Word Clock

1 Introduzione 3

1.1 Informazioni sul progetto 3

1.2 Abstract 3

1.3 Scopo 3

Analisi 4

1.4 Analisi del dominio 4

1.5 Analisi e specifica dei requisiti 4

1.6 Pianificazione 5

1.7 Analisi dei mezzi 6

1.7.1 Software 6

1.7.2 Hardware 6

2 Progettazione 7

2.1 Design dell’architettura del sistema 7

2.2 Design dei dati e database 7

2.3 Design delle interfacce 7

2.4 Design procedurale 7

3 Implementazione 8

4 Test 8

4.1 Protocollo di test 8

4.2 Risultati test 8

4.3 Mancanze/limitazioni conosciute 9

5 Consuntivo 9

6 Conclusioni 9

6.1 Sviluppi futuri 9

6.2 Considerazioni personali 9

7 Bibliografia 9

7.1 Bibliografia per articoli di riviste: 9

7.2 Bibliografia per libri 9

7.3 Sitografia 9

8 Allegati 10

# Introduzione

## Informazioni sul progetto

* Allievi: Gabriele Alessi, Mattia Lazzaroni, Paolo Claudio Weishaupt.   
  Superiore professionale: Adriano Barchi.
* Scuola d’Arti e Mestieri di Trevano, sezione Informatica, classe 3, modulo 306.
* Data inizio: 13.02.2019.  
  Data fine: 17.05.2019.

## Abstract

## Scopo

Lo scopo di questo progetto è quello di creare un prototipo funzionante di un orologio a parole (Word Clock) per poi in seguito presentarlo a una commissione che deciderà se stanziare i fondi per poterne realizzare una versione più grande montata sulla torre sinistra della scuola.

# Analisi

## Analisi del dominio

L’entrata del quarto piano è poco invitante e non ci sono esposti dei progetti realizzati dagli allievi.   
Quindi questo prodotto colmerebbe questo spazio vuoto mostrando qualcosa di utile (l’orario) in maniera diversa e interessante. Il contesto in cui il prodotto dovrà funzionare è adatto e sarebbe l’ideale metterlo all’inizio del corridoio. Attualmente esiste un piccolo orologio binario nella zona degli elettronici, ma questa soluzione è più chiara e leggibile, quindi non ci sarebbero problemi nel metterlo in mostra.  
Gli utenti sono principalmente gli allievi del quarto piano, che arrivando al corridoio verrebbero a conoscenza dell’orario solamente alzando lo sguardo e sapere se sono in ritardo per la lezione o meno. Quindi gli utenti del prodotto non necessitano e non necessiteranno di particolari competenze (come con l’orologio binario).

## Analisi e specifica dei requisiti

|  |  |
| --- | --- |
| **ID: REQ-001** | |
| **Nome** | Verifica componenti hardware |
| **Priorità** | 1 |
| **Versione** | 1.0 |
| **Note** | È necessario verificare che i componenti hardware (Fishino e LED) funzionino correttamente. |

|  |  |
| --- | --- |
| **ID: REQ-002** | |
| **Nome** | Implementare il Word Clock |
| **Priorità** | 1 |
| **Versione** | 1.0 |
| **Note** | È necessario implementare il Word Clock in modo che mostri l’orario sotto forma di linguaggio parlato. |
| **Sotto requisiti** | |
| **001** | Scrivere il testo con le differenze di 5 minuti |
| **002** | Mostrare i secondi tramite l’utilizzo di 12 LED, quindi 5 secondi per ogni pallino acceso. |
| **003** | Si dovrà poter controllare e impostare l’orario e le impostazione tramite il modello fisico. |
| **004** | Cambiare i colori dei LED in base all’orario. |

|  |  |
| --- | --- |
| **ID: REQ-003** | |
| **Nome** | Controllo via web |
| **Priorità** | 2 |
| **Versione** | 1.0 |
| **Note** | Si dovrà poter controllare e impostare l’orario e le impostazione tramite una pagina web. |

## Pianificazione

La pianificazione del progetto è stata effettuata mediante la realizzazione di un diagramma di Gantt.



## Analisi dei mezzi

### Software

I software utilizzati per la realizzazione di questo progetto sono:

* Word 2016
* Power Point 2016
* Arduino 1.8.7
* GanttProject 2.8.9 Pilsen
* Google Chrome 71.0.3578.98
* Atom 1.34.0

### Hardware

* Computer Mattia Lazzaroni:
  + Modello: Acer Aspire E 15
  + Processore: Intel Core i7 7500u
  + RAM: 16GB LPDDR4 2133 MHz
  + GPU: NVIDIA 940MX
  + SSD: 256 GB
* Computer Gabriele Alessi:
  + Modello: HP ENVY Notebook,
  + Processore: Intel Core i7
  + RAM: 16GB
* Computer Paolo Weishaupt:
  + Modello: Huawei MateBook X Pro
  + Processore: Intel Core i7-8550U
  + RAM: 8GB
  + Display 13.9” 3000x2000px
  + SSD: 512 GB
* Fishino UNO REV2:
  + Compatibile al 100% con Arduino UNO
  + Modulo WiFi integrato
  + Slot per schede MicroSD integrato
  + Modulo RTC integrato
  + Connettore sfalsato per facilitare l’uso con breadboards
  + Sezione di alimentazione a 3.3V potenziata
  + Compatibilità con le schede millefori
  + Schemi elettrici, file Eagle e pinout della scheda
* Adafruit NeoPixel

# 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.

## 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:** | 1. Go to “Cards manager” menu,  in main page click “Import Profiles” link, Select the “1.2.001.xml” file, Import the Profile 2. 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 3. Research the “41795924770” Card, Click the imsi card link Check the card details 4. 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.

## 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 Gannt 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 libri

* Massimo Del Fedele, *Fishino*, Futura Group srl, 2017, 978-88-909529-5-1.

## Sitografia

* <https://fishino.it/fishino-uno-it.html>, 15.02.2019
* <https://atom.io/packages/git-plus>, 15.02.2019

# 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
* …