**Esempio di documentazione**

* **Indice**

1 Indice 2

2 Introduzione 3

2.1 Informazioni sul progetto 3

2.2 Abstract 3

2.3 Scopo 3

3 Analisi 4

3.1 Analisi del dominio 4

3.2 Analisi e specifica dei requisiti 4

3.3 Use case 7

3.4 Pianificazione 7

3.5 Analisi dei mezzi 9

3.5.1 Software 9

3.5.2 Hardware 9

4 Progettazione 9

4.1 Design dell’architettura del sistema 9

4.2 Design dei dati e database 9

4.3 Design delle interfacce 12

4.4 Design procedurale 12

5 Implementazione 13

6 Test 13

6.1 Protocollo di test 13

6.2 Risultati test 15

6.3 Mancanze/limitazioni conosciute 15

7 Consuntivo 15

8 Conclusioni 17

8.1 Sviluppi futuri 17

8.2 Considerazioni personali 17

9 Bibliografia 18

9.1 Bibliografia per articoli di riviste: 18

9.2 Bibliografia per libri 18

9.3 Sitografia 18

10 Glossario 19

11 Indice delle figure 19

12 Allegati 19

* **Introduzione**
* **Informazioni sul progetto**

Allievo: Leonardo Sciara

Docente Responsabile: Geo Petrini

SAMT Sezione Informatica 3BB Modulo 306

Inizio: 04.09.2024

Termine:

* **Abstract**

È una breve e accurata rappresentazione dei contenuti di un documento, senza notazioni critiche o valutazioni. Lo scopo di un abstract efficace dovrebbe essere quello di far conoscere all’utente il contenuto di base di un documento e metterlo nella condizione di decidere se risponde ai suoi interessi e se è opportuno il ricorso al documento originale.

Può contenere alcuni o tutti gli elementi seguenti:

* **Background/Situazione iniziale**
* **Descrizione del problema e motivazione**: Che problema ho cercato di risolvere? Questa sezione dovrebbe includere l'importanza del vostro lavoro, la difficoltà dell'area e l'effetto che potrebbe avere se portato a termine con successo.
* **Approccio/Metodi**: Come ho ottenuto dei progressi? Come ho risolto il problema (tecniche…)? Quale è stata l’entità del mio lavoro? Che fattori importanti controllo, ignoro o misuro?
* **Risultati**: Quale è la risposta? Quali sono i risultati? Quanto è più veloce, più sicuro, più economico o in qualche altro aspetto migliore di altri prodotti/soluzioni?
* **Scopo**

L’obiettivo del progetto è sviluppare un'applicazione web interattiva che consenta agli utenti di registrarsi inserendo il proprio nome, scegliere una categoria e una difficoltà per partecipare a un quiz, e infine visualizzare i propri risultati. L'applicazione mira a fornire un'esperienza utente semplice, intuitiva e coinvolgente, implementando la gestione di dati utente e la memorizzazione delle informazioni in un file XML per monitorare le prestazioni di ciascun giocatore.

* **Analisi**
* **Analisi del dominio**

l dominio di questo progetto riguarda i sistemi di quiz interattivi online che permettono a un utente di:

* Registrarsi inserendo il proprio nome.
* Scegliere categorie di domande (es. sport, storia, cultura), livelli di difficoltà e quantità di domande.
* Rispondere a delle domande con più opzioni di risposta.
* Visualizzare i punteggi accumulati al termine del quiz.
* Visualizzare i dettagli di ogni domanda.

Il sistema sarà gestito attraverso JAVA

* **Analisi e specifica dei requisiti**

Inserimento nome giocatore:

* Al caricamento nella pagina Username, l'utente deve inserire il proprio nome tramite un input text che viene salvato in un file data
* Se l'utente non inserisce niente, il bottone “Impostazioni” non compare

Inserimento impostazioni:

* file:
* Dopo aver premuto “Impostazioni”, l’utente va nel file Impostazioni
* Il file contiene 4 tasti; "Categoria", “Difficoltà”, “Quantità Domande” e “Inizio Gioco”
* Se l'utente non ha selezionato una categoria, una difficoltà o la quantità di domande, “Inizia gioco” non appare
* Categoria:
* L’utente sceglie una tra 6 categorie “Sport”, “Storia”, “Cinema”, “Scienza”, “Musica” e “Matematica” tramite dei RadioButton
* Difficoltà:
* L'utente sceglie una tra 3 difficoltà "Facile", "Medio" e "Difficile" tramite uno slider
* Quantità domande:
* L’utente sceglie un numero tra “1” e “30” tramite uno slider

Inizio del gioco:

* Quando l'utente preme su "Inizia il gioco", seleziona le quantità di domande e risposte dal file data e poi inizia il quiz
* Le domande cambiano in base alla categoria e alla difficoltà scelta
* Ogni domanda ha più possibili risposte, delle quali una o più sono corrette
* Per rispondere alle domande ci sono 30 secondi a disposizione, dopo i 30 secondi viene calcolato come risposta sbagliata
* Le risposte sbagliate vengono selezionate casualmente in base alla domanda
* Per ogni risposta corretta, viene assegnato un punteggio al giocatore
* Domande e risposte possono essere anche con delle immagini

Inserimento risposte:

* Le riposte saranno sottoforma di checkbox, se con una risposta singola, e sottoforma di radiobutton, se con più risposte
* Per passare alla domanda successiva, bisogna avere dato al meno una risposta, altrimenti non apparirà nessun bottone
* Ogni volta che si avrà selezionato una o più risposte, per verificare la correttezza si dovrà cliccare “Visualizza Risposta”
* Se l’utente non ha selezionato al meno una risposta, il bottone “Visualizza Risposta” non appare
* Dopo aver premuto “Visualizza Risposta”, il testo del bottone cambierà in “Procedi”, passando alla domanda successiva

Visualizzazione dei risultati:

* Al termine del quiz, l’utente può visualizzare la classifica dei 10 migliori giocatori e il suo punteggio totale totalizzato
* L’utente può scegliere il bottone “Visualizza Dettagli", che porta in un file dove ti dice i punteggi fatti per ogni domanda e le risposte corrette a quella domanda
* Dopo aver visualizzato i dettagli dei punteggi, l’utente deve cliccare “Indietro” per Tornare alla pagina di visualizzazione dei risultati
* L’utente può scegliere il bottone "Nuova Partita", che riporta al file per selezionare categorie, difficoltà e quantità di domande diverse
* L’utente può scegliere il bottone "Nuovo Giocatore", che riporta allo Username file per inserire un nuovo giocatore
* L’utente può scegliere il bottone "Chiudi Gioco", che chiude il gioco

|  |  |
| --- | --- |
| **Req-01** | |
| **Nome** | Creazione interfaccia banca dati |
| **Priorità** | 1 |
| **Versione** | 1.0 |
| **Note** | Si necessitano i permessi di root / Dipende dal requisito Req-001 (Creazione DB) |
| **Sotto requisiti** | |
| **Req-012\_1** | Si necessita una maschera di login |
| **Req-012\_2** | Si dovranno poter immettere nuovi allievi |
| **Req-012\_3** | Dovrà essere possibile la ricerca di allievi |

**Spiegazione elementi tabella dei requisiti:**

**ID**: identificativo univoco del requisito

**Nome**: breve descrizione del requisito

**Priorità**: indica l’importanza di un requisito nell’insieme del progetto, definita assieme al committente. Ad esempio, poter disporre di report con colonne di colori diversi ha priorità minore rispetto al fatto di avere un database con gli elementi al suo interno. Solitamente si definiscono al massimo di 2-3 livelli di priorità.

**Versione**: indica la versione del requisito. Ogni modifica del requisito avrà una versione aggiornata.

Sulla documentazione apparirà solamente l’ultima versione, mentre le vecchie dovranno essere inserite nei diari.

**Note**: eventuali osservazioni importanti o riferimenti ad altri requisiti.

**Sotto requisiti**: elementi che compongono il requisito.

* **Use case**

I casi d’uso rappresentano l’interazione tra i vari attori e le funzionalità del prodotto.

* **Pianificazione**

Prima di stabilire una pianificazione bisogna avere almeno una vaga idea del modello di sviluppo che si intende adottare. In questa sezione bisognerà inserire il modello concettuale di sviluppo che si seguirà durante il progetto. Gli elementi di riferimento per una buona pianificazione derivano da una scomposizione top-down della problematica del progetto.

La pianificazione può essere rappresentata mediante un diagramma di Gantt:

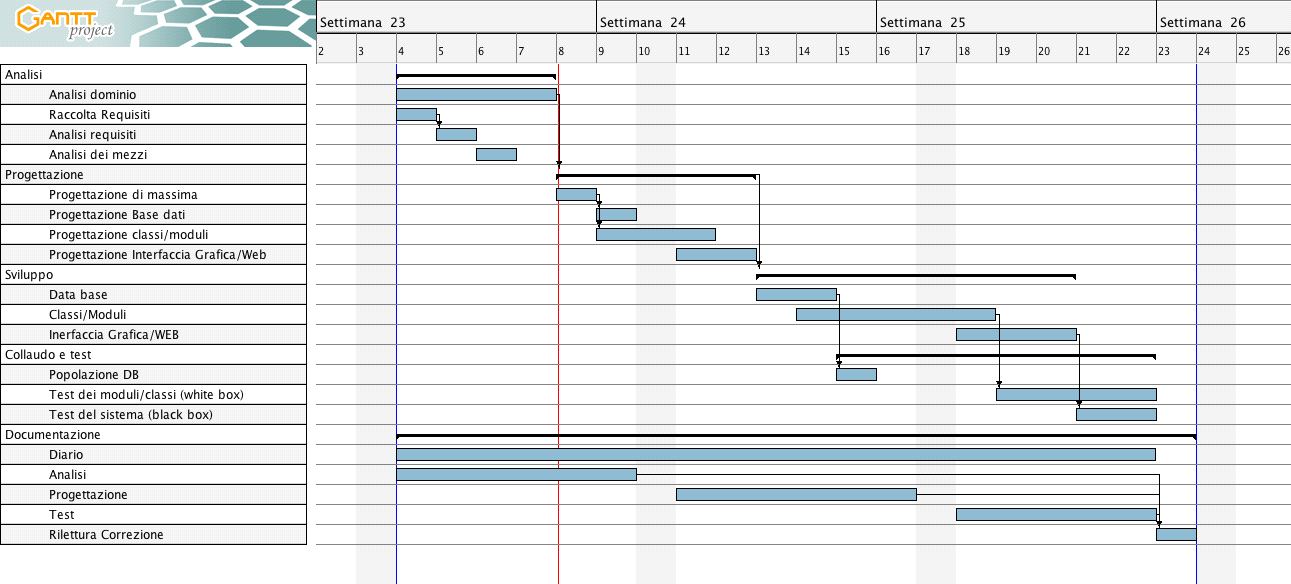


Figura 1 Esempio di diagramma di Gantt

Se si usano altri metodi di pianificazione (p.es. scrum), dovranno apparire in questo capitolo.

Stile orizzontale

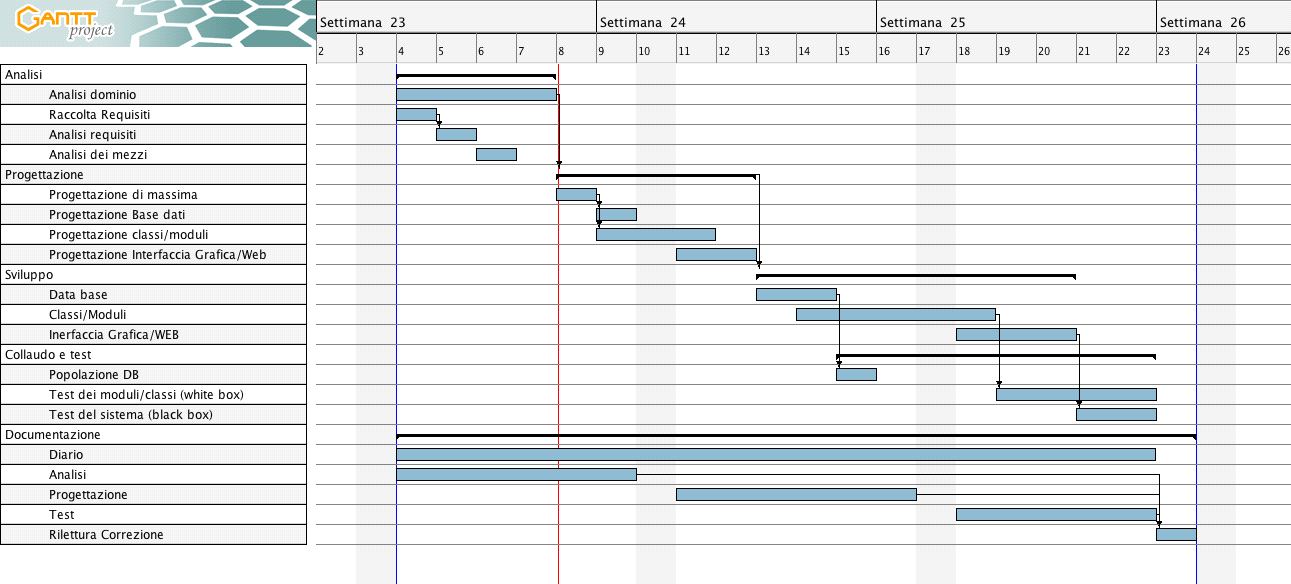


Figura 2 Esempio di diagramma di Gantt

* **Analisi dei mezzi**

Elencare e descrivere i mezzi disponibili per la realizzazione del progetto. Ricordarsi di sempre descrivere nel dettaglio le versioni e il modello di riferimento.

* **Software**

SDK, librerie, tools utilizzati per la realizzazione del progetto e eventuali dipendenze.

* **Hardware**

Su quale piattaforma dovrà essere eseguito il prodotto? Che hardware particolare è coinvolto nel progetto? Che particolarità e limitazioni presenta? Che HW sarà disponibile durante lo sviluppo?

* **Progettazione**

Questo capitolo descrive esaustivamente come deve essere realizzato il prodotto fin nei suoi dettagli. Una buona progettazione permette all’esecutore di evitare fraintendimenti e imprecisioni nell’implementazione del prodotto.

* **Design dell’architettura del sistema**

Descrive:

* La struttura del programma/sistema lo schema di rete...
* Gli oggetti/moduli/componenti che lo compongono.
* I flussi di informazione in ingresso ed in uscita e le relative elaborazioni. Può utilizzare *diagrammi di flusso dei dati* (DFD).
* Eventuale sitemap
* **Design dei dati e database**

Descrizione delle strutture di dati utilizzate dal programma in base agli attributi e le relazioni degli oggetti in uso.

Schema E-R, schema logico e descrizione.

Se il diagramma E-R viene modificato, sulla doc dovrà apparire l’ultima versione, mentre le vecchie saranno sui diari.

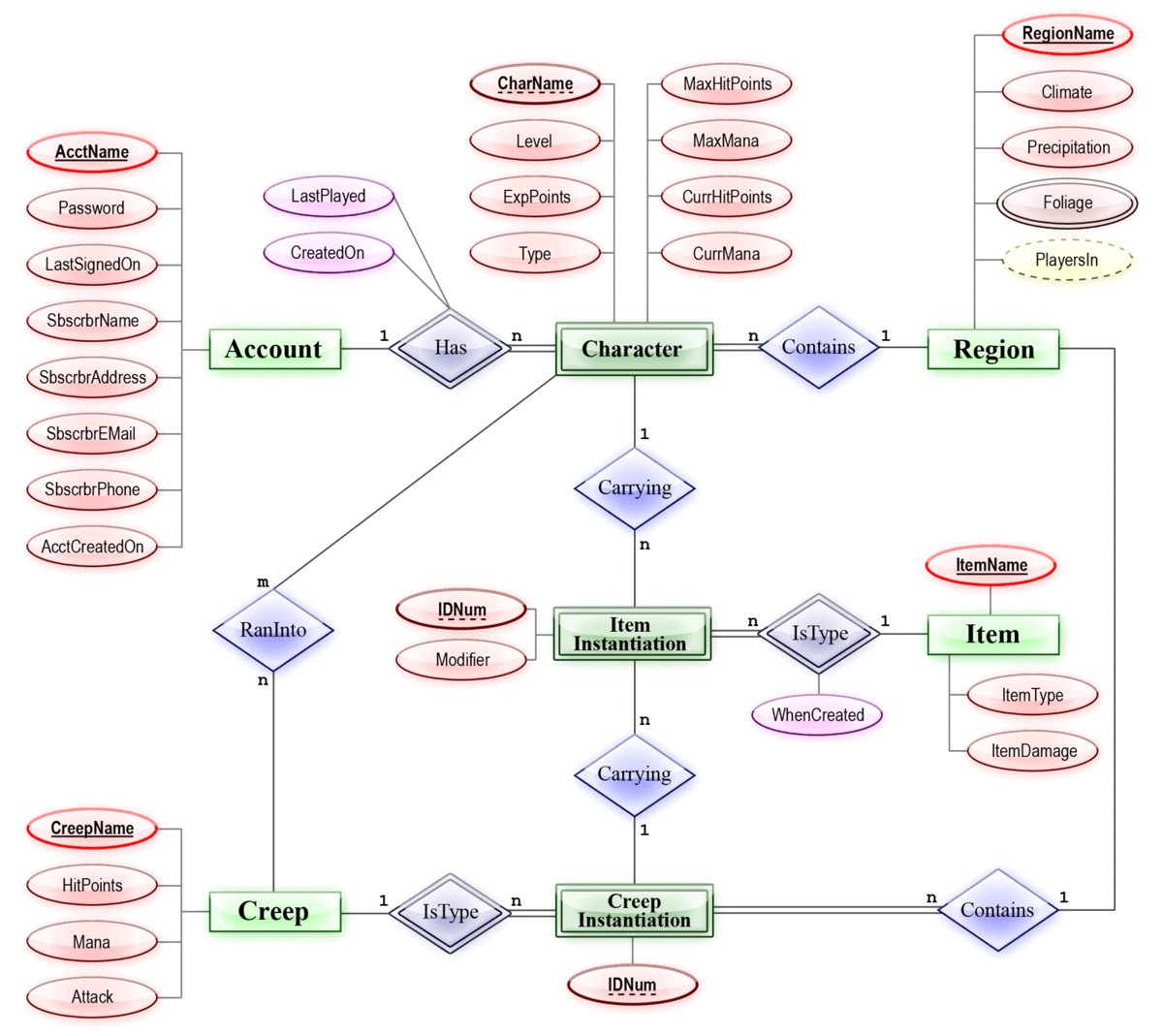
Diagramma Chen

Figura 3 Diagramma ER Chen

Diagramma Barker/Bachmann

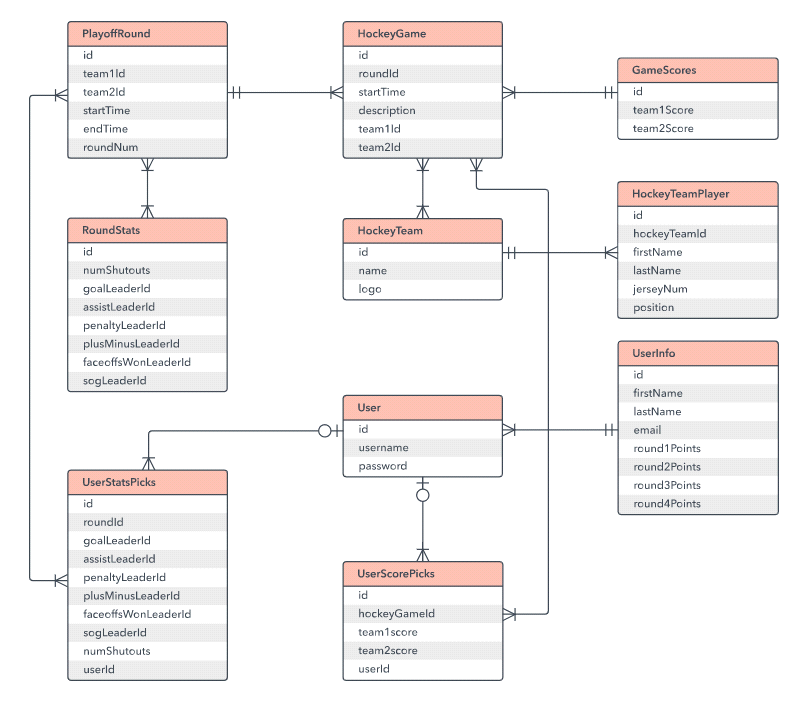


Figura 4 Diagramma ER Barker

* **Design delle interfacce**

Descrizione delle interfacce interne ed esterne del sistema e dell’interfaccia utente. La progettazione delle interfacce è basata sulle informazioni ricavate durante la fase di analisi e realizzata tramite mockups.

* **Design procedurale**

Descrive i concetti dettagliati dell’architettura/sviluppo utilizzando ad esempio:

* Diagrammi di flusso e Nassi.
* Tabelle.
* Classi e metodi.
* Tabelle di routing
* Diritti di accesso a condivisioni …

Questi documenti permetteranno di rappresentare i dettagli procedurali per la realizzazione del prodotto.

* **Implementazione**

In questo capitolo dovrà essere mostrato come è stato realizzato il lavoro. Questa parte può differenziarsi dalla progettazione in quanto il risultato ottenuto non per forza può essere come era stato progettato.

Sulla base di queste informazioni il lavoro svolto dovrà essere riproducibile.

In questa parte è richiesto l’inserimento di codice sorgente - Print Screen - di maschere solamente per quei passaggi particolarmente significativi e/o critici.

Inoltre, dovranno essere descritte eventuali varianti di soluzione o scelte di prodotti con motivazione delle scelte.

Non deve apparire nessuna forma di guida d’uso di librerie o di componenti utilizzati. Eventualmente questa va allegata.

Per eventuali dettagli si possono inserire riferimenti ai diari.

* **Test**
* **Protocollo di test**

Definire in modo accurato tutti i test che devono essere realizzati per garantire l’adempimento delle richieste formulate nei requisiti. I test fungono da garanzia di qualità del prodotto. Ogni test deve essere ripetibile alle stesse condizioni.

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case**  **Riferimento** | TC-001  REQ-012 | **Nome** | Import a card with KIC, KID and KIK keys, but not shown with the GUI |
| **Descrizione** | Import a card with KIC, KID and KIK keys with no obfuscation, but not shown with the GUI | | |
| **Prerequisiti** | Store on local PC: Profile\_1.2.001.xml (appendix n\_n) and Cards\_1.2.001.txt (appendix n\_n).  PIN (OTA\_VIEW\_PIN\_PUK\_KEY) and ADM (OTA\_VIEW\_ADM\_KEY) user right not set. | | |
| **Procedura** | * Go to “Cards manager” menu,  in main page click “Import Profiles” link, Select the “1.2.001.xml” file, Import the Profile * Go to “Cards manager” menu,  in main page click “Import Cards” link, Select the “1.2.001.txt” file, Delete the cards,  Select the “1.2.001.txt” file, Import the cards * Research the “41795924770” Card, Click the imsi card link Check the card details * Execute the SQL: SELECT imsi, dir, keyset, cntr, rawtohex(kickey), rawtohex(kidkey), rawtohex(kikkey), rawtohex(chv), rawtohex(dap)FROM otacardkey a where imsi='340041795924770' ORDER BY keyset; | | |
| **Risultati attesi** | Keys visible in the DB (OtaCardKey) but not visible in the GUI (Card details) | | |

* **Risultati test**

Tabella riassuntiva in cui si inseriscono i test riusciti e non del prodotto finale. Se un test non riesce e viene corretto l’errore, questo dovrà risultare nel documento finale come riuscito (la procedura della correzione apparirà nel diario), altrimenti dovrà essere descritto l’errore con eventuali ipotesi di correzione.

|  |  |  |
| --- | --- | --- |
| **Test Case** | **Risultato ottenuto** | **Stato** |
| TC-001 | Testo o immagine a conferma del test | Passato |
| TC-002 | Testo o immagine a conferma del test | Fallito |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

* **Mancanze/limitazioni conosciute**

Descrizione con motivazione di eventuali elementi mancanti o non completamente implementati, al di fuori dei test case. Non devono essere riportati gli errori e i problemi riscontrati e poi risolti durante il progetto.

* **Consuntivo**

Consuntivo del tempo di lavoro effettivo e considerazioni riguardo le differenze rispetto alla pianificazione (cap. 1.7) (ad esempio Gantt consuntivo).

Stile orizzontale

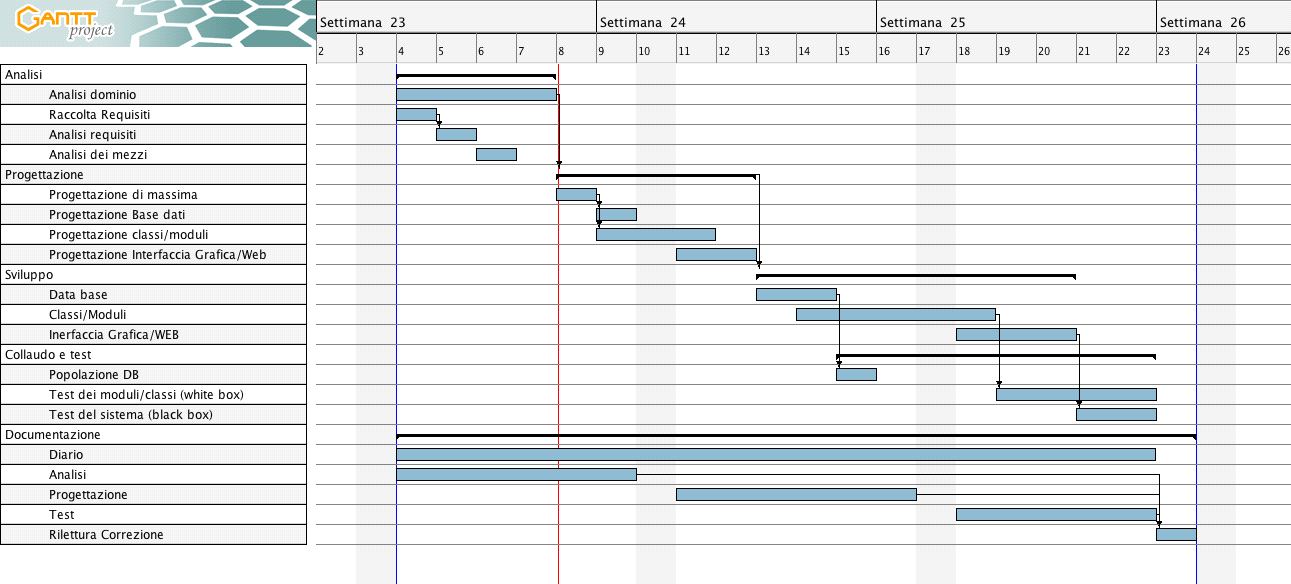


Figura 5 Esempio di diagramma di Gantt consuntivo

* **Conclusioni**

Quali sono le implicazioni della mia soluzione? Che impatto avrà? Cambierà il mondo? È un successo importante? È solo un’aggiunta marginale o è semplicemente servita per scoprire che questo percorso è stato una perdita di tempo? I risultati ottenuti sono generali, facilmente generalizzabili o sono specifici di un caso particolare? ecc.

* **Sviluppi futuri**

Migliorie o estensioni che possono essere sviluppate sul prodotto.

* **Considerazioni personali**

Cosa ho imparato in questo progetto? ecc.

* **Bibliografia**
* **Bibliografia per articoli di riviste:**
* Cognome e nome (o iniziali) dell’autore o degli autori, o nome dell’organizzazione,
* Titolo dell’articolo (tra virgolette),
* Titolo della rivista (in italico),
* Anno e numero
* Pagina iniziale dell’articolo.
* **Bibliografia per libri**
* Cognome e nome (o iniziali) dell’autore o degli autori, o nome dell’organizzazione,
* Titolo del libro (in italico),
* ev. Numero di edizione,
* Nome dell’editore,
* Anno di pubblicazione,
* ISBN.
* **Sitografia**
* URL del sito (se troppo lungo solo dominio, evt completo nel diario),
* Eventuale titolo della pagina (in italico),
* Data di consultazione (GG-MM-AAAA).

**Esempio:**

* <http://standards.ieee.org/guides/style/section7.html>, *IEEE Standards Style Manual*, 07-06-2008.

* **Glossario**

|  |  |
| --- | --- |
| **Termine** | **Significato** |
| ASCII | American Standard Code for Information Interchange, codice per la codifica di caratteri. |
|  |  |
|  |  |
|  |  |
|  |  |
| ZFS | Zettabyte File System, filesystem opernsource a 128 bit con funzionalità avanzate. |

* **Indice delle figure**

Figura 1 Esempio di diagramma di Gantt 7

Figura 2 Esempio di diagramma di Gantt 8

Figura 3 Diagramma ER Chen 10

Figura 4 Diagramma ER Barker 11

Figura 5 Esempio di diagramma di Gantt consuntivo 16

* **Allegati**

Elenco degli allegati, esempio:

* Diari di lavoro
* Codici sorgente/documentazione macchine virtuali
* Istruzioni di installazione del prodotto (con credenziali di accesso) e/o di eventuali prodotti terzi
* Documentazione di prodotti di terzi
* Eventuali guide utente / Manuali di utilizzo
* Mandato e/o QdC
* Prodotto
* …