Piano dei tests

Release 0.5

December 5, 2009
Firenze

Approvazione, redazione, lista distribuzione

approvato da	il giorno	firma
Marco Tinacci	December 5, 2009	

redatto da	il giorno	firma
Manuele Paulantonio	December 5, 2009	
Daniele Poggi	December 5, 2009	
Massimo Nocentini	December 5, 2009	

distribuito a	il giorno	firma
Francesco Calabri	December 5, 2009	
Niccoló Rogai	December 5, 2009	
Marco Tinacci	December 5, 2009	

Contents

Part I

Planning

Chapter 1

Master Plan

1.1 Scope

Questo è il *Master Plan* per il progetto **PMango**. Questo piano considera solo gli elementi software relativi alle aggiunte/modifiche descritte nel documento di analisi.

L'obiettivo primario di questo piano di test è quello di assicurare che la nuova versione di **PMango** offrirà lo stesso livello di informazioni e dettagli reso disponibile della versione corrente e aggiungerà tutte quelle informazioni necessarie per raggiungere gli obiettivi modellati dal processo di analisi.

Il progetto avrà tre livelli di testing:

- unit
- system/integration
- acceptance

I dettagli di ogni livello verranno definiti nella sezione 1.6.1 e negli specifici *level plan*.

Il quanto temporale stimato per questo progetto è molto compatto, quindi *ogni* ritardo nella fase di progettazione, sviluppo, installazione e verifica possono avere effetti significati sul deploy finale.

1.2 Elementi software sottoposti a test

- some components produced by the detailed desing team

1.3 Funzionalità che verranno testate

- processo per la generazione di *Gantt chart*
- processo per la generazione di *WBS chart*
- processo per la generazione di *Task Network chart*
- interfaccia grafica per la selezione delle nuove *UserOption* per ogni *Chart*
- aggiunta di ogni *Chart* alla sezione di reportistica

1.4 Software Risk Issues

Ci sono alcuni punti che ci portano a definire questa sezione:

- Reverse engineering di codice sorgente esistente, documentato nei sorgenti ma non ha documenti ufficiali (apparte le tesi) che descrivino in modo chiaro la struttura statica e dinamica di tutto il lavoro esistente.
- Uso di librerie esterne per la generazione delle immagini e dei documenti pdf.

1.5 Strategia

1.5.1 Testing machines

Identifichiamo due classi di macchine sulle quali viene condotto il processo di testing:

- *develop machine* macchine dove avviene sia lo sviluppo del codice che parte del testing.
- *acceptance machine* macchine (molto probabilmente unica, a meno di guasti) dove avviene solo il processo di testing.

1.5.2 Strumenti

Durante il processo di testing usufruiamo dei seguenti strumenti:

- controllo visivo umano per quanto riguarda il confronto dell'output con l'output atteso (descritto nei documenti *Test case specification*) per quanto riguarda immagini
- utilizzo di verifica automatica di batterie di test usando l'insieme di oggetti contenuti nella distribuzione di *phpUnit* oppure attraverso classi di test non appartenenti al precedente framework ma che hanno la stessa idea di verifica (controllo automatico tra output e risultato previsto).

1.5.3 Test failure's metrics

Modelliamo il concetto di *failure* aggiungendo delle informazioni, in modo da specializzarlo per il nostro processo di testing. Ogni specializzazione ha il significato di esprimere la gravità (importanza) del fallimento. Quindi, l'esito di un test può essere *success* o *failure*, nel secondo caso aggiungiamo queste specializzazioni:

minor il fallimento del test non è da considerarsi un evento grave.
Gli oggetti software che hanno prodotto questa failure possono essere comunque inseriti nella release di **PMango 3**, non impediscono l'avanzare dello sviluppo. Possiamo identificare di questa failure come un *defect*.

critical il fallimento del test è da considerarsi un evento grave.
Gli oggetti software che hanno prodotto questa failure non possono essere inseriti nella release di **PMango 3**, necessitano di ricontrollare il codice relativo a tali oggetti; non impediscono l'avanzare dello sviluppo.

blocking il fallimento del test è da considerarsi un evento grave.
Gli oggetti software che hanno prodotto questa failure non possono essere inseriti nella release di **PMango 3**, necessitano di ricontrollare il codice relativo a tali oggetti e, se necessario, ricontrollare il relativo documento di progettazione; impediscono l'avanzare dello sviluppo.

Queste misure sono valide per tutte le failure di test appartenenti a ogni level plan descritto in 1.6.

1.6 Level plans

1.6.1 Definitions

Il processo di testing per il progetto **PMango** consiste nei livelli seguenti.

unit questo livello viene effettuato da tutti gli sviluppatori e stilato dal team dei verificatori con un rapprensentante degli sviluppatori.

Ogni motivazione riguardo ogni singolo unit test deve essere resa disponibile e documentata in modo chiaro o in un documento apposito[1] oppure nel codice nel caso che

- viene utilizzato un strumento automatico indicato nelle sezione 1.5.2
- la motivazione ha una dimensione ragionevolmente corta che è possibile inserirla come commento nel codice

[1] inserire un riferimento al relativo documento di test case specification

Questo livello viene esercitato su macchine di tipo *develop machine*.

system/integration questo livello viene eseguito dal team dei verificatori in presenza di un rappresentante degli sviluppatori se necessario.

Ogni motivazione e descrizione di questi test deve essere esposta nei documenti *Test case specification*

Questo livello viene esercitato su macchine di tipo *acceptance machine* e *develop machine*.

acceptance questo livello viene eseguito dal cliente in presenza di un rapprensentante dei verificatori.

Una volta terminato il livello di *acceptance* il prodotto viene rilasciato al cliente il quale può continuare la fase di testing in parallelo alla fase di utilizzo.

Questo livello viene esercitato su macchine di tipo *develop machine*.

1.6.2 Precedence Relation

Possiamo costruire la relazione di precedenza $\sqsubset$ fra coppie di level plan che permette ad un oggetto software di avanzare nel processo di testing, in modo da garantire il suo corretto funzionamento durante la fase di revisione congiunta oppure nel collaudo.

Definiamo $\sqsubset$ in questo modo:

unit $\sqsubset$ system/integration ogni oggetto software entra nel processo di testing dal level plan *unit*.

Quando soddisfa tutti i suoi *unit* test oppure, per ogni fallimento, la metrica misura *minor*, allora può essere disponibile per il level plan *system/integration* se è richiesto da qualche *system/integration* test.

system/integration $\sqsubset$ acceptance quando un oggetto (o gruppo di oggetti) superano tutti i test oppure, per ogni fallimento, la metrica misura *minor*, allora l'oggetto (o gruppo di oggetti) può avanzare nel level plan *acceptance*

La relazione $\sqsubset$ è riflessiva, in quanto un test permane in un level plan finche non soddisfa i requisiti per passare nel successivo; non è ne simmetrica ne transitiva, in quanto vogliamo garantire al committente la sequenzialità del processo di testing.

Osservazione: quando un test passa da un livello al successivo deve essere continuo rispetto alla batteria dei test specificata nel livello che lascia. Ovvero: se avanza di livello deve continuare a soddisfare le condizioni che gli hanno permesso di avanzare fino al livello corrente.

1.7 Pass/fail criteria

Il risultato dell'intero piano delle prove è dato dalla seguente relazione:

risultato	**criteri**
success	numero di *minor* failures ≤ 10
minor failure	numero di *minor* failures > 10
critical failure	almeno una *critical* failure
blocking failure	almeno una *blocking* failure

Table 1.1: La colonna *criteri* si riferisce all'insieme dei *design specification* definite in II

Il processo di testing verrà completato nella data in cui avverrà il collaudo con il committente oppure quando questo piano da esito *success* in base alla relazione definita dalla tabella, prima del collaudo. Dalla fine del processo di testing, la nuova versione di PMango viene considerata *live*.

Nel caso in cui il nostro team non riuscisse a portare a termini gli impegni presi entro la data del collaudo, il processo di testing proseguirà fino alla data in cui si considera terminato il tempo a disposizione per eseguire l'esame.

1.8 Enviromental needs

I seguenti elementi sono richiesti per supportare l'intero processo di testing:

- Sia *develop machine* che *acceptance machine* devono avere installato una istanza di un server (L/W/M)AMP, con tutti i necessari permessi per la corretto funzionamento, relativamente al sistema operativo presente
- Sia *develop machine* che *acceptance machine* devono offrire tutte quelle *third party resources* necessarie per l'utilizzo della nuova versione di PMango (fonts microsoft, ...) compresi tutte quelle necessarie per la versione di PMango attuale.

Part II

Desing Specification

Chapter 2

Generate Chart Design

2.1 Features to be tested

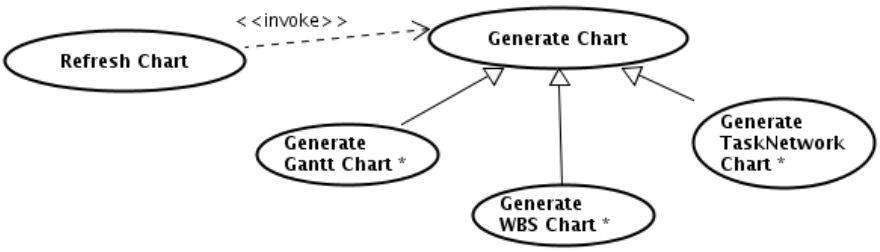

Figure 2.1: spaccato tratto dal package *Commons* del documento di analisi, parte *Use cases*

Verranno testati tutti gli use case espressi dalla figura tranne "Refresh Chart" in quanto è stato riportato solo per favorire la comprensione del contesto e perche la sua attività di costruire la *UserOptionChoice* e recuperare l'insieme dei *Task* vengono testate nel documento [1]

2.2 Raffinamento della strategia

costruzione strutture di input Gli usi case descritti nella sezione 2.1, verranno testati usando come input strutture valide e strutture non valide. Le strutture di input (sia valide che non) verranno definite in modo che lo use case venga esercitato per la generazione dei dettagli descritti nel documento di analisi e cercare di provare la generazione di gran parte delle notazioni grafiche (relative al tipo *Chart* che si sta considerando).

[1] make a test document for these use cases.

confronto dell'output Per ogni use case di "generazione" verranno prodotti dei modelli, in notazione *mockup* disegnati a mano, per poter effettuare il confronto con le immagini di output generate dall'implementazione.

modalità di esecuzione Ogni use case verrà esercitato in modo *individuale*, non verranno esercitate combinazioni con altri use case, come la figura esprime.

2.3 Tests identification

Sotto vi sono i reference a tutte le *case specification* relative a questo *desing*, definite in III:

- Generate Chart - Invalid input
-
-

2.4 Pass/fail criteria

Il risultato di questo desing è espresso dalla seguente relazione rappresentata da questa tabella:

risultato	**criteri**
success	numero di *minor* failures ≤ 5
minor failure	numero di *minor* failures > 5
critical failure	almeno una *critical* failure
blocking failure	almeno una *blocking* failure

Table 2.1: La colonna *criteri* si riferisce all'insieme di tests identificati nella sezione 2.3

Part III

Case Specification

Chapter 3

Generate Chart - Invalid input

3.1 Input

3.2 Environment

3.3 Output

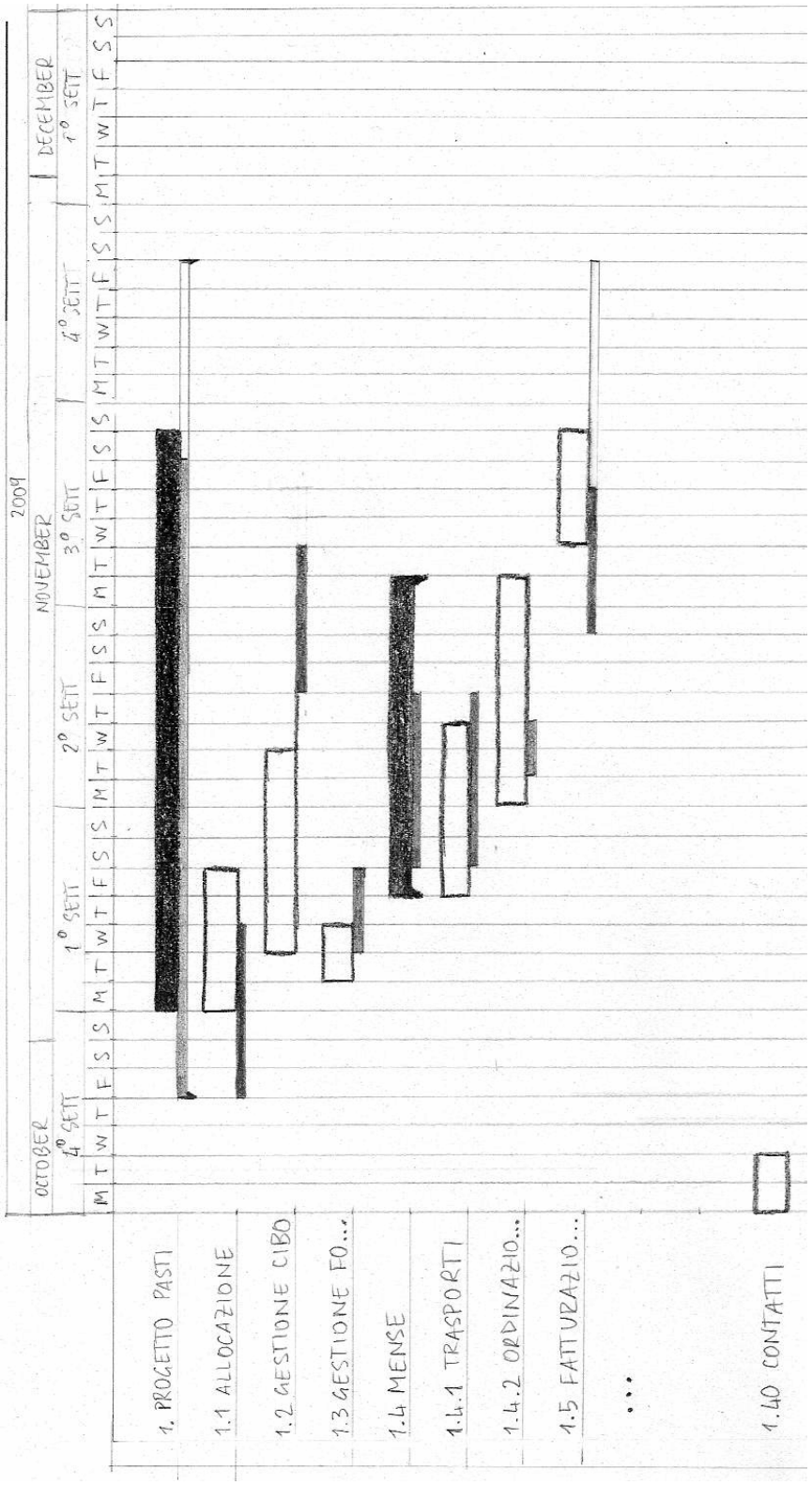

Figure 3.1: output for the invalid input structure