Progetto Ingegneria del software

Fase 2 – Design: UML

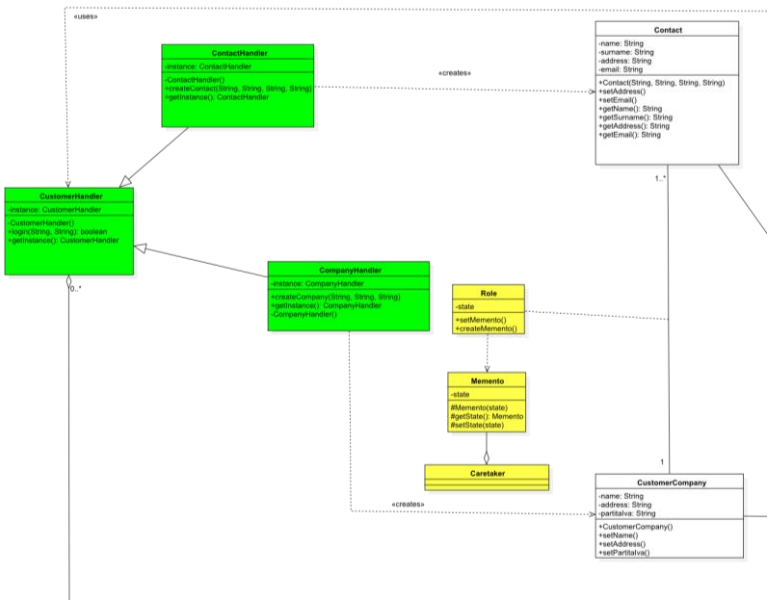
Introduzione

Per la descrizione del software CRM abbiamo realizzato i seguenti Diagrammi UML:

1. Class Diagram
2. 2 Activity Diagram
3. 2 Sequence Diagram
4. 2 Statechart Diagram
5. Object Diagram
6. Deployment Diagram
7. Component Diagram
8. Collaboration Diagram
9. UseCase Diagram

Class diagram

In prima istanza abbiamo lavorato sul *class diagram*. Prendendo in considerazione sia il diagramma *i-star* precedentemente realizzato e pensando al futuro codice Java che andremo ad implementare abbiamo costruito la struttura delle classi con i rispettivi attributi e metodi, impostando parallelamente le associazioni e le dipendenze tra di esse. Per quanto riguarda il particolare delle dipendenze tra i gestori e il dipendente che hanno <<uses>> come *stereotype*, vorremmo specificare che nella realtà non è il dipendente stesso a chiamarne i metodi, come si potrebbe evincere dal diagramma, bensì l'interfaccia grafica.



Un dettaglio del *class diagram.* Si noti il *factory pattern*, evidenziato in verde, e il *memento pattern,* evidenziato in giallo.

Inoltre abbiamo organizzato alcune classi utilizzando dei *design pattern*, poiché ritenevamo si adattassero alle situazioni che dobbiamo descrivere.

Per le classi che necessitano di essere istanziate una sola volta abbiamo ritenuto fossero adatte ad essere implementate attraverso il *singleton pattern*.

Il login abbiamo ritenuto fosse opportuno gestirlo attraverso un *chain of responsability* che permette di gestire i vari messaggi che il *log handler* restituisce.

Il *memento* è stato utilizzato ogni qualvolta avevamo necessità di memorizzare stati di istanze o istanze intere.

Infine abbiamo utilizzato il *factory pattern* in quelle situazioni in cui avevamo a che fare con la creazione di oggetti di sottoclassi di una stessa classe.

Component diagram

Il *component diagram* è stato sviluppato immaginando di dividere il nostro *class diagram* in componenti che raggruppassero più classi, e poi pensando di evidenziare come queste componenti interagissero, specificando le interfacce fornite e le richieste dalle componenti stesse.

Deployment diagram

Allo stesso modo abbiamo realizzato il *deployment*, solo che in questo caso abbiamo considerato le componenti fisiche che interagiscono con il sistema ed evidenziato il modo o i protocolli che permettono l'interazione tra esse.

Use case diagram

Lo *use case* è stato impostato in stretta relazione con il diagramma istar, quindi evidenziando gli obiettivi dei singoli attori e, le azioni che permettono il loro raggiungimento.

Object diagram

Riguardo all'*object diagram* si è trattato semplicemente di fare un esempio dell'istanza dipendente e di alcuni clienti collegati ad esso.

Activity diagram

Per gli *activity* abbiamo evidenziato quelle che ritenevamo essere due attività chiave nell'utilizzo del sistema: la creazione del profilo dipendente da parte del manager e la creazione del profilo cliente da parte del dipendente.

Statechart diagram

Gli *statechart* sono invece strutturati sugli stati che può assumere il cliente nel corso della gestione del suo profilo da parte del dipendente.

Sequence diagram

I *sequence* sono invece stati pensati allo scopo di evidenziare in uno la sequenza temporale delle azioni, nell'altro l'organizzazione.