Documentazione

[1 Introduzione 2](#__RefHeading___Toc1221_2866232661)

[1.1 Informazioni sul progetto 2](#__RefHeading___Toc1223_2866232661)

[1.2 Abstract 2](#__RefHeading___Toc1225_2866232661)

[1.3 Scopo 2](#__RefHeading___Toc1227_2866232661)

[2 Analisi 3](#__RefHeading___Toc1229_2866232661)

[2.1 Analisi del dominio 3](#__RefHeading___Toc1231_2866232661)

[2.2 Analisi e specifica dei requisiti 3](#__RefHeading___Toc1233_2866232661)

[2.3 Use case 6](#__RefHeading___Toc1235_2866232661)

[2.4 Pianificazione 6](#__RefHeading___Toc1237_2866232661)

[2.5 Analisi dei mezzi 7](#__RefHeading___Toc1239_2866232661)

[2.5.1 Software 7](#__RefHeading___Toc1241_2866232661)

[2.5.2 Hardware 7](#__RefHeading___Toc1243_2866232661)

[3 Progettazione 7](#__RefHeading___Toc1245_2866232661)

[3.1 Design dell’architettura del sistema 7](#__RefHeading___Toc1247_2866232661)

[3.2 Design dei dati e database 8](#__RefHeading___Toc1249_2866232661)

[3.3 Design delle interfacce 8](#__RefHeading___Toc1251_2866232661)

[3.4 Design procedurale 8](#__RefHeading___Toc1253_2866232661)

[4 Implementazione 9](#__RefHeading___Toc1255_2866232661)

[5 Test 9](#__RefHeading___Toc1257_2866232661)

[5.1 Protocollo di test 9](#__RefHeading___Toc1259_2866232661)

[5.2 Risultati test 10](#__RefHeading___Toc1261_2866232661)

[5.3 Mancanze/limitazioni conosciute 10](#__RefHeading___Toc1263_2866232661)

[6 Consuntivo 10](#__RefHeading___Toc1265_2866232661)

[7 Conclusioni 10](#__RefHeading___Toc1267_2866232661)

[7.1 Sviluppi futuri 10](#__RefHeading___Toc1269_2866232661)

[7.2 Considerazioni personali 10](#__RefHeading___Toc1271_2866232661)

[8 Bibliografia 10](#__RefHeading___Toc1273_2866232661)

[8.1 Bibliografia per articoli di riviste: 10](#__RefHeading___Toc1275_2866232661)

[8.2 Bibliografia per libri 10](#__RefHeading___Toc1277_2866232661)

[8.3 Sitografia 10](#__RefHeading___Toc1279_2866232661)

[9 Allegati 11](#__RefHeading___Toc1281_2866232661)

# Introduzione

## Informazioni sul progetto

Allievi coinvolti nel progetto: Stefano Mureddu

Classe: Informatica 3AC Scuola Arti e Mestieri Trevano

Docenti responsabili: Geo Petrini

Data inizio: 03.09.2020  
Data consegna: 17.12.2020

## Abstract

Cos’è un un net scanner? Un net scanner non è altro che un programma che ti permette di scoprire gli apparecchi attaccati alla rete locale e su quali porte questi sono accessibili. Tramite una semplice interfaccia grafica in cui bisogna immettere un range di ip e le porte si può controllare se ci sono dispositivi attaccati alla rete a cui sei connesso tu, permettendoti così di scoprire se è effettivamente la tua rete ad essere lenta o è colpa del vicino che si è attaccato per scaricarsi i film.

## Scopo

Lo scopo del progetto NetScanner è quello di creare una applicazione che funge appunto da scanner per la rete locale.

Questa applicazione dovrà permette in modo semplice di scoprire se ci sono apparecchi connessi alla rete e con quale porta, tramite un semplice interfaccia grafica in cui immettere un range di ip su cui effettuare la scansione e le varie porte da controllare.

# Analisi

## Analisi del dominio

È stata richiesta un’applicazione che tramite un’interfaccia grafica permetta all’utente di scoprire le macchine connesse alla rete e le loro porte e generare da esso un report. Esistono già un sacco di applicazioni simili in internet e anche io ho già fatto qualcosa di simile in power shell.

## Analisi e specifica dei requisiti

L’applicativo deve avere un’interfaccia grafica il più user friendly possibile con vari pulsanti e icone per cambiare il range di IP e di porte sul quale viene eseguito il controllo, quali informazioni riportare nel report ed eventualmente il metodo di ricerca da usare.

I report devono venir generati ogni volta che si fa un check.

|  |  |
| --- | --- |
| **ID: REQ-001** | |
| **Nome** | Applicativo con interfaccia grafica |
| **Priorità** | 1 |
| **Versione** | 1.0 |
| **Note** |  |

|  |  |
| --- | --- |
| **ID: REQ-002** | |
| **Nome** | Scoprire l’ip macchine connesse alla rete |
| **Priorità** | 1 |
| **Versione** | 1.0 |
| **Note** |  |

|  |  |
| --- | --- |
| **ID: REQ-003** | |
| **Nome** | Scoprire le porte aperte delle macchine connesse alla rete |
| **Priorità** | 1 |
| **Versione** | 1.0 |
| **Note** |  |

|  |  |
| --- | --- |
| **ID: REQ-004** | |
| **Nome** | Generare dei report |
| **Priorità** | 1 |
| **Versione** | 1.0 |
| **Note** |  |

|  |  |
| --- | --- |
| **ID: REQ-005** | |
| **Nome** | Scegliere il range di ip su cui effettuare il controllo |
| **Priorità** | 2 |
| **Versione** | 1.0 |
| **Note** |  |

|  |  |
| --- | --- |
| **ID: REQ-006** | |
| **Nome** | Scegliere il range di porte da controllare |
| **Priorità** | 2 |
| **Versione** | 1.0 |
| **Note** |  |

|  |  |
| --- | --- |
| **ID: REQ-007** | |
| **Nome** | Scegliere quali informazioni mostrare nei report |
| **Priorità** | 2 |
| **Versione** | 1.0 |
| **Note** |  |
| **ID: REQ-008** | |
| **Nome** | Scegliere i metodi di ricerca |
| **Priorità** | 3 |
| **Versione** | 1.0 |
| **Note** |  |

## Use case



Un utente avviando l’applicazione dovrà inserire due ip che fungeranno da range e delle porte su cui effettuare la scansione. Dopo la scansione verrà generato automaticamente un report sulla base delle informazione volute dall’utente.

## Pianificazione

|  |
| --- |
| Figura 1: Gantt preventivo del progetto |

### 2.4.1



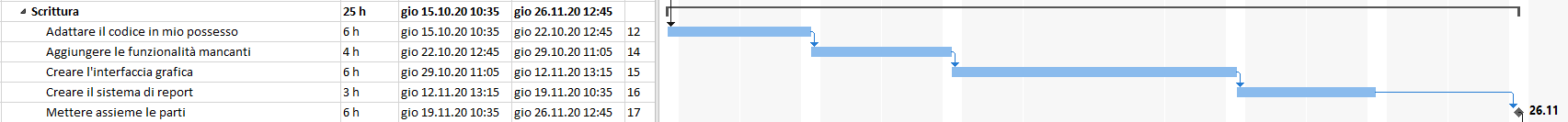
Ho deciso di suddividere i progetto in 4 fasi. La prima consiste nella preparazione per l’inizio vero del progetto, facendo requisiti progettazione e preparando i mezzi con cui fare il progetto, come la macchina virtuale e i vari programmi per scrivere il programma.

### 2.4.2



La seconda parte consiste invece nel progettare nei dettagli il progetto e informarsi su come farlo in modo più comodo ed efficace. Infatti c’è una buona parte in cui si raccolgono solo informazioni sui modi per creare report, interfacce grafiche, e modi per scannerizzare, in modo da scegliere il linguaggio più comodo con cui fare il programma. Infine c’è da fare un schema per le classi su cui basarsi successivamente durante la scrittura del programma.

### 2.4.3



La terza parte è quella in cui si scrive effettivamente il programma. È diviso in sotto parti dove si scrivono i vari pazzi del programma, che si metteranno poi assieme alla fine della fase di scrittura.

### 2.4.4



La quarta e ultima parte consiste nel testare e migliorare l’applicazione. Verso l’inizio ci sarà da testare l’applicazione completa risolvendo poi eventuali bug. Per la sotto parte successiva mi sono tenuto più largo, infatti “aggiungere funzionalità extra” è una cosa che si può fare se tutto va secondo il programma. Se invece si fosse rimasti indietro si potrebbe utilizzare questa sotto fase per continuare a lavorare su parti importanti o fondamentali. In fine ci sono l’ultima revisione con eventuali modifiche e la consegna

La documentazione verrà fatta durante tutto l’andamento del progetto.

## Analisi dei mezzi

### Software

* Google chrome
* NetBeans IDE 8.2
* VMWare workstation player 14

### Hardware

Progetto sviluppato sul PC A423-10

# Progettazione

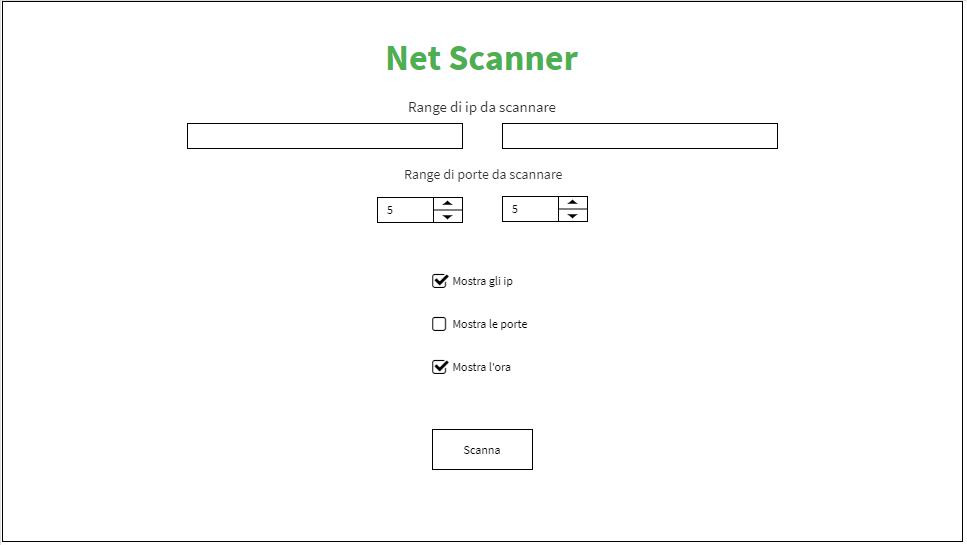
## Design dell’architettura del sistema

L’architettura del NetScanner sarà molto semplice, avrà solo due classi, una contenente la parte grafica(gui) e una con il NetScanner. La parte grafica sarà intuitiva, piccola, e semplice da utilizzare. Questa sarà collegata al NetScanner, che conterrà la parte di codice che permetterà di fare lo scan in base a quello selezionato nella gui.

## Design dei dati e database

I dati ricavati dallo scan nella rete verranno salvati su un file txt, ed eventualmente su un csv.

## Design delle interfacce



Ci sarà una sola interfaccia grafica in cui si potranno inserire i vari parametri per poi cliccare il pulsante scanna e fare partire lo scan.

## 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 articoli di riviste:

1. Cognome e nome (o iniziali) dell’autore o degli autori, o nome dell’organizzazione,
2. Titolo dell’articolo (tra virgolette),
3. Titolo della rivista (in italico),
4. Anno e numero
5. Pagina iniziale dell’articolo,

## Bibliografia per libri

1. Cognome e nome (o iniziali) dell’autore o degli autori, o nome dell’organizzazione,
2. Titolo del libro (in italico),
3. ev. Numero di edizione,
4. Nome dell’editore,
5. Anno di pubblicazione,
6. ISBN.

## Sitografia

1. URL del sito (se troppo lungo solo dominio, evt completo nel diario),
2. Eventuale titolo della pagina (in italico),
3. Data di consultazione (GG-MM-AAAA).

**Esempio:**

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

# 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
* …