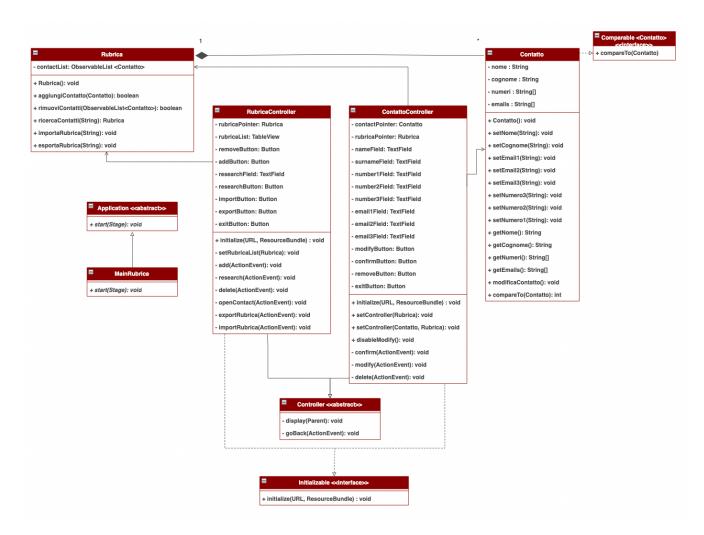
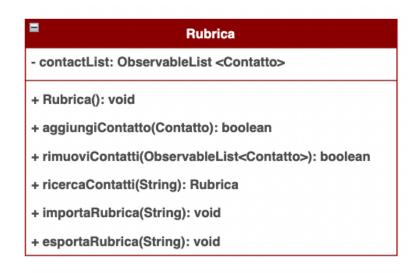
# Documento Design Diagramma delle classi



### Coesione delle classi



La classe Rubrica ha un livello di coesione funzionale, poiché si occupa di gestire un unico compito: le operazioni su una lista di contatti.

# Ξ

# Contatto

- nome : String

- cognome : String

- numeri : String[]

- emails : String[]

+ Contatto(): void

+ setNome(String): void

+ setCognome(String): void

+ setEmail1(String): void

+ setEmail2(String): void

+ setEmail3(String): void

+ setNumero3(String): void

+ setNumero2(String): void

+ setNumero1(String): void

+ getNome(): String

+ getCognome(): String

+ getNumeri(): String[]

+ getEmails(): String[]

+ modificaContatto(): void

+ compareTo(Contatto): int

La classe Contatto ha un livello di coesione funzionale poiché si focalizza sulla gestione dei dati del singolo contatto

# RubricaController

- rubricaPointer: Rubrica

- rubricaList: TableView

- removeButton: Button

- addButton: Button

- researchField: TextField

- researchButton: Button

- importButton: Button

- exportButton: Button

- exitButton: Button

- setRubricaList(Rubrica): void

- add(ActionEvent): void

- research(ActionEvent): void

- delete(ActionEvent): void

- openContact(ActionEvent): void

- export(ActionEvent): void

- import(ActionEvent): void

Il livello di coesione per RubricaController è comunicazionale poiché i metodi lavorano sugli stessi dati (oggetto di tipo rubrica).

## 

# ContattoController

- contactPointer: Contatto

- rubricaPointer: Rubrica

- nameField: TextField

- surnameField: TextField

- number1Field: TextField

- number2Field: TextField

- number3Field: TextField

- email1Field: TextField

- email2Field: TextField

- email3Field: TextField

- modifyButton: Button

- confirmButton: Button

- removeButton: Button

- exitButton: Button

+ initialize(URL, ResourceBundle): void

+ setController(Rubrica): void

+ setController(Contatto, Rubrica): void

+ disableModify(): void

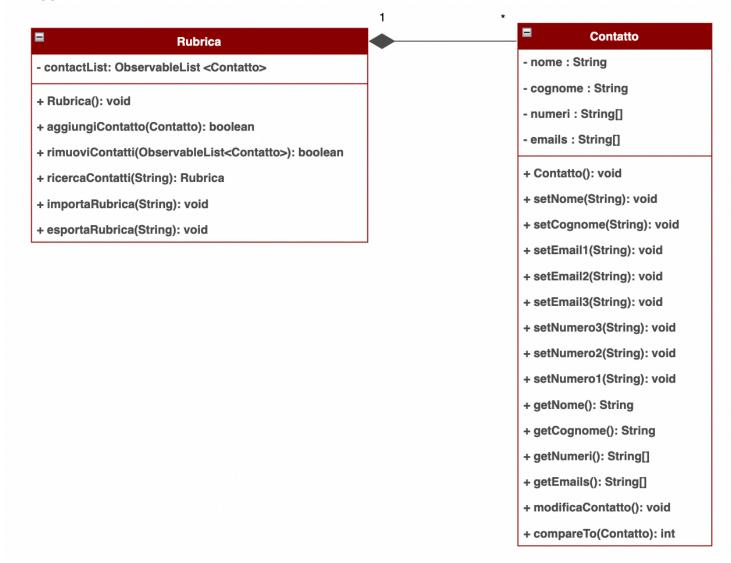
- confirm(ActionEvent): void

- modify(ActionEvent): void

- delete(ActionEvent): void

Il livello di coesione per ContattoController è comunicazionale perché include funzionalità che lavorano sugli stessi dati (singolo contatto).

## **Accoppiamento**



Accoppiamento sui dati perché la classe Rubrica interagisce con la classe contatto esclusivamente attraverso metodi pubblici richiamati singolarmente.

- contactList: ObservableList <Contatto>

+ Rubrica(): void
+ aggiungiContatto(Contatto): boolean
+ rimuoviContatti(ObservableList<Contatto>): boolean
+ ricercaContatti(String): Rubrica
+ importaRubrica(String): void
+ esportaRubrica(String): void

RubricaController - rubricaPointer: Rubrica - rubricaList: TableView - removeButton: Button - addButton: Button - researchField: TextField - researchButton: Button - importButton: Button - exportButton: Button - exitButton: Button + initialize(URL, ResourceBundle) : void - setRubricaList(Rubrica): void - add(ActionEvent): void - research(ActionEvent): void - delete(ActionEvent): void - openContact(ActionEvent): void - exportRubrica(ActionEvent): void - importRubrica(ActionEvent): void

Le interazioni tra la classe Rubrica e RubricaController avvengono rispettando il principio di accoppiamento sui dati: un modulo passa all'altro solo le informazioni strettamente necessarie per il suo funzionamento (RubricaController mantiene un riferimento ad un oggetto di tipo Rubrica, in quanto il suo compito è quello di gestirla).

Rubrica - contactList: ObservableList <Contatto> + Rubrica(): void + aggiungiContatto(Contatto): boolean + rimuoviContatti(ObservableList<Contatto>): boolean **ContattoController** + ricercaContatti(String): Rubrica - contactPointer: Contatto + importaRubrica(String): void - rubricaPointer: Rubrica + esportaRubrica(String): void - nameField: TextField - surnameField: TextField - number1Field: TextField - number2Field: TextField - number3Field: TextField - email1Field: TextField - email2Field: TextField - email3Field: TextField - modifyButton: Button - confirmButton: Button - removeButton: Button - exitButton: Button + initialize(URL, ResourceBundle) : void + setController(Rubrica): void + setController(Contatto, Rubrica): void + disableModify(): void - confirm(ActionEvent): void - modify(ActionEvent): void - delete(ActionEvent): void

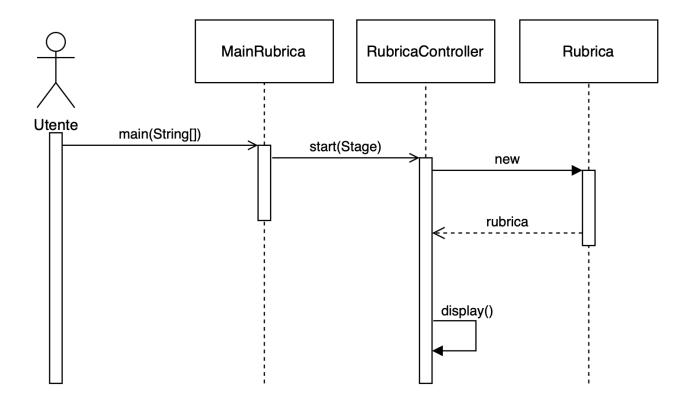
L'accoppiamento tra Rubrica e ContattoController è per timbro, in quanto il controller mantiene informazioni riguardo la totalità della rubrica, senza accedere però a tutte le sue funzioni (metodi di esportazione/importazione e di ricerca non vengono usati, ovvero sono informazioni extra che il controller possiede).

### Contatto ContattoController - contactPointer: Contatto - nome : String - rubricaPointer: Rubrica - cognome : String - nameField: TextField numeri : String[] surnameField: TextField - emails : String[] number1Field: TextField + Contatto(): void - number2Field: TextField + setNome(String): void - number3Field: TextField + setCognome(String): void - email1Field: TextField + setEmail1(String): void - email2Field: TextField + setEmail2(String): void - email3Field: TextField + setEmail3(String): void - modifyButton: Button + setNumero3(String): void - confirmButton: Button + setNumero2(String): void - removeButton: Button + setNumero1(String): void - exitButton: Button + getNome(): String + getCognome(): String + initialize(URL, ResourceBundle) : void + setController(Rubrica): void + getNumeri(): String[] + setController(Contatto, Rubrica): void + getEmails(): String[] + disableModify(): void + modificaContatto(): void - confirm(ActionEvent): void + compareTo(Contatto): int - modify(ActionEvent): void - delete(ActionEvent): void

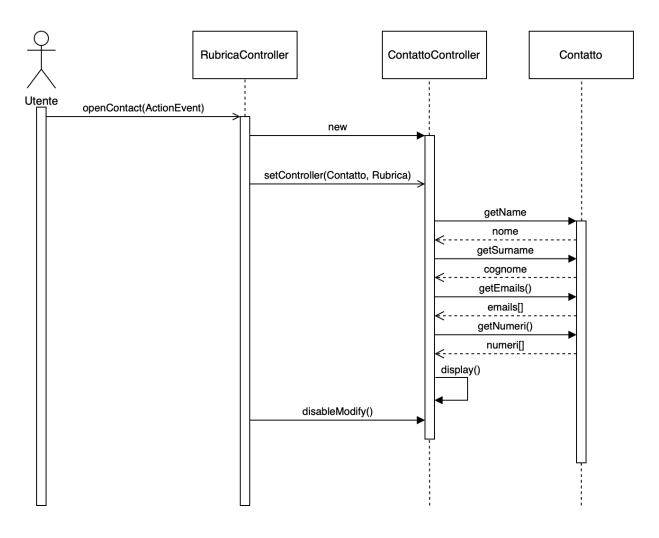
La classe Contatto si relaziona con la classe ContattoController unicamente tramite metodi pubblici che restituiscono singole informazioni: pertanto preserva un tipo di accoppiamento sui dati.

# Diagrammi di sequenza

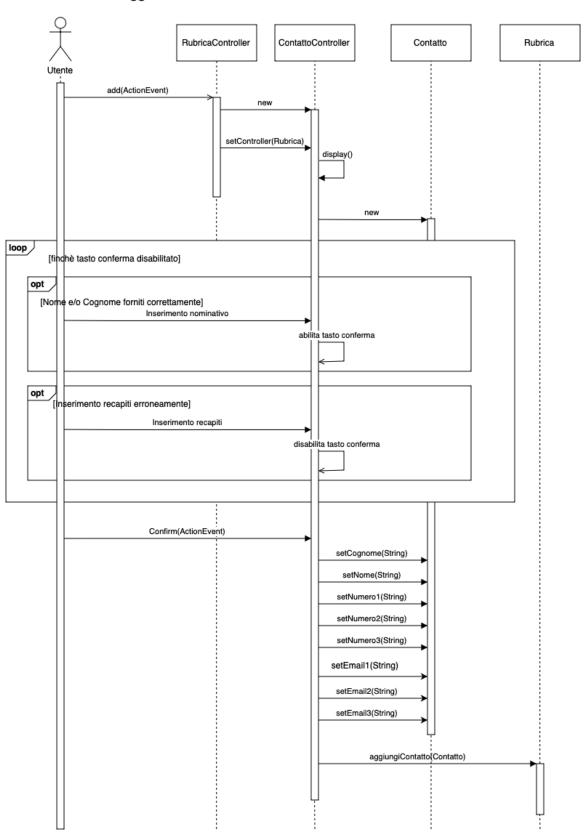
# VisualizzaRubrica



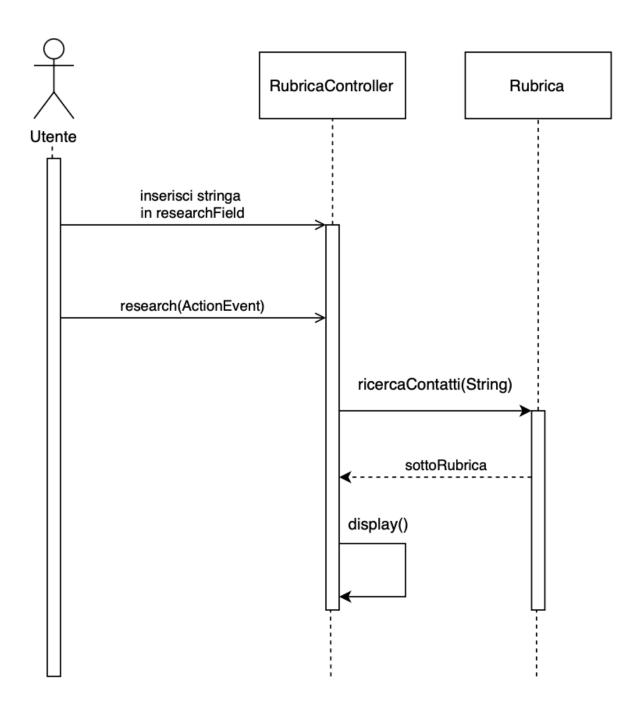
# AccessoContatto



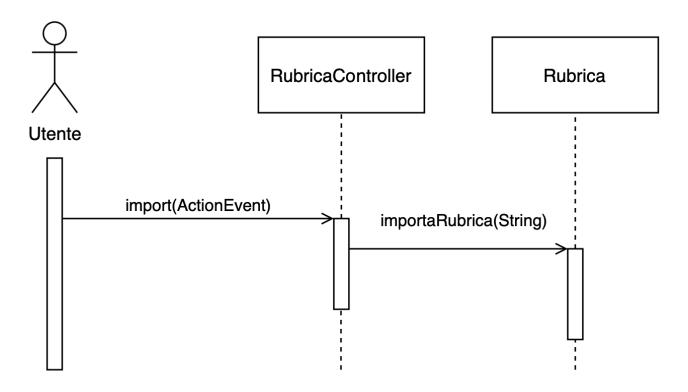
# AggiuntaContatto



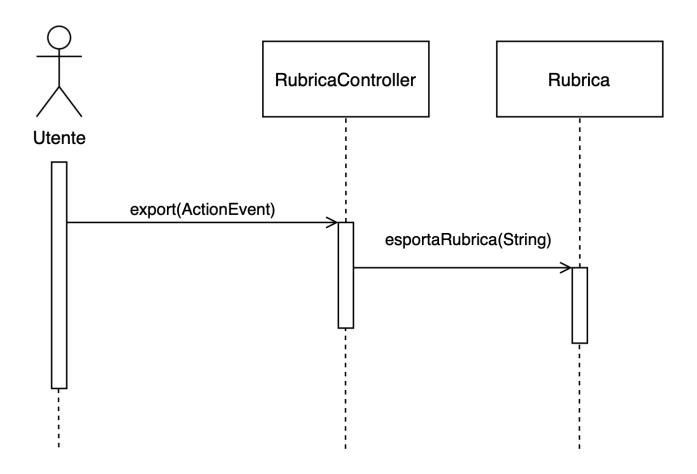
# RicercaContatto



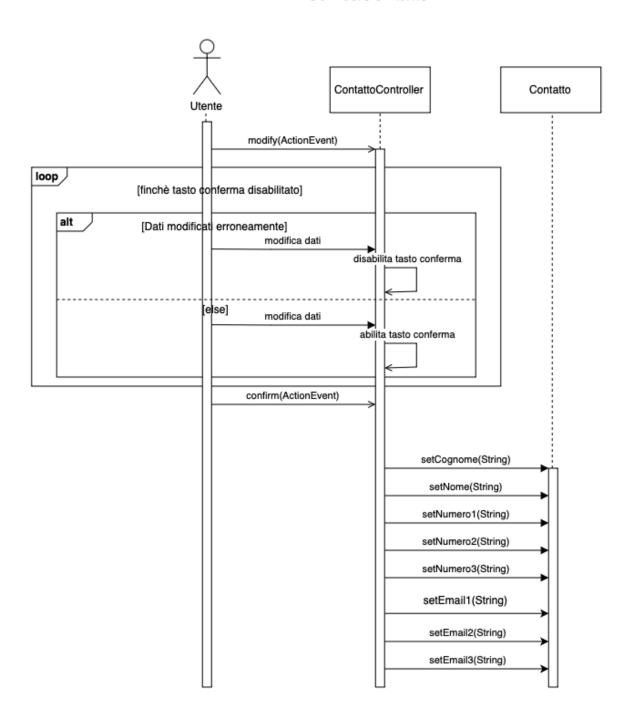
# ImportaRubrica



# EsportaRubrica



# ModificaContatto



# RubricaController Rubrica Rubrica ContattoController Rubrica Gelete(ActionEvent) rimuoviContatti(ObservableList<Contatto>)

# Diagramma dei package

