CPT

Documentazione del progetto net scanner di Stefano Mureddu

DOCUMENTAZIONEStefano Mureddu

Progetto NetScanner

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 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 informazioni volute dall’utente.

## Pianificazione

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

### 2.4.1



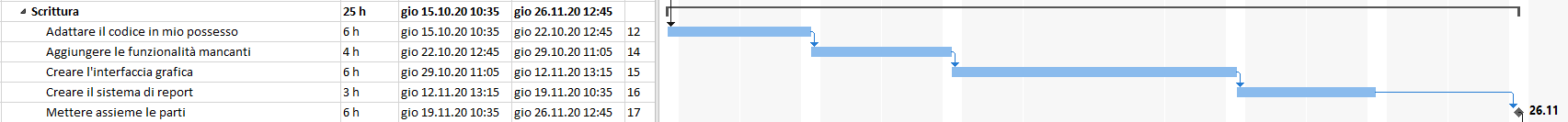
Ho deciso di suddividere il progetto in 4 fasi. La prima consiste nella preparazione per l’inizio vero del progetto, facendo requisiti progettazione e preparando i mezzi con cui fare il progetto, come la macchina virtuale e i vari programmi per scrivere il programma.

### 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
* Java 1.12

### Hardware

Progetto sviluppato sul PC A423-10

# Progettazione

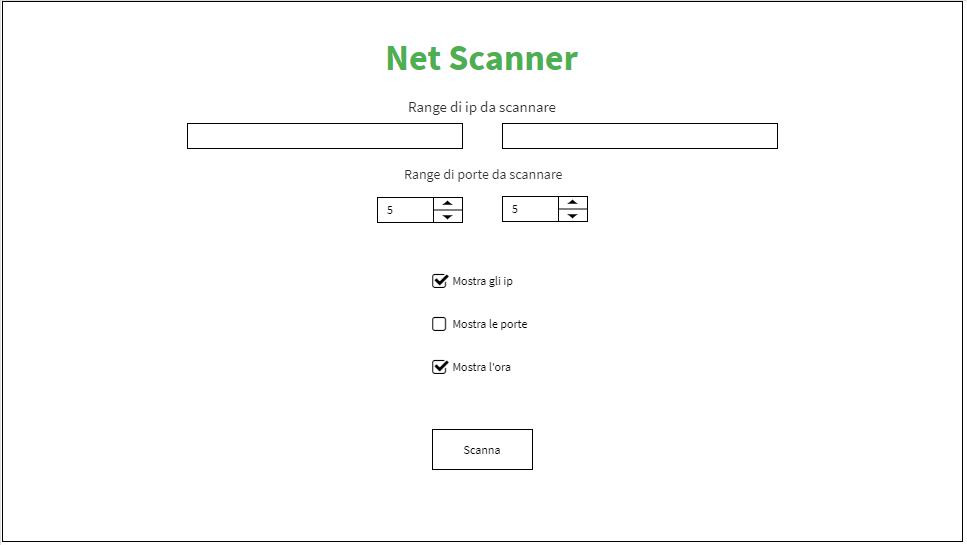
## Design dell’architettura del sistema

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

* NetScanner.java
  + Costruttore
    - Controlli di input
  + pingPort
    - Creazione di oggetti per pingare
  + isValid
    - Valida gli indirizzi ip
  + isValidPort
    - Valide le porte
  + cicli
    - Calcola i cicli necessari per scannerizzare tutto
* Finestra.java
  + Interfaccia grafica
  + Avvio del programma e ciclo di per pingare quando si clicca il pulsante

# Implementazione

Sono partito con l’intento di utilizzare il mio vecchio codice di script come base ma alla fine questo è risultato inutile. Infatti il metodo di ping utilizzato in java è un po’ diverso da quello di script. Inoltre dovendo anche fare un ping sulle porte il modo in cui lo facevo su script non poteva funzionare.

Inoltre, avendo alla fine netBeans sulla macchina della scuola ho abbandonato la macchina virtuale da me precedentemente creata.

Come prima cosa ho implementato una classe NetScanner da creare poi nell’interfaccia grafica dove nel costruttore si assegnano i dati(range di ip e porte).

Successivamente ho provato un metodo per fare un ping agli ip e ho implementato un metodo che permette di ciclare gli ip così da scannerizzarli tutti.

Dopo questo ho lavorato sull’interfaccia grafica così da poter inserire i dati da essa. Durante questa fase mi sono però reso conto che il metodo che utilizzavo per scannerizzare non prevedeva l’utilizzo delle porte, di conseguenza ho dovuto cambiarlo con quello attuale che permette di farlo.

Successivamente ho lavorato sulla generazione dei report. Il programma infatti aggiunge ad una lista di stringhe solo i parametri voluti e li scrive poi su un file txt.

Dopo questo mi sono concentrato sul controllo dei dati inseriti, creando dei metodi apposta per controllare la validità degli ip e delle porte inserite, e ho fatto successivamente in modo che le porte oltre che singole possano anche essere inserite come range.

Per finire ho aggiunto una scelta che permette di creare un report in formato csv.

Ho anche provato ad aggiungere una barra di caricamento ma questa avrebbe necessitato di threed e il tempo a disposizione era troppo poco per una modifica di queste dimensioni.

# Test

## Protocollo di test

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case:**  **Riferimento**: | TC-001  REQ-001 | **Nome:** | Avvio con interfaccia grafica |
| **Descrizione:** | Avvio del .jar e apparizione dell’interfaccia grafica | | |
| **Prerequisiti:** | Il .jar | | |
| **Procedura:** | 1. Trova netScanner.jar 2. Clicca 2 volte sopra questo | | |
| **Risultati attesi:** | Avvio con interfaccia grafica | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case:**  **Riferimento**: | TC-002  REQ-005 | **Nome:** | Immissione di range di ip |
| **Descrizione:** | Controllo di accettazione parameter di range degli ip | | |
| **Prerequisiti:** | TC-001 | | |
| **Procedura:** | 1. Inserire parametri errati nel range di ip e cliccare scanna  2. Inserire parametri utilizzabili nel range di ip e cliccare scanna | | |
| **Risultati attesi:** | 1 Rifiuto dei dati inseriti. 2 Accettazione dei dati inseriti | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case:**  **Riferimento**: | TC-003  REQ-006 | **Nome:** | Immissione di range di porte |
| **Descrizione:** | Controllo di accettazione parameter di range delle porte | | |
| **Prerequisiti:** | TC-001 | | |
| **Procedura:** | 1. Inserire parametri errati nel range di porte e cliccare scanna  2. Inserire parametri utilizzabili nel range di porte e cliccare scanna | | |
| **Risultati attesi:** | 1 Rifiuto dei dati inseriti. 2 Accettazione dei dati inseriti | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case:**  **Riferimento**: | TC-004  REQ-004 | **Nome:** | Controllo report |
| **Descrizione:** | Avvio scansione con I check box attivi per vedere se crea il report | | |
| **Prerequisiti:** | TC-002, TC-003, | | |
| **Procedura:** | 1. Inserire i dati  2. Attivare I check box  3. Cliccare su scanna  4. Controllare che il report venga generato | | |
| **Risultati attesi:** | File report.txt comparso o sovrascritto nella cartella in cui è programma | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case:**  **Riferimento**: | TC-005  REQ-002 | **Nome:** | Controllo dati report ip |
| **Descrizione:** | Controllo che i dati ip del report siano giusti | | |
| **Prerequisiti:** | TC-004 | | |
| **Procedura:** | 1. Aprire il file report.txt  2. Controllare che ci siano ip che sono stati raggiunti che corrispondo a macchine accese e connesse alla rete | | |
| **Risultati attesi:** | Ip esatti nel report | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case:**  **Riferimento**: | TC-006  REQ-004 | **Nome:** | Controllo dati report porte |
| **Descrizione:** | Controllo che i dati porte del report siano giusti | | |
| **Prerequisiti:** | TC-004 | | |
| **Procedura:** | 1. Aprire il file report.txt  2. Controllare che ci siano porte che sono state raggiunte che corrispondo a quelle aperte nelle machine accese e connesse alla rete | | |
| **Risultati attesi:** | Porte esatte nel report | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case:**  **Riferimento**: | TC-007  REQ-007 | **Nome:** | Inserimento di parametri per il report |
| **Descrizione:** | Controllo che i dati del report siano solo quelli voluti | | |
| **Prerequisiti:** | TC-004 | | |
| **Procedura:** | 1. Dopo aver inserito i range di ip e porte selezionare i check box con i parametri voluti  2. Controllare che nel report ci siano solo i parametri selezionati dai check box | | |
| **Risultati attesi:** | parametri selezionati dai check box nel report | | |

## Risultati test

|  |  |  |
| --- | --- | --- |
| Test | Parametri | Risultato |
| TC-001 | - | Corretto |
| TC-002 | 10.253.20.d, 10.10.10.1 | Corretto |
| TC-003 | 10,das10, 10,80-85 | Corretto |
| TC-004 | 10.20.4.100, 10.20.4.200 80,8080 | Corretto |
| TC-005 | = a sopra | Corretto |
| TC-006 | = a sopra | Corretto |
| TC-007 | = a sopra + ip raggiungibili + porte raggiungibili | Corretto |
| TC-007 | 10.20.4.100, 10.20.4.200 80,8080 + ip non raggiungibili + porte non raggiungibili | Errato, probabilmente il problema è nel fatto che salvo i dati nella lista dividendo di principio ciò che è raggiungibile da ciò che non è raggiungibile |

## Mancanze/limitazioni conosciute

Sicuramente si potevano aggiungere migliorie all’interfaccia grafica per rendere il programma ancora più user friendly, inoltre anche se per fare lo scan ho 2 metodi diversi alla fine uso sempre lo stesso.

Inoltre i report generati con combinazioni poco utilizzate come gli ip non raggiungibili insieme alle porte raggiungibili non solo del tutto corretti.

# Consuntivo

# 

Alcune parti finali secondarie come le migliorie dell’estetica non sono riuscito a farle per mancanza di tempo visto che ne ho impiegato di più per l’

# Conclusioni

La mia soluzione non è altro che una delle tante che già girano su internet, tuttavia a differenza delle altre questa la ho fatta io.

Sicuramente non è quella fatta meglio ma sicuramente svolge il suo compito senza grossi problemi.

## Sviluppi futuri

Una barra di caricamento per sapere a che punto della scansione si è e varie migliorie per rendere l’interfaccia grafica bella.

## Considerazioni personali

Ho sicuramente confermato il fatto che nulla va mai come speri e quindi ho fatto bene a tenermi largo con i tempi, inoltre ho scoperto cose interessanti come l’utilizzo dei socket e di tutto il sistema di ping delle reti utilizzato in java senza librerie esterne. È stata anche la prima volta che utilizzavo un’interfaccia grafica da cui prendere i dati in java, anche se già lo avevo fatto con C#. Escludendo i momenti terribili come tutto quello che centra con proxy e macchine virtuali fare questo progetto è stato anche abbastanza divertente, soprattutto nei momenti in cui quello a cui hai lavorato inizia effettivamente a funzionare. Un’altra cosa che questo progetto mi ha confermato è quanto poco mi piaccia lavorare sulle macchine virtuali visto che, dalle mie esperienze personali, so bene che se di base nulla va come previsto queste non vanno e basta.

# Bibliografia

## Sitografia

* <https://docs.oracle.com/javase/8/docs/api/javax/swing/JProgressBar.html>, *JProgressBar (Java Platform SE 8 ),* 03-12-2020.

* <https://docs.oracle.com/javase/8/docs/technotes/guides/swing/1.4/ftf.html>, *Formatted Text Field,*  19-11-2020.
* <https://www.techiedelight.com/validate-ip-address-java/>, *Validate an IP address in Java - Techie Delight,* 12-11-2020.
* <https://www.w3schools.com/>, *W3Schools,* durante tutta la durata del progetto.
* <https://stackoverflow.com/>, *StackOverflow,* durante tutta la durata del progetto.

# Allegati