LAN Speed Tester

[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: Samuel Agustoni

Classe: Informatica 3AC presso la sede Scuola Arti e Mestieri Trevano

Docenti responsabili: Geo Petrini

Data inizio: 03.09.2020  
Data consegna: 17.12.2020

## Abstract

Hai bisogno di testare le prestazioni di trasmissione della tua rete LAN? Questo software potrebbe essere la soluzione che fa per te! Il LAN Speed Tester è un’applicazione portable multipiattaforma, in grado di fornire una statistica molto dettagliata delle velocità di trasmissione con la quale le macchine all’interno della tua rete comunicano. L’obbiettivo di questo progetto è quello di sviluppare un software semplice, leggero e facilmente trasferibile da un dispositivo all’altro, pensato per testare le prestazioni dei collegamenti di una rete LAN.

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?

Esempio di abstract:

*As the size and complexity of today’s most modern computer chips increase, new techniques must be developed to effectively design and create Very Large Scale Integration chips quickly. For this project, a new type of hardware compiler is created. This hardware compiler will read a C++ program, and physically design a suitable microprocessor intended for running that specific program. With this new and powerful compiler, it is possible to design anything from a small adder, to a microprocessor with millions of transistors. Designing new computer chips, such as the Pentium 4, can require dozens of engineers and months of time. With the help of this compiler, a single person could design such a large-scale microprocessor in just weeks.*

## Scopo

Lo scopo del progetto “LAN Speed Tester” è quello di creare un software facilmente trasferibile e multipiattaforma, in grado di misurare, con una precisione al millisecondo, la velocità con cui una macchina Server e una macchina Client comunicano all’interno di una rete LAN. L’applicazione, al termine dell’esecuzione del programma, dovrà mostrare a schermo, sulla macchina che abbiamo definito come Client, una statistica significativa delle velocità. Contenente il numero di pacchetti mandati, le dimensioni in byte dei vari pacchetti e le medie delle velocità impiegate per ricevere una risposta dal server ad ogni trasmissione. Un software a parer mio molto utile a livello sistemistico in quanto permette al sistemista di tenere sotto controllo il buon funzionamento e l’efficienza della rete.

# Analisi

## Analisi del dominio

È stato richiesto lo sviluppo di un’applicazione multipiattaforma facilmente trasferibile con la funzione di Speed Tester. Ovvero un tester per la velocità di trasmissione all’interno di una rete LAN. Dovrà essere compatibile con la maggior parte dei sistemi operativi Linux/Windows/Unix. Il software non avrà interfaccia grafica per mia scelta, verrà perciò lanciato via terminale da linea di comando.

Concettualmente l’applicazione si comporta in questo modo: una macchina (server) della rete si mette in ascolto su una porta pubblica disponibile, un’altra macchina (client) della stessa rete trasmette al server in ascolto sulla porta precedentemente definita, una o più sequenze di byte sequenzialmente o contemporaneamente. Infine la macchina client produce in output una statistica significativa delle velocità delle varie trasmissioni. In questo modo si possono osservare le prestazioni della rete.

Il software è diviso in due parti, poiché deve avere due ruoli. Una parte per l’utilizzo su client e una parte per l’utilizzo su server. Una volta lanciata l’applicazione sul server, bisogna prima di tutto scegliere il ruolo giusto (SV), dopodiché bisogna scegliere la porta sulla quale il server rimane in ascolto, se non viene scelta alcuna porta viene impostata automaticamente la prima disponibile. Quando il server è pronto a ricevere connessioni viene mostrato un avvito. Fatto ciò, il software viene lanciato sul client che sceglie il proprio ruolo, manualmente si inserisce la porta sulla quale il server è in ascolto e il numero e le dimensioni dei vari pacchetti di byte che verranno trasmessi al server.

## Analisi e specifica dei requisiti

Il progettista, dopo aver ricevuto il mandato, in collaborazione con il committente redige una lista di requisiti. Durante questi incontri, tramite interviste (da inserire nei diari), il progettista deve cercare di rispondere alle seguenti domande:

* Quali sono i bisogni del committente?
* Quali funzioni deve svolgere il prodotto?
* Come devono essere implementate?
* L’utente, come vorrebbe/dovrebbe interagire con il prodotto?
* Come verrà utilizzato il prodotto?
* Che tipo di interfaccia si immagina?
* Che prestazioni minime deve fornire il prodotto?
* Che grado di sicurezza deve avere il prodotto?
* …
* Ma che bella giornata!

In base alla lista dei requisiti e all’analisi degli stessi, il progettista redige una *specifica dei requisiti* in cui elenca e descrive in modo dettagliato quali sono le funzionalità che il prodotto fornirà. La specifica dovrebbe essere abbastanza dettagliata da poter essere utilizzata come base per lo sviluppo, ma non troppo; ad esempio non dovrebbe contenere dettagli di implementazione, o definizioni dettagliate dell’interfaccia grafica a meno che questi non siano considerati cruciali. Non si deve scordare che i requisiti non rappresentano delle attività bensì delle caratteristiche che il prodotto dovrà possedere.

|  |  |
| --- | --- |
| **ID: REQ-001** | |
| **Nome** | Il software deve essere portable. |
| **Priorità** | 1 |
| **Versione** | 1.0 |
| **Note** | Deve essere possibile trasferire facilmente il software da una macchina all’altra e non deve lasciare files o impostazioni nel sistema in cui viene eseguito. |

|  |  |
| --- | --- |
| **ID: REQ-002** | |
| **Nome** | Il software deve essere multipiattaforma. |
| **Priorità** | 1 |
| **Versione** | 1.0 |
| **Note** | Deve essere possibile utilizzare il software su qualsiasi sistema che non sia mobile. |

|  |  |
| --- | --- |
| **ID: REQ-003** | |
| **Nome** | L’utente server deve poter scegliere la porta di ascolto del server. |
| **Priorità** | 2 |
| **Versione** | 1.0 |
| **Note** | La porta scelta dall’utente deve essere una di quelle disponibili. |
| **Sotto requisiti** | |
| **001** | Se l’utente non sceglie una porta il server ne propone una libera. |

|  |  |
| --- | --- |
| **ID: REQ-004** | |
| **Nome** | Una volta scelta la porta, il server deve mostrare il suo stato (se è pronto e in ascolto) e quali sono i parametri che permettono al client di connettersi. |
| **Priorità** | 2 |
| **Versione** | 1.0 |
| **Note** |  |

|  |  |
| --- | --- |
| **ID: REQ-005** | |
| **Nome** | Il software non avrà una GUI, verrà lanciato da terminale. |
| **Priorità** | 1 |
| **Versione** | 1.0 |
| **Note** |  |

|  |  |
| --- | --- |
| **ID: REQ-006** | |
| **Nome** | L’utente client deve poter scegliere il numero di connessioni da mandare al server. |
| **Priorità** | 2 |
| **Versione** | 1.0 |
| **Note** | Deve scegliere il numero di pacchetti di dati da mandare e se vanno mandati in sequenza o contemporaneamente. |

|  |  |
| --- | --- |
| **ID: REQ-007** | |
| **Nome** | L’utente client deve poter scegliere la dimensione dei dati per ogni connessione da scambiare con il server. |
| **Priorità** | 2 |
| **Versione** | 1.0 |
| **Note** |  |

|  |  |
| --- | --- |
| **ID: REQ-008** | |
| **Nome** | Il client deve produrre in output una statistica significativa dei dati scambiati con il server. |
| **Priorità** | 1 |
| **Versione** | 1.0 |
| **Note** | Per ogni prova viene creata una statistica contenente la media, il massimo e il minimo della velocità di risposta dal server. I tempi vengono misurati al millisecondo. |

|  |  |
| --- | --- |
| **ID: REQ-009** | |
| **Nome** | Il client deve essere multithread. |
| **Priorità** | 2 |
| **Versione** | 1.0 |
| **Note** | Il client può trasmettere al server più connessioni contemporaneamente. |

|  |  |
| --- | --- |
| **ID: REQ-010** | |
| **Nome** | Il server deve essere multithread. |
| **Priorità** | 2 |
| **Versione** | 1.0 |
| **Note** | Il server deve poter accettare più di una connessione in entrata contemporaneamente (sulla stessa porta o su porte diverse). |

## 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:

|  |
| --- |
| gantt  Figura 1: Esempio di diagramma di Gantt. |

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

## 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

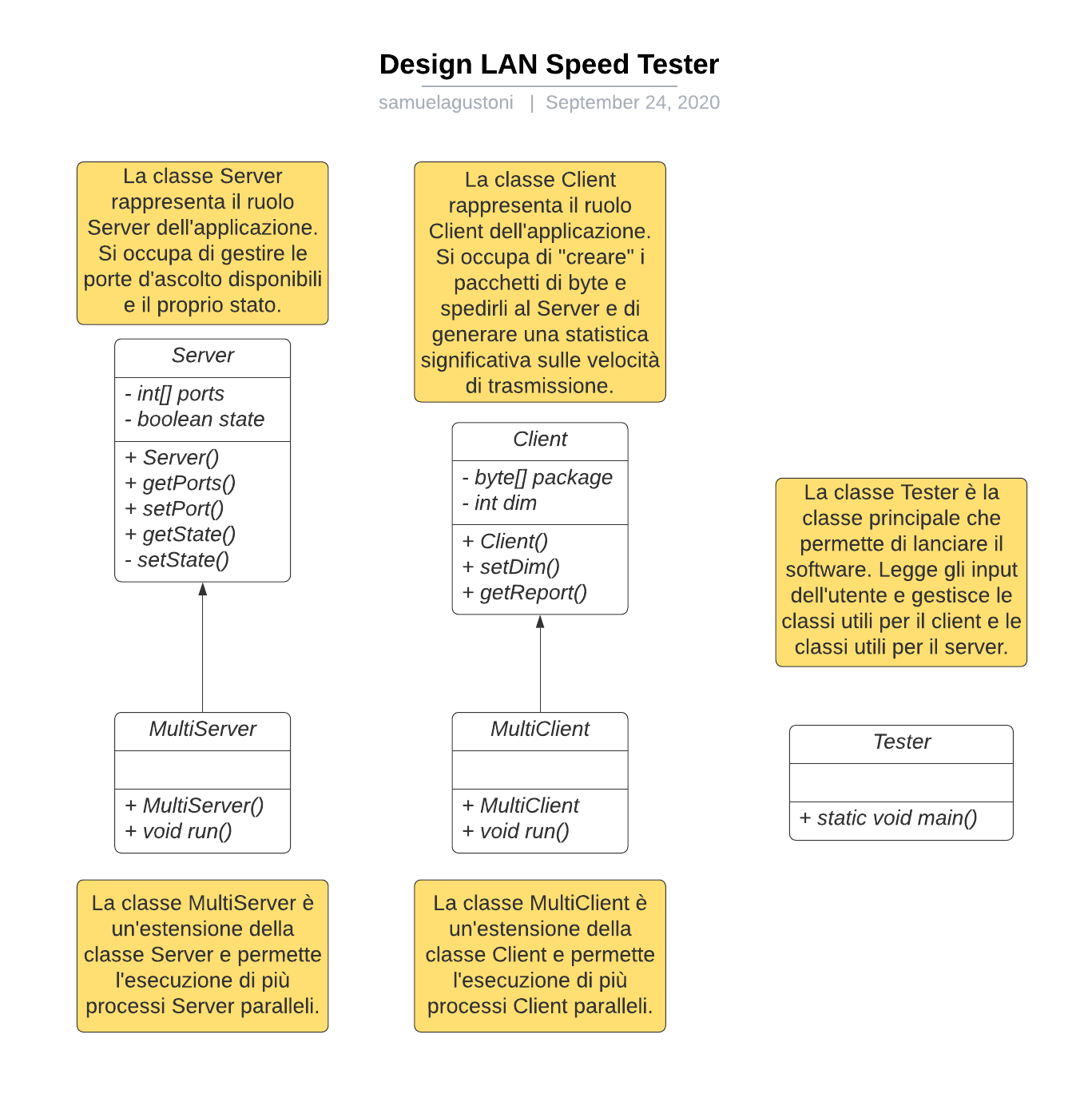
Su quale piattaforma dovrà essere eseguito il prodotto? Che hardware particolare è coinvolto nel progetto? Che particolarità e limitazioni presenta? Che hw sarà disponibile durante lo sviluppo?

# 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.

## 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 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
* …