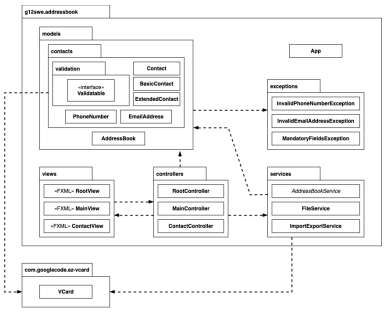

Il diagramma dei package descrive la struttura del progetto attraverso le macrofunzioni implementate. L'intero progetto risiede nel package "addressbook", con diversi sotto package per le varie funzioni. Si è scelto di assegnare ad un package "services" la gestione delle operazioni su file, al package "models" la gestione della struttura dell'elenco ed ai package "view" e "controller" rispettivamente la vista e la gestione dell'interfaccia utente.

AddressBook:
si occupa di contenere la collezione dei contatti (oggetti della classe contatto) e comunica con le classi FileService ed ImportExportService per fare in modo che file ed elenco siano coerenti con modifiche, aggiunte o eliminazioni di elementi della collezione.

Metodi pubblici:
addContact(Contact c):
il metodo aggiunge il contatto passato come parametro alla collezione.
invariante: l'istanza c di Contact
PreCondizione: c non deve essere null
PostCondizione: c è presente all'interno della collezione

removeContact(Contact c):
il metodo rimuove il contatto passato come parametro dalla collezione.
Invariante: l'istanza c di Contact
PreCondizione: c deve essere presente nella collezione
PostCondizione: la collezione non presenta più alcuna istanza del contatto c

getContactList():
restituisce una lista immutabile degli elementi della collezione
invariante: la collezione
PreCondizione:--
PostCondizione:--

Contact:
La classe contatto si occupa di mantenere le informazioni del contatto, con metodi di modifica e di gestione che non andranno ad intaccare la visualizzazione nella struttura dei contatti. Contiene collezioni di EmailAddress e PhoneNumber per associare istanze di queste due classi al contatto.

Metodi pubblici:
Contact(String name, String surname):
Inizializza un nuovo contatto con le informazioni inserite
Invariante: parametri name e surname
PreCondizione: La reference utilizzata non era già stata inizializzata
PostCondizione: i campi name e surname ora sono popolati con i parametri inseriti

getName():
restituisce il contenuto del campo name dell'istanza
invariante: l'istanza
PreCondizione: il campo name è stato popolato in precedenza
PostCondizione: la funzione ha restituito il valore di name

getSurname():
restituisce il contenuto del campo surname dell'istanza
Invariante: l'istanza
PreCondizione: il campo surname è stato popolato in precedenza
PostCondizione: la funzione ha restituito il valore di surname

addPhoneNumber(String phoneNumber):
aggiunge il numero di telefono contenuto in phoneNumber al contatto
invariante: phoneNumber
PreCondizione: la stringa contiene un numero di telefono valido
PostCondizione: al contatto è associato il numero di telefono

Il DesignByContract descrive il funzionamento delle classi utilizzate astraendo gli obblighi ed i servizi che la classe si impegna a rispettare ed erogare a chi si interfaccia ad essa.

Il diagramma di sequenza descrive il comportamento delle funzioni principali del progetto attraverso i messaggi scambiati tra gli oggetti delle classi interessate.

Il diagramma delle classi, generato automaticamente da EasyUML, descrive le varie classi utilizzate nel progetto. Mostra le informazioni relative alla visibilità degli attributi e dei metodi di ogni classe, esplicitando anche le relazioni tra classi.