impossibile da identificare.

Lo scopo del documento DP è definire nel dettaglio l'architettura del sistema già descritta nel documento Specifica Tecnica. Si devono raffinare a livello di progettazione di dettaglio i seguenti punti:

- tipo, obiettivo e funzione;
- relazioni d'uso di altre componenti;
- interfacce e relazioni d'uso da altre componenti;
- attività svolte e dati trattati

già analizzati nel documento Specifica Tecnica. Perciò scopo di questo documento è anche guidare i programmatori durante la codifica, poichè non si devono occupare di trovare soluzioni e implementazioni improvvisate. Infatti per ogni classe sono già state definite le liste dei metodi e dei campi dati, alle quali i programmatori si devono attenere.

La struttura del documento DP dovrebbe essere riconducibile a questa forma:
Parte 1 - Descrizione generale

1. **Introduzione**
   (a) Scopo del documento
   (b) Scopo del prodotto
   (c) Glossario
      - Definizioni
      - Acronimi
      - Abbreviazioni
   (d) Riferimenti
      - Normativi
      - Informativi
2. **Standard di progetto**
   (a) Standard di progettazione architetturale
   (b) Standard di documentazione del codice
   (c) Standard di denominazione di entità e relazioni
   (d) Standard di programmazione
   (e) Strumenti di lavoro

Parte 2 - Specifica delle componenti

1. Raffinamento a livello di progettazione di dettaglio della specifica fornita in sezione 3 della **Specifica Tecnica**

Appendice

1. **Codice sorgente**
2. **Tracciamento della relazione componenti - requisiti**

Il documento finale dovrebbe quindi risultare:

- Completo

- Ben organizzato
- In accordo con l'Analisi dei Requisiti
- In accordo con la Specifica Tecnica
- Privo di inconsistenze
- Privo di ambiguità
- Privo di ridondanze
- Privo di imprecisioni terminologiche

6.2.2 Verifica DP

La verifica, deve accertare che ci siano i presupposti affinchè il prodotto venga costruito nel modo giusto, valutando l'aderenza del documento alle norme sopra specificate. Intende appurare che non vengano introdotti errori e che la chiarezza espressiva e strutturale, l'atomicità e l'aggregazione siano rispettate

7 Sviluppo e verifica Glossario

Il Glossario si configura come il documento che raccoglie la spiegazione di tutti i termini specialistici usati nei documenti prodotti al fine di eliminare ogni ambiguità relativa al linguaggio.

Deve pertanto essere

Completo contenendo cioè tutti i termini che sono stati evidenziati con sottolineatura nei vari documenti;

Non ambiguo la spiegazione dei vari termini dev'essere precisa e chiara;

Non ridondante non contenendo cioè ripetizioni o spiegazioni accoppiate;

Essenziale contenendo cioè non più di quanto strettamente e realmente necessario;

Ordinato i termini devono essere disposti in ordine alfabetico in modo da permettere una ricerca rapida del termine cercato;

Dal punto di vista formale il glossario deve adeguarsi alle norme esistenti per la documentazione. Sarà cura del redattore definire la forma di presentazione di termine-spiegazione.

La verifica dovrà assicurarsi che il documento rispetti le caratteristiche sopra elencate e che si presenti in una forma leggibile e diretta.

8 Norme di sviluppo e codifica

8.1 Livello presentation

Il prodotto, a livello di presentation layer, dovrà essere caratterizzato dall'utilizzo del linguaggio HTML in pagine XHTML v1.1 gestite con CSS v2.0. Ognuno di questi utilizzi dovrà aderire completamente agli standard W3C esistenti e quindi dovrà superare positivamente la verifica effettuata con gli appositi tools on-line messi a disposizione dal W3C.
Specifiche XHTML (http://www.w3.org/TR/xhtml1/), rispettivo validatore (http://validator.w3.org/).
Specifiche CSS (http://www.w3.org/TR/2006/WD-CSS21-20060411/), rispettivo validatore (http://jigsaw.w3.org/css-validator/).

Sarà inoltre necessario sviluppare il portale secondo le linee guida esistenti in temini di accessibilità WAI-AAA (http://www.w3.org/WAI/WCAG1AAA-Conformance/).

Bisognerà infine verificare che:

- il portale risulti visualizzabile con tempi di caricamento accettabili per ogni connessione remota;
- il portale risulti completamente visualizzabile a partire da una risoluzione monitor di 800x600px, requisito che deve essere esplicitato a fondo di ogni pagina;

8.2 Livello logic

Il prodotto, a livello di logic layer, dovrà essere caratterizzato dall'utilizzo del linguaggio PHP 5.2.0 secondo le specifiche indicate all'indirizzo http://it2.php.net/. Di importanza primaria è che in fase di realizzazione di un componente siano tenute in considerazione le modalità di verifica previste nel Piano di qualifica. Dovranno inoltre essere seguite le indicazioni riportate in questa sezione.

8.2.1 Costrutti di controllo

I costrutti di controllo dovrebbero avere uno spazio tra la keyword e la parentesi aperta per distinguerli dalle chiamate di funzione, è fortemente consigliato l'uso delle parentesi graffe anche quando sono opzionali, per rendere il codice più leggibile e manutenibile. Esempio di riferimento:

```
(condizione2)) {
    azione1;
} elseif ((condizione3) && (condizione4)) {
    azione2;
} else {
    azionedidefault;
}
?>
```

8.2.2 Chiamate di funzione

Le funzioni e i metodi dovrebbero essere sempre chiamati senza spazi fra il nome della funzione e la parentesi aperta e fra la parentesi e il primo argomento:

```
<?php
$var = funzione($uno, $due, $tre);
?>
```

In caso di più assegnamenti indentare come segue:

```
<?php
$corta            = funzione($uno);
$variabile_lunga = funzione($due);
?>
```

I parametri passati ad una funzione non devono essere più di 6.

8.2.3 Commenti

I commenti sono molto importanti per la manutenzione del codice in quanto assicurano leggibilità e facilitano la compresione, è possibile utilizzare lo stile C (/* */) o lo standard C++ (//).

- **Commenti d'intestazione**

  Per la manutenibilità del codice sono necessarie delle linee di commento sia all'inizio del file, per descrivere il contenuto del file stesso, sia prima della definizione di ogni classe. Per la realizzazione di questi commenti si faccia riferimento al seguente esempio che comprende entrambi i casi:

```
<?php

/**
* Breve descrizione del file
*
* Verbosa descrizione del file (se necessaria)...
*
* PHP 5
*
*
* @category    SIAGAS
* @package     NomePackage
* @author      Autore Originale <autore@example.com>
* @author      Altro Autore <altro@example.com>
* @copyright   2006 - 2007 Wheelsoft
* @license     http://www.gnu.org/copyleft/lesser.html
*              LGPL License 2.1
* @version     X.Y
* @link        http://www.wheelsoft.org
* @last_mod    L'ultimo ad aver modificato il file
* @mod         Note importanti relative alle modifiche
*/
```

```
/**
* Breve descrizione della classe
*
* Verbosa descrizione della classe (se necessaria)...
*
* @category    SIAGAS
* @package     NomePackage
* @author      Autore Originale <autore@example.com>
* @author      Altro Autore <altro@example.com>
* @copyright   2006 - 2007 Wheelsoft
* @license     http://www.gnu.org/copyleft/lesser.html
*              LGPL License 2.1
* @version     X.Y
* @link        http://www.wheelsoft.org
*/
class Esempio
{
}
?>
```

- **Commenti dei campi dati**

  Ogni campo dati va commentato come segue:

```
/**
 * Descrizione del campo dati
 *
 * Note, ad esempio i possibili valori
 *
 * @var string
 * @access [private|public] in questo caso private
 */
 var $_esempio = "buono";
```

- **Commenti dei metodi**

  Ogni metodo andrebbe commentato usando lo stile che viene processato dal *phpDocumentator*, le descrizioni vanno fatte in terza persona, devono essere brevi e preferibilmente cominciare con un verbo. La descrizione serve a documentare quel che sta "dietro" il nome del metodo, quindi se il metodo è semplice si può omettere la descrizione.

  Una buona descrizione è: *Carica i dati dello studente.*

  Descrizioni meno buone: *Carico i dati*, oppure *Questo metodo carica i dati dello studente.*

  Seguire il seguente esempio:

```
/**
 * Descrizione del metodo.
```

```
 *
 * @param string $arg1  la stringa da stampare
 * @param int    $arg2  l'intero da stampare
 *
 * @return int  l'intero stampato
 * @throws classeeccezione  [description]
 *
 * @access public
 * @static
 */
function stampaDue($arg1, $arg2 = 0)
{
 /*
  * Un blocco di commento.
  * Non avento doppio asterisco all'inizio
  * non è processato dal phpDocumentator
  */
  stampaString($arg1);
  stampaInt($arg2);
  return $arg2;
}
```

8.2.4 URL d'esempio

Per la **RFC 2606** quando si necessita di URL d'esempio o indirizzi e-mail d'esempio si devono utilizzare i seguenti esempi:
example.com, *example.org* o *example.net*.

8.2.5 Nomi di classi, metodi e costanti

- **Classi**

  I nomi delle classi dovrebbero avere nomi descrittivi, evitando gli acronimi dove sia possibile. Dovrebbero cominciare sempre con una lettera maiuscola.
  Esempi di buoni nomi per le classi: Log, Studente, Proponente.

- **Metodi**

  L'iniziale del nome del metodo è minuscola mentre ogni lettera che comincia una nuova parola maiuscola. Esempi: getNome(), esci(), getNuoviStudentiIscritti().

  I membri privati (metodi o campi dati) sono preceduti da un singolo *underscore*.
  Esempi: _distruggi(), _cancellaStudente().

- **Costanti**

  Le costanti dovrebbero sempre avere tutte le lettere maiuscole.

8.2.6 Compilazione

Per snellire le attività di verifica è necessario controllare durante la fase stessa di realizzazione che il codice

- risulti sintatticamente corretto;
- risulti ben strutturato;
- non includa codice logicamente non raggiungibile;
- non generi eccezioni non controllate;

PHP5 fornisce supporto a questa esigenze attraverso la funzione di segnalazione degli errori Error handling and logging che genera notifiche sia per errori sintattici, che per una serie di errori logici, come variabili non inizializzate e rami non raggiungibili.

8.3 Livello data

Il prodotto, a livello di data layer, dovrà essere caratterizzato dall'utilizzo del linguaggio SQL:2003 secondo gli standard ISO ed ANSI disponibili gratuitamente; le specifiche del SQL:2006 essendo a pagamento non sono state consultate.
SQL dovrà essere utilizzato nella scrittura delle create table del DB del sistema e nelle query di interrogazione del DB dal livello logico.

9 Norme per le attività di verifica

9.1 Documentazione

Oltre alle norme specifiche indicate per i vari documenti nelle rispettive sezioni sopra citate, la verifica deve portare a resoconti scritti, quindi documentati delle anomalie ed inconsistenze riscontrate. La correzione, per semplicità di gestione, dev'essere affidata all'autore ultimo del documento in questione.

Attenzione dovrà essere posta nella verifica dei vari tracciamenti presenti.

9.2 Codice

La verifica del codice in maniera parallela alla realizzazione è un'attività di vitale importanza.

In particolare dovranno essere effettuate la verifica della aderenza del codice alle norme di codific e l'analisi statica al momento della compilazione.

Nel momento in cui la realizzazione di un meotodo-componente raggiungerà uno stato prototipale si procederà in quest'ordine:

1. Rilevazione delle metriche previste relative al metodo-componente in modo da selezionare il tipo e la quantità di test necessari;
2. Verifica dell'esistenza di test adatti al meotodo-componente in analisi. Realizzazione dei test funzionali sottoforma di script ad hoc o forniti dai package di supporto previsti (JWebUnit e SimpleTest);
3. Registrazione degli esiti;

4. In caso di anomalie o malfunzionamenti gli esiti in questione andranno esportati ed inseriti nell'apposita segnalazione dei bug nel sistema di bugtracking fornito da sourceforge.net realtivo al progetto siagas. Tutti i campi necessari della segnalazione andranno inseriti. L'assenza di segnalazioni assicurerà il buon esito dei test, i cui esiti però andranno comunque documentati secondo il template che sarà definito a breve.

Maggiori dettagli sulle modalità di utilizzo di JWebUnit e di SimpleTest saranno presenti nella nuova revisione delle Norme di Progetto.

9.3 Prodotto

Il prodotto finale dovrà essere testato prima attraverso i test d'integrazione previsti nel Piano di Qualifica e poi con i test di collaudo. Norme specifiche per questi test saranno presenti nella nuova revisione delle Norme di Progetto.

10 Tecniche e strumenti

In questa sezione vengono elencati e descritti gli strumenti software e hardware il cui utilizzo sarà necessario per la realizzazione del prodotto SIAGAS.

Si tratta di strumenti software di varia natura, ad esempio applicazioni, linguaggi, librerie, ect... e di strumenti hardware, ovvero le macchine sulle quali si svilupperà il prodotto o che comunque avranno un ruolo importante per tale scopo.

Per il loro utilizzo si faccia riferimento alla documentazione fornita nei pacchetti.

10.1 Linguaggi

10.1.1 PHP

1. **Nome**: PHP
2. **Versione**: 5.2.0
3. **Descrizione**: linguaggio utilizzato per la realizzazione di applicazioni Web
4. **Link**: *http://php.net*

10.1.2 MySQL

1. **Nome**: MySQL
2. **Versione**: 5.0.33
3. **Descrizione**: motore di database libero
4. **Link**: *http://www.mysql.com*

10.2 Package esterni

10.2.1 pear.MDB2

1. **Nome**: pear.MDB2
2. **Versione**: 2.3.0
3. **Descrizione**: package per accesso a database
4. **Link**: *http://pear.php.net*

10.2.2 FPDF

1. **Nome**: FreePDF
2. **Versione**: 1.52
3. **Descrizione**: classe PHP che permette di generare files PDF direttamente da PHP
4. **Link**: *http://www.fpdf.org/*

10.2.3 JWebUnit

1. **Nome**: JWebUnit
2. **Versione**: 1.4
3. **Descrizione**: JWebUnit è un framework Java che facilita la creazione e l'esecuzione di tests per applicazioni web. Usa il motore di JUnit. JWebUnit fornisce un set di high-level API per verificare la navigazione in un'applicazione web unite ad una serie di asserzioni per verificarne la correttezza. Include test per form entry e submission, validazione dei contenuti di tabelle e per altre caratteristiche tipiche di web application. Permette inoltre l'esportazione del codice di scripting creato e degli esiti.
4. **Link**: *http://jwebunit.sourceforge.net/*

10.2.4 SimpleTest

1. **Nome**: Simple Test
2. **Versione**: 1.0.0.1
3. **Descrizione**: Framework di web testing apposito per la creazione di una test suite per PHP. Si integra come modulo all'interno di JWebUnit.
4. **Link**: *http://www.lastcraft.com/simple_test.php, http://sourceforge.net/projects/simpletest/*

10.3 Applicazioni

10.3.1 Subversion

1. **Nome**: svn
2. **Versione**: 1.3.2
3. **Descrizione**: sistema di versionamento
4. **Link**: *http://subversion.tigris.org/*

10.3.2 gedit

1. **Nome**: gedit
2. **Versione**: 2.14.2
3. **Descrizione**: editor di testi
4. **Link**: *http://gnome.org/projects/gedit/*

10.3.3 Apache HTTPD

1. **Nome**: Apache HTTPD
2. **Versione**: 2.2.4
3. **Descrizione**: Web server
4. **Link**: http://httpd.apache.org/

10.3.4 LaTeX

1. **Nome**: latex
2. **Versione**: 2.5
3. **Descrizione**: sistema di composizione di testo
4. **Link**:

10.3.5 Openoffice

1. **Nome**: Openoffice
2. **Versione**: 2.0
3. **Descrizione**: suite da ufficio libera
4. **Link**: *http://www.openoffice.org*

10.3.6 MSN messenger

1. **Nome**: MSN messenger
2. **Versione**: 7.5
3. **Descrizione**: Chat client multiprotocollo
4. **Link**: *http://it.msn.com*

10.3.7 OpenWorkBench

1. **Nome**: Open Workbench
2. **Versione**: 1.1.4
3. **Descrizione**: Open Source project scheduling
4. **Link**: *http://www.openworkbench.org/*

10.4 Hardware

10.4.1 wheelsoft serverSVN

1. Hardware:
   - Architettura: x86
   - Processore: AMD(R) Athlon(R) K7 900 MHz
2. Sistema operativo:
   - Sistema operativo: Windows Xp sp2
3. Servizi':
   - SVN-server: Subversion 1.4.3

10.4.2 wheelsoft serverHTTP

Wheelsoft serverhttp è il server di Wheelsoft, sul quale risiedono le nostre risorse. Il computer appartiene ad Alessio Rambaldi.

1. Hardware:
   - Architettura: x86
   - Processore: Intel(R) Pentium(R) 3 CPU 450 MHz
2. Sistema operativo:
   - Sistema operativo: debian GNU/Linux *sarge ETCH*
   - Kernel: 2.4.8-2-3861
3. Servizi':
   - Web-server: Apache 2.2.4
   - SSH: openssh 4.5
   - Database: MySQL 5.0.33