

# KALEIDOSCODE

## SWEDesigner

### SOFTWARE PER DIAGRAMMI UML

SPECIFICA TECNICA V1.0.0



#### Informazioni sul documento

<b>Versione</b>	1.0.0
<b>Data Redazione</b>	01/05/2017
<b>Redazione</b>	Bonolo Marco Pace Giulio Pezzuto Francesco Sanna Giovanni
<b>Verifica</b>	Sovilla Matteo
<b>Approvazione</b>	Bonato Enrico
<b>Uso</b>	Esterno
<b>Distribuzione</b>	<i>Prof. Vardanega Tullio</i> <i>Prof. Cardin Riccardo</i> <i>Zucchetti s.p.a.</i>

---

## Diario delle Modifiche

Versione	Data	Autore	Descrizione
0.0.4	02/05/2017	Pezzuto Francesco	Sistemata Introduzione; Aggiunte sezioni Tecnologie Utilizzate, Architettura generale, Design pattern utilizzati
0.0.1	01/05/2017	Pace Giulio	Creazione scheletro del documento e stesura della sezione Introduzione

---

# Indice

<b>1</b>	<b>Introduzione</b>	<b>1</b>
1.1	Scopo del documento . . . . .	1
1.2	Scopo del prodotto . . . . .	1
1.3	Glossario . . . . .	1
1.4	Riferimenti utili . . . . .	1
1.4.1	Riferimenti normativi . . . . .	1
1.4.2	Riferimenti informativi . . . . .	1
<b>2</b>	<b>Tecnologie utilizzate</b>	<b>3</b>
2.1	HTML5 . . . . .	3
2.2	CSS . . . . .	3
2.3	Javascript . . . . .	3
2.4	JointJS . . . . .	4
2.5	jQuery . . . . .	4
2.6	Lodash . . . . .	5
2.7	Backbone.js . . . . .	5
2.8	Node.js . . . . .	6
2.9	JSON . . . . .	6
2.10	AJAX . . . . .	7
2.11	RequireJS . . . . .	7
2.12	MySQL . . . . .	7
<b>3</b>	<b>Architettura generale</b>	<b>9</b>
3.1	Architettura client . . . . .	9
3.1.1	Diagrammi editabili . . . . .	9
3.2	Architettura server . . . . .	10
3.2.1	Comunicazioni server-client . . . . .	10
<b>4</b>	<b>Componenti e classi principali</b>	<b>11</b>
4.1	SWEDesigner . . . . .	11
4.2	SWEDesigner::Client . . . . .	11
4.3	SWEDesigner::Client::Model . . . . .	11
4.3.1	SWEDesigner::Client::Model::Command . . . . .	11
4.3.2	SWEDesigner::Client::Model::ConcreteCommand . . . . .	11
4.3.3	SWEDesigner::Client::Model::State . . . . .	12
4.3.4	SWEDesigner::Client::Model::DAO . . . . .	12
4.3.5	SWEDesigner::Client::Model::MainModel . . . . .	12
4.3.6	SWEDesigner::Client::Model::TitleBarModel . . . . .	12
4.3.7	SWEDesigner::Client::Model::ToolBarModel . . . . .	12
4.3.8	SWEDesigner::Client::Model::PackageToolbar . . . . .	13
4.3.9	SWEDesigner::Client::Model::ClassToolbar . . . . .	13
4.3.10	SWEDesigner::Client::Model::ActivityToolbar . . . . .	13
4.3.11	SWEDesigner::Client::Model::BubbleToolbar . . . . .	13
4.3.12	SWEDesigner::Client::Model::AddressModel . . . . .	13
4.3.13	SWEDesigner::Client::Model::EditPanelModel . . . . .	13

4.3.14	SWEDesigner::Client::Model::ItemPanel . . . . .	13
4.3.15	SWEDesigner::Client::Model::DiagramTree . . . . .	14
4.3.16	SWEDesigner::Client::Model::Diagram . . . . .	14
4.3.17	SWEDesigner::Client::Model::PackageDiagram . . . . .	14
4.3.18	SWEDesigner::Client::Model::ClassDiagram . . . . .	14
4.3.19	SWEDesigner::Client::Model::ActivityDiagram . . . . .	14
4.3.20	SWEDesigner::Client::Model::BubbleDiagram . . . . .	14
4.4	SWEDesigner::Client::Model::RequestHandler . . . . .	15
4.4.1	SWEDesigner::Client::Model::RequestHandler::Sender . . . . .	15
4.4.2	SWEDesigner::Client::Model::RequestHandler::Receiver . . . . .	15
4.5	SWEDesigner::Client::View . . . . .	15
4.5.1	SWEDesigner::Client::View::MainView . . . . .	15
4.5.2	SWEDesigner::Client::View::TitleBarView . . . . .	15
4.5.3	SWEDesigner::Client::View::ToolBarView . . . . .	16
4.5.4	SWEDesigner::Client::View::AddressView . . . . .	16
4.5.5	SWEDesigner::Client::View::EditPanelView . . . . .	16
4.5.6	SWEDesigner::Client::View::Paper . . . . .	16
4.6	SWEDesigner::Server . . . . .	16
4.7	SWEDesigner::Server::CodeGenerator . . . . .	17
4.7.1	SWEDesigner::Server::CodeGenerator::CodeGenerator . . . . .	17
4.8	SWEDesigner::Server::CodeGenerator::Builder . . . . .	17
4.8.1	SWEDesigner::Server::CodeGenerator::Builder::Builder . . . . .	17
4.9	SWEDesigner::Server::CodeGenerator::Coder . . . . .	17
4.9.1	SWEDesigner::Server::CodeGenerator::Coder::JavaCoder . . . . .	17
4.9.2	SWEDesigner::Server::CodeGenerator::Coder::JavascriptCoder . . . . .	18
4.9.3	SWEDesigner::Server::CodeGenerator::Coder::CoderClass . . . . .	18
4.9.4	SWEDesigner::Server::CodeGenerator::Coder::CoderOperation . . . . .	18
4.9.5	SWEDesigner::Server::CodeGenerator::Coder::CodeParameter . . . . .	18
4.9.6	SWEDesigner::Server::CodeGenerator::Coder::CoderActivity . . . . .	18
4.9.7	SWEDesigner::Server::CodeGenerator::Coder::CodedProgram . . . . .	18
4.9.8	SWEDesigner::Server::CodeGenerator::Coder::Coder . . . . .	19
4.9.9	SWEDesigner::Server::CodeGenerator::Coder::CoderElement . . . . .	19
4.10	SWEDesigner::Server::CodeGenerator::Parser . . . . .	19
4.10.1	SWEDesigner::Server::CodeGenerator::Parser::Parser . . . . .	19
4.11	SWEDesigner::Server::CodeGenerator::Zipper . . . . .	19
4.11.1	SWEDesigner::Server::CodeGenerator::Zipper::Zipper . . . . .	19
4.12	SWEDesigner::Server::DAO . . . . .	20
4.13	SWEDesigner::Server::RequestHandler . . . . .	20
4.13.1	SWEDesigner::Server::RequestHandler::Sender . . . . .	20
4.13.2	SWEDesigner::Server::RequestHandler::Receiver . . . . .	20
<b>5</b>	<b>Tracciamento</b>	<b>21</b>
5.1	Tracciamento requisiti-componenti . . . . .	21

---

<b>A</b>	<b>Design pattern utilizzati</b>	<b>36</b>
A.1	Strutturali . . . . .	36
A.1.1	Facade . . . . .	36
A.2	Creazionali . . . . .	37
A.2.1	Singleton . . . . .	37
A.2.2	Factory . . . . .	38
A.3	Comportamentali . . . . .	39
A.3.1	Observer . . . . .	39
A.3.2	Command . . . . .	40

---

## Elenco delle tabelle

2	Tracciamento Requisiti-Use case . . . . .	35
---	---	----

## Elenco delle figure

1	Esempi delle possibili comunicazioni client-server . . . . .	10
2	Esempio pattern Facade . . . . .	36
3	Esempio pattern Singleton . . . . .	37
4	Esempio pattern Factory . . . . .	38
5	Esempio pattern Observer . . . . .	39
6	Esempio pattern Command . . . . .	40

# 1 Introduzione

## 1.1 Scopo del documento

Con il presente documento si intende definire la progettazione ad alto livello del progetto *SWEDesigner*.

Verrà presentata innanzi tutto l'architettura generale secondo la quale verranno organizzate le componenti software. Successivamente verranno descritti i Design pattern<sub>G</sub> utilizzati.

## 1.2 Scopo del prodotto

Lo scopo del progetto è la realizzazione di un software di costruzione di diagrammi UML<sub>G</sub> con la relativa generazione di codice Java<sub>G</sub> e Javascript<sub>G</sub> utilizzando tecnologie web. Il prodotto deve essere conforme ai vincoli qualitativi richiesti dal committente.

## 1.3 Glossario

Al fine di evitare ogni ambiguità di linguaggio e massimizzare la comprensione dei documenti i termini tecnici, di dominio, gli acronimi e le parole che necessitano di essere chiarite sono riportate nel documento *Glossario v2.0.0*.

La prima occorrenza di ciascuno di questi vocaboli è marcata da una "G" maiuscola in pedice.

## 1.4 Riferimenti utili

### 1.4.1 Riferimenti normativi

- **Capitolato<sub>G</sub> d'appalto:**  
<http://www.math.unipd.it/~tullio/IS-1/2016/Progetto/C6.pdf> (09/03/2017);
- **Norme di progetto:** *Norme di progetto v2.0.0*;
- **Analisi dei requisiti:** *Analisi dei requisiti v2.0.0*;
- **Verbali esterni:**
  - Verbale incontro con *Zucchetti s.p.a.* in data 05/05/2017.

### 1.4.2 Riferimenti informativi

- **Slide dell'insegnamento di Ingegneria del Software 1° semestre:**
  - Design pattern strutturali:  
<http://www.math.unipd.it/~tullio/IS-1/2016/Dispense/E04.pdf> (02/05/2017);
  - Design pattern creazionali:  
<http://www.math.unipd.it/~tullio/IS-1/2016/Dispense/E05.pdf> (02/05/2017);
  - Design pattern comportamentali:  
<http://www.math.unipd.it/~tullio/IS-1/2016/Dispense/E06.pdf> (02/05/2017);



- Design pattern architetturali:  
<http://www.math.unipd.it/~tullio/IS-1/2016/Dispense/E07.pdf> (02/05/2017),  
<http://www.math.unipd.it/~tullio/IS-1/2016/Dispense/E08.pdf> (02/05/2017);
- Stili architetturali:  
<http://www.math.unipd.it/~tullio/IS-1/2016/Dispense/E09.pdf> (02/05/2017);
- **Design Patterns: Elements of reusable object-oriented software**  
E. Gamma, R. Helm, R. Johnson, J. Vlissides - 1st Edition (2002)
  - Capitolo 3: Creational patterns;
  - Capitolo 4: Structural patterns;
  - Capitolo 5: Behavioral patterns.

## 2 Tecnologie utilizzate

### 2.1 HTML5

Linguaggio per la strutturazione delle pagine web, come richiesto dal Proponente.

#### Principali vantaggi

- Possibilità di gestire immagini (canvas) e audio direttamente attraverso Javascript;
- Linguaggio ben documentato e quindi di ridotta complessità;
- Il suo corretto utilizzo permette di separare struttura e contenuti delle pagine web, dalla loro presentazione e comportamento che vengono realizzate con altri linguaggi, aumentando quindi la manutenibilità del prodotto.

#### Principali svantaggi

- Linguaggio non ancora pienamente supportato da tutti i browser<sub>G</sub>.

### 2.2 CSS

Linguaggio per la formattazione e presentazione delle pagine HTML<sub>G</sub>, come richiesto dal Proponente.

#### Principali vantaggi

- Il suo corretto utilizzo permette di separare totalmente la presentazione delle pagine HTML;
- Diminuisce i tempi di sviluppo e restyling di un sito, aumentandone quindi la manutenibilità;
- Consente di produrre pagine più leggere, riducendo i tempi di attesa per gli utenti;
- Il suo corretto utilizzo consente di aumentare l'accessibilità di un sito a screen-reader<sub>G</sub>, browser testuali e dispositivi alternativi.

#### Principali svantaggi

- La versione 3 del linguaggio non è ancora pienamente supportata da tutti i browser.

### 2.3 Javascript

Linguaggio utilizzato per la realizzazione del comportamento delle pagine HTML, come richiesto dal Proponente.

### Principali vantaggi

- Il codice Javascript viene eseguito dal browser dell'utente; di conseguenza il server non è sfruttato più del dovuto;
- Rende dinamiche le pagine web, permettendo al sito di reagire ad eventi generati dall'interazione dell'utente;
- Rende possibile interagire con il DOM<sub>G</sub>.

### Principali svantaggi

- Il codice è visibile e può essere letto da chiunque.

## 2.4 JointJS

Libreria Javascript scelta per la creazione dell'editor dei diagrammi UML.  
(<https://www.jointjs.com/opensource> - 02/05/2017)

### Principali vantaggi

- Elementi grafici di diagrammi UML pronti all'uso;
- Elementi e collegamenti interattivi;
- Serializzazione/de-serializzazione da/a formato JSON;
- Supporto a Node.js<sub>G</sub>.

### Principali svantaggi

- La versione open-source utilizzata è dipendente da altre librerie (jQuery, Lodash, Backbone), rendendo le ulteriori scelte tecnologiche obbligate all'utilizzo di queste ultime.

## 2.5 jQuery

Libreria Javascript utilizzata da JointJS utile allo sviluppo di script<sub>G</sub> lato client.  
(<https://jquery.com/> - 02/05/2017)

### Principali vantaggi

- Facilita la manipolazione del DOM;
- Facilita lo sviluppo di comunicazioni asincrone tra client e server utilizzando AJAX;
- Facilita la realizzazione di animazioni a livello base;
- Buon supporto da parte della community.

### Principali svantaggi

- Non tutta la libreria è sviluppata rispettando uno standard comune, rendendo eventualmente necessario manipolarne il codice per i propri bisogni;
- Può rallentare un sito nel caso di manipolazioni multiple simultanee del DOM.

## 2.6 Lodash

Libreria Javascript utilizzata da JointJS utile per svolgere operazioni di base all'interno di script.

(<https://lodash.com/> - 02/05/2017)

### Principali vantaggi

- Fornisce molte funzionalità per cercare di supportare il maggior numero di ambienti e requisiti;
- Buon supporto da parte della community.

## 2.7 Backbone.js

Framework<sub>G</sub> Javascript utilizzato da JointJS utile per fornire una struttura ad applicazioni web rendendo disponibili modelli con binding chiave-valore ed eventi personalizzati, collezioni con una API<sub>G</sub> contenente funzioni enumerabili, viste con gestione degli eventi dichiarativa ed un'interfaccia JSON RESTful.

(<http://backbonejs.org/> - 02/05/2017)

### Principali vantaggi

- Compatto e versatile;
- Contiene solo le componenti base necessarie a strutturare una web app secondo il pattern MVC;
- Ha una buona documentazione, inoltre è presente una versione commentata del codice sorgente dove è quindi spiegato come lavora nel dettaglio;
- Supporta plugins<sub>G</sub> di terze parti.

### Principali svantaggi

- Non supporta il data binding<sub>G</sub> bidirezionale;
- È difficile eseguire test di unità sulle views scrivendo poco codice per mocking<sub>G</sub>.

## 2.8 Node.js

Runtime Javascript open-source scelto per sviluppare la parte server di *SWEDesigner* come richiesto dal Proponente (Requisito R0V1); utilizza un modello I/O non bloccante ad eventi asincroni ed è costruito sul motore Javascript v8; non è multi-threaded<sub>G</sub>, ma funziona in un singolo thread con il concetto di callback, inoltre esegue loop basato su eventi a singolo thread così da rendere non bloccanti tutte le esecuzioni.

(<https://nodejs.org/it/> - 02/05/2017)

### Principali vantaggi

- I/O ad eventi asincroni aiutano la gestione di richieste simultanee;
- Condivide la stessa porzione di codice con entrambi i lati client e server;
- Community molto attiva con molto codice condiviso via GitHub<sub>G</sub>, ecc.

### Principali svantaggi

- Rende complessa la gestione di database<sub>G</sub> relazionali.

## 2.9 JSON

JavaScript Object Notation, è il formato scelto per l'interscambio di dati tra client e server; è basato su Javascript, inoltre viene utilizzato in AJAX come alternativa a XML<sub>G</sub>. JSON è basato su due strutture:

- Un insieme di coppie nome/valore; in diversi linguaggi questo è realizzato come un oggetto, uno struct, una tabella hash, un array associativo, ecc.;
- Un elenco ordinato di valori; nella maggior parte dei linguaggi questo è realizzato con un array, un vettore, un elenco, ecc.;

### Principali vantaggi

- Leggero e supportato bene da tutti i browser;
- Formato conciso grazie al suo approccio basato su coppia nome/valore;
- Modo completamente automatizzato di serializzare/de-serializzare gli oggetti Javascript che richiede poco sviluppo di codice;
- API semplice;
- Supportato da molti toolkit di AJAX e librerie Javascript.

### Principali svantaggi

- Nessun supporto a namespace che porta ad una scarsa estensibilità;
- Nessun sostegno per la definizione della grammatica formale; di conseguenza i contratti di interfaccia sono difficili da comunicare e far rispettare;
- Supporta strumenti di sviluppo limitati.

## 2.10 AJAX

Asynchronous Javascript And Xml, è la tecnica scelta per lo sviluppo della comunicazione dei client verso il server.

### Principali vantaggi

- Migliora l'esperienza utente, “nascondendo” l'aggiornamento della pagina web;
- Riduce l'uso di banda e velocizza i caricamenti;
- È compatibile con molti linguaggi ed è supportato da molti browser.

### Principali svantaggi

- Può rendere difficile il debug della pagina web sulla quale è utilizzato poiché aumenta la dimensione del suo codice sorgente;
- Può incrementare il carico sul server web nel caso in cui si utilizzi per aggiornare troppo frequentemente una pagina.

## 2.11 RequireJS

Loader di moduli e file Javascript ottimizzato per l'uso in-browser ma anche per altri ambienti Javascript come Node.js.

(<http://requirejs.org/> - 02/05/2017)

### Principali vantaggi

- Buon supporto alla separazione del codice;
- Supporto a plugins;
- Può caricare moduli in modo asincrono “su richiesta”.

### Principali svantaggi

- Strumento difficile da usare efficacemente.

## 2.12 MySQL

DBMS<sub>G</sub> SQL relazionale scelta per lo sviluppo della base dati del sistema.

(<https://www.mysql.com/> - 02/05/2017)

### Principali vantaggi

- Tanto famoso quanto solido per sviluppare basi di dati;
- È progettato con in mente il web, il cloud e big data<sub>G</sub>;
- Supporta moli di dati che possono essere anche molto grandi ed è veloce.

### **Principali svantaggi**

- Non supporta alcune tipologie di join;
- Non supporta la possibilità di fare sub-query senza rieseguirle ogni volta.

## 3 Architettura generale

*SWEDesigner* è realizzato utilizzando un'architettura client-server, in particolare:

- Il **client** corrisponde alla parte dell'applicativo che funzionerà nel browser dell'utente;
- Il **server** avrà il compito di fornire la pagina dell'applicativo al client e ne gestirà le richieste ricevute riguardanti la generazione del codice sorgente o le attività "bubble" da inserire nell'editor.

### 3.1 Architettura client

Il client (parte front-end<sub>G</sub>) è una Single Page Application (SPA<sub>G</sub>) scritta con i linguaggi HTML5, CSS<sub>G</sub> e Javascript.

La sua architettura è costruita utilizzando il framework Backbone.js che offre un'architettura di tipo Model-View ed è quindi principalmente suddivisa nei seguenti moduli:

- **Model:** organizza la logica alla base dei diagrammi dell'editor.
- **View:** gestisce l'interfaccia grafica dell'editor e, seguendo la struttura definita da Backbone.js, "contiene" la componente controller per la gestione degli eventi;

#### 3.1.1 Diagrammi editabili

In ogni diagramma creabile all'interno dell'applicazione è offerto solamente un sottoinsieme del totale dei formalismi definiti dal linguaggio UML standard. Si possono individuare quattro tipi di diagrammi:

- Diagramma dei package<sub>G</sub>;
- Diagramma delle classi;
- Diagramma delle attività;
- Diagramma delle bubble.

Il diagramma dei package è logicamente correlato con il diagramma delle classi. Per ogni elemento (package o classe) all'interno di questi diagrammi è possibile assegnare un livello di importanza attraverso il quale si può "filtrare" gli oggetti a schermo visualizzabili nell'editor.

Per la corretta generazione del codice, nei diagrammi delle attività è previsto che l'utente approfondisca il loro livello di astrazione fino ad arrivare ad un diagramma costituito solamente da bubble (diagramma delle bubble) che verranno fornite nell'editor come se fossero delle attività specifiche.



## 3.2 Architettura server

Il server (parte back-end<sub>G</sub>) è sviluppato in Node.js ed offre i seguenti servizi:

- Fornire la Single Page Application ai client che la richiedono;
- Fornire la lista di bubble utilizzabili nell'editor;
- Generare il codice sorgente, nel formato desiderato dal client, del progetto inviatogli.

In particolare, la componente che genera il codice sorgente è stata realizzata utilizzando un'architettura di tipo Pipe And Filter, in modo tale da assegnare un compito ben preciso ad ogni modulo per attuare una procedura sequenziale a "catena di montaggio". L'ultimo modulo ha il compito di creare un file compresso .zip del codice generato che sarà poi inviato al client.

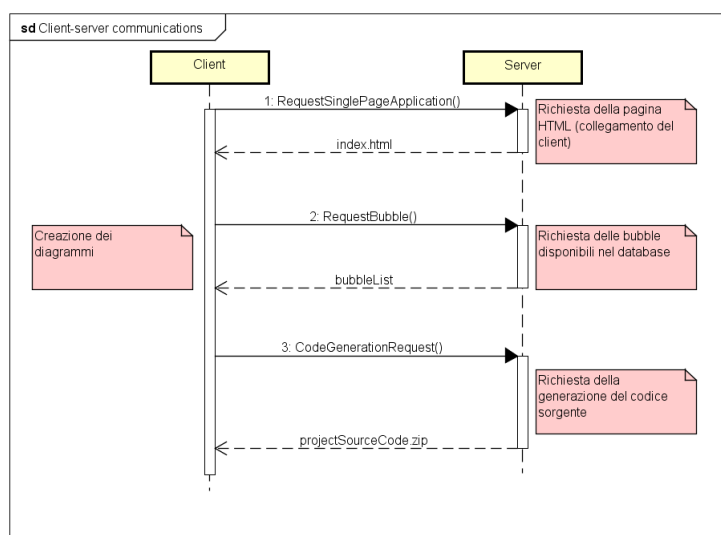
Le bubble saranno salvate in una base dati per poter garantire una futura estendibilità del numero di queste ultime, eventualmente anche in altri domini da quello considerato al momento (i giochi da tavolo).

### 3.2.1 Comunicazioni server-client

La Single Page Application viene fornita al client semplicemente attraverso una pagina HTML.

Per la richiesta e fornitura delle bubble, client e server utilizzano AJAX per lo scambio di dati in formato JSON in modo tale da alleggerire il traffico.

Per la richiesta della generazione del codice, il client invia i dati del progetto in formato JSON utilizzando AJAX ed il server una volta elaborata la richiesta procede con l'inviare il file .zip precedentemente descritto.



**Figura 1:** Esempi delle possibili comunicazioni client-server

## 4 Componenti e classi principali

### 4.1 SWEDesigner

I package contenuti al suo interno sono:

- SWEDesigner::Client;
- SWEDesigner::Server.

Questo package non contiene delle classi.

### 4.2 SWEDesigner::Client

I package contenuti al suo interno sono:

- SWEDesigner::Client::Model;
- SWEDesigner::Client::View.

Questo package non contiene delle classi.

### 4.3 SWEDesigner::Client::Model

I package contenuti al suo interno sono:

- SWEDesigner::Client::Model::RequestHandler.

Le classi contenute al suo interno verranno elencate qui di seguito.

#### 4.3.1 SWEDesigner::Client::Model::Command

È l'interfaccia che rappresenta un generico comando impartito dai moduli View ai Model.  
FAN-IN:

- ConcreteCommand;
- MainView;
- State;

Non ci sono dipendenze OUT.

#### 4.3.2 SWEDesigner::Client::Model::ConcreteCommand

Implementa l'interfaccia Command per la rappresentazione concreta dei singoli comandi impartiti dai moduli View ai Model.

Non ci sono dipendenze IN.

Non ci sono dipendenze OUT.

#### 4.3.3 SWEDesigner::Client::Model::State

Gestisce la cronologia delle operazioni svolte permettendo le operazioni di unDo e reDo.  
FAN-IN:

- MainView;

Non ci sono dipendenze OUT.

#### 4.3.4 SWEDesigner::Client::Model::DAO

Si occupa della persistenza dei dati, in particolare del salvataggio su file system locale del progetto già esistente.  
FAN-IN:

- MainView;

Non ci sono dipendenze OUT.

#### 4.3.5 SWEDesigner::Client::Model::MainModel

È il componente del programma che si occupa di gestire la parte logica dell'editor.  
FAN-IN:

- ConcreteCommand;
- DAO;
- MainView;

Non ci sono dipendenze OUT.

#### 4.3.6 SWEDesigner::Client::Model::TitleBarModel

Si occupa di gestire la parte logica della barra del titolo.  
FAN-IN:

- MainModel;

Non ci sono dipendenze OUT.

#### 4.3.7 SWEDesigner::Client::Model::ToolBarModel

È il componente del programma che si occupa di gestire la parte logica delle toolbar.  
FAN-IN:

- MainModel;

Non ci sono dipendenze OUT.

#### **4.3.8 SWEDesigner::Client::Model::PackageToolbar**

Rappresenta la particolare toolbar legata all'editor del diagramma dei package.

Non ci sono dipendenze IN.

Non ci sono dipendenze OUT.

#### **4.3.9 SWEDesigner::Client::Model::ClassToolbar**

Rappresenta la particolare toolbar legata all'editor del diagramma delle classi.

Non ci sono dipendenze IN.

Non ci sono dipendenze OUT.

#### **4.3.10 SWEDesigner::Client::Model::ActivityToolbar**

Rappresenta la particolare toolbar legata all'editor del diagramma delle attività.

Non ci sono dipendenze IN.

Non ci sono dipendenze OUT.

#### **4.3.11 SWEDesigner::Client::Model::BubbleToolbar**

Rappresenta la particolare toolbar legata all'editor del bubble flowchart.

Non ci sono dipendenze IN.

Non ci sono dipendenze OUT.

#### **4.3.12 SWEDesigner::Client::Model::AddressModel**

Si occupa di gestire la parte logica della barra degli indirizzi.

FAN-IN:

- MainModel;

Non ci sono dipendenze OUT.

#### **4.3.13 SWEDesigner::Client::Model::EditPanelModel**

Si occupa di gestire la parte logica del pannello di editing laterale.

FAN-IN:

- ItemPanel;
- MainModel;

Non ci sono dipendenze OUT.

#### **4.3.14 SWEDesigner::Client::Model::ItemPanel**

Estende la funzionalità di EditPanelModel specificamente per ciascun oggetto selezionato.

Non ci sono dipendenze IN.

Non ci sono dipendenze OUT.

#### **4.3.15 SWEDesigner::Client::Model::DiagramTree**

È la collezione di tutti i model associati a ciascun diagramma.

FAN-IN:

- MainModel;

Non ci sono dipendenze OUT.

#### **4.3.16 SWEDesigner::Client::Model::Diagram**

Si occupa di gestire la parte logica di un diagramma.

FAN-IN:

- ActivityDiagram;
- BubbleDiagram;
- ClassDiagram;
- DiagramTree;
- PackageDiagram;

Non ci sono dipendenze OUT.

#### **4.3.17 SWEDesigner::Client::Model::PackageDiagram**

Estende le funzionalità di Diagram per rappresentare un diagramma dei package.

Non ci sono dipendenze IN.

Non ci sono dipendenze OUT.

#### **4.3.18 SWEDesigner::Client::Model::ClassDiagram**

Estende le funzionalità di Diagram per rappresentare un diagramma delle classi.

Non ci sono dipendenze IN.

Non ci sono dipendenze OUT.

#### **4.3.19 SWEDesigner::Client::Model::ActivityDiagram**

Estende le funzionalità di Diagram per rappresentare un diagramma delle attività.

Non ci sono dipendenze IN.

Non ci sono dipendenze OUT.

#### **4.3.20 SWEDesigner::Client::Model::BubbleDiagram**

Estende le funzionalità di Diagram per rappresentare un bubble flowchart.

Non ci sono dipendenze IN.

Non ci sono dipendenze OUT.

## 4.4 SWEDesigner::Client::Model::RequestHandler

Questo package non contiene dei sottopackage.

Le classi contenute al suo interno verranno elencate qui di seguito.

### 4.4.1 SWEDesigner::Client::Model::RequestHandler::Sender

Si occupa di gestire le comunicazioni in uscita verso il server.

FAN-IN:

- MainModel;

Non ci sono dipendenze OUT.

### 4.4.2 SWEDesigner::Client::Model::RequestHandler::Receiver

Si occupa di gestire le comunicazioni in entrata dal server.

FAN-IN:

- Sender;

Non ci sono dipendenze OUT.

## 4.5 SWEDesigner::Client::View

Questo package non contiene dei sottopackage. Le classi contenute al suo interno verranno elencate qui di seguito.

### 4.5.1 SWEDesigner::Client::View::MainView

È il componente del programma che si occupa di gestire l'interfaccia grafica. Nella particolare declinazione MVC adottata da Backbone.js, si occupa anche di gestire gli input dell'utente e si interfaccia con il model attraverso dei command. È un aggregatore di altre classi View specializzate nella gestione dei diversi elementi dell'interfaccia grafica, in particolare TitleBarView, ToolBarView, AddressView, EditPanelView e Paper.

Non ci sono dipendenze IN.

Non ci sono dipendenze OUT.

### 4.5.2 SWEDesigner::Client::View::TitleBarView

È il componente del programma che fa la funzione di view per la barra del titolo, dove saranno collocati il menu dell'applicazione e gli shortcut.

FAN-IN:

- MainView;

Non ci sono dipendenze OUT.

#### 4.5.3 SWEDesigner::Client::View::ToolBarView

È il componente del programma che fa la funzione di view per la toolbar dove saranno collocati gli strumenti per editare i diagrammi.

FAN-IN:

- MainView;

Non ci sono dipendenze OUT.

#### 4.5.4 SWEDesigner::Client::View::AddressView

È il componente del programma che fa la funzione di view per il cosiddetto breadcrumb dove viene inserita la posizione corrente.

FAN-IN:

- MainView;

Non ci sono dipendenze OUT.

#### 4.5.5 SWEDesigner::Client::View::EditPanelView

È il componente del programma che fa la funzione di view per le informazioni editabili degli elementi che fanno parte dei diversi diagrammi.

FAN-IN:

- MainView;

Non ci sono dipendenze OUT.

#### 4.5.6 SWEDesigner::Client::View::Paper

È il componente del programma che fa la funzione di view per i diversi diagrammi.

FAN-IN:

- MainView;

Non ci sono dipendenze OUT.

### 4.6 SWEDesigner::Server

I package contenuti al suo interno sono:

- SWEDesigner::Server::CodeGenerator;
- SWEDesigner::Server::DAORequestHandler;
- SWEDesigner::Server::RequestHandler;

Questo package non contiene delle classi.

## 4.7 SWEDesigner::Server::CodeGenerator

I package contenuti al suo interno sono:

- SWEDesigner::Server::CodeGenerator::Builder;
- SWEDesigner::Server::CodeGenerator::Coder;
- SWEDesigner::Server::CodeGenerator::Parser;
- SWEDesigner::Server::CodeGenerator::Zipper;

Le classi contenute al suo interno verranno elencate qui di seguito.

### 4.7.1 SWEDesigner::Server::CodeGenerator::CodeGenerator

E' il componente che rende disponibile la funzionalità per cui, dato un file valido in formato JSON, restituisce un pacchetto in formato .zip contenente i file del codice sorgente che costituiscono il programma rappresentato dal file in input. I file prodotti sono strutturati in packages, come indicato nel file JSON in input.

FAN-IN:

- Sender;

Non ci sono dipendenze OUT.

## 4.8 SWEDesigner::Server::CodeGenerator::Builder

Questo package non contiene dei sottopackage.

Le classi contenute al suo interno verranno elencate qui di seguito.

### 4.8.1 SWEDesigner::Server::CodeGenerator::Builder::Builder

È il componente che rende disponibile la funzionalità, dato un file JSON in input che rappresenti un programma, di ottenere un oggetto contenitore del codice sorgente corrispondente al contenuto del file di input. Tale codice è suddiviso e strutturato come indicato nel file di input.

Non ci sono dipendenze IN.

Non ci sono dipendenze OUT.

## 4.9 SWEDesigner::Server::CodeGenerator::Coder

Questo package non contiene dei sottopackage.

Le classi contenute al suo interno verranno elencate qui di seguito.

### 4.9.1 SWEDesigner::Server::CodeGenerator::Coder::JavaCoder

È il componente che rende disponibile la funzionalità, dato un oggetto in input che rappresenta un file JSON parsificato, di ottenere un oggetto contenente il codice sorgente, in linguaggio Java, corrispondente all'oggetto in input.

Non ci sono dipendenze IN.

Non ci sono dipendenze OUT.



#### **4.9.2 SWEDesigner::Server::CodeGenerator::Coder::JavascriptCoder**

È il componente che rende disponibile la funzionalità, dato un oggetto in input che rappresenta un file JSON parsificato, di ottenere un oggetto contenente il codice sorgente, in linguaggio Javascript, corrispondente all'oggetto in input.

Non ci sono dipendenze IN.

Non ci sono dipendenze OUT.

#### **4.9.3 SWEDesigner::Server::CodeGenerator::Coder::CoderClass**

È il componente che mette a disposizione la funzionalità, data una stringa in input in formato JSON che rappresenta una classe valida, di ottenere il corrispondente codice sorgente di tale classe.

Non ci sono dipendenze IN.

Non ci sono dipendenze OUT.

#### **4.9.4 SWEDesigner::Server::CodeGenerator::Coder::CoderOperation**

È il componente che mette a disposizione la funzionalità, data una stringa in input in formato JSON che rappresenta un'operazione valida, di ottenere il corrispondente codice sorgente di tale operazione.

Non ci sono dipendenze IN.

Non ci sono dipendenze OUT.

#### **4.9.5 SWEDesigner::Server::CodeGenerator::Coder::CodeParameter**

È il componente che mette a disposizione la funzionalità, data una stringa in input in formato JSON che rappresenta un parametro di una lista valido, di ottenere il corrispondente codice sorgente di tale parametro. È possibile scegliere fra la codifica in Java o Javascript.

Non ci sono dipendenze IN.

Non ci sono dipendenze OUT.

#### **4.9.6 SWEDesigner::Server::CodeGenerator::Coder::CoderActivity**

È il componente che mette a disposizione la funzionalità, data una stringa in input in formato JSON che rappresenta un diagramma delle attività valido, di ottenere il corrispondente codice sorgente di tale attività. È possibile scegliere fra la codifica in Java o Javascript.

Non ci sono dipendenze IN.

Non ci sono dipendenze OUT.

#### **4.9.7 SWEDesigner::Server::CodeGenerator::Coder::CodedProgram**

È il componente che contiene il codice sorgente prodotto dal Coder.

Non ci sono dipendenze IN.

Non ci sono dipendenze OUT.

#### **4.9.8 SWEDesigner::Server::CodeGenerator::Coder::Coder**

Componente che funge da interfaccia alle operazioni di codifica di una stringa, in formato JSON che rappresenta un programma valido; tali operazioni permettono di ottenere un oggetto contenente il codice sorgente, in Java o Javascript, corrispondente alla stringa in input.

Non ci sono dipendenze IN.

Non ci sono dipendenze OUT.

#### **4.9.9 SWEDesigner::Server::CodeGenerator::Coder::CoderElement**

Componente astratta che offre la funzionalità di ottenere, data una stringa in input in formato JSON che rappresenta un elemento di classe valido, il corrispondente codice sorgente, in Java o Javascript.

FAN-IN:

- Coder;

Non ci sono dipendenze OUT.

#### **4.10 SWEDesigner::Server::CodeGenerator::Parser**

Questo package non contiene dei sottopackage. Le classi contenute al suo interno verranno elencate qui di seguito.

##### **4.10.1 SWEDesigner::Server::CodeGenerator::Parser::Parser**

È il componente che rende disponibile la funzionalità, dato un file JSON valido in input, di ottenere un oggetto contenente le informazioni che costituiscono il file in input.

FAN-IN:

- CodeGenerator;

Non ci sono dipendenze OUT.

#### **4.11 SWEDesigner::Server::CodeGenerator::Zipper**

Questo package non contiene dei sottopackage.

Le classi contenute al suo interno verranno elencate qui di seguito.

##### **4.11.1 SWEDesigner::Server::CodeGenerator::Zipper::Zipper**

E' il componente che rende disponibile la funzionalità per cui, dato un file valido in formato JSON, restituisce un pacchetto in formato .zip contenente i file del codice sorgente che costituiscono il programma rappresentato dal file in input. I file prodotti sono strutturati in packages, come indicato nel file JSON in input.

Non ci sono dipendenze IN.

Non ci sono dipendenze OUT.

## 4.12 SWEDesigner::Server::DAO

Questo package non contiene dei sottopackage.

Questo package non contiene delle classi.

## 4.13 SWEDesigner::Server::RequestHandler

Questo package non contiene dei sottopackage.

Le classi contenute al suo interno verranno elencate qui di seguito.

### 4.13.1 SWEDesigner::Server::RequestHandler::Sender

Si occupa di gestire le comunicazioni in uscita verso il client.

FAN-IN:

- Zipper;

Non ci sono dipendenze OUT.

### 4.13.2 SWEDesigner::Server::RequestHandler::Receiver

Si occupa di gestire le comunicazioni in entrata dal client.

FAN-IN:

- Sender;

Non ci sono dipendenze OUT.

## 5 Tracciamento

Segue la tabella di tracciamento tra requisiti e componenti. Per facilità di lettura nel caso di sottocomponenti fortemente legati al componente di livello superiore si è scelto di riportare solo quest'ultimo.

### 5.1 Tracciamento requisiti-componenti

Codice Requisiti	Codice Use case
R0F1	SWEDesigner::Client::View SWEDesigner::Client::Model
R0F2	SWEDesigner::Client::View SWEDesigner::Client::Model::TitleBarModel SWEDesigner::Client::Model::ClassToolbar SWEDesigner::Client::Model::EditPanelModel SWEDesigner::Client::Model::ClassDiagram
R0F3	SWEDesigner::Client::View SWEDesigner::Client::Model::TitleBarModel SWEDesigner::Client::Model::ActivityToolbar SWEDesigner::Client::Model::ActivityDiagram
R0F4	SWEDesigner::Client::View SWEDesigner::Client::Model::TitleBarModel SWEDesigner::Client::Model::ClassToolbar SWEDesigner::Client::Model::EditPanelModel SWEDesigner::Client::Model::ClassDiagram
R0F4.1	SWEDesigner::Client::View SWEDesigner::Client::Model::TitleBarModel SWEDesigner::Client::Model::ClassToolbar SWEDesigner::Client::Model::EditPanelModel SWEDesigner::Client::Model::ClassDiagram
R0F5	SWEDesigner::Server::CodeGenerator::CoderGenerator
R0F6	SWEDesigner::Server::CodeGenerator::CoderGenerator
R0F7	SWEDesigner::Server::CodeGenerator::CoderGenerator
R1F8	SWEDesigner::Client::View SWEDesigner::Client::Model::TitleBarModel SWEDesigner::Client::Model::BubbleToolbar SWEDesigner::Client::Model::BubbleDiagram
R0F11	SWEDesigner::Client::View SWEDesigner::Client::Model::TitleBarModel

Codice Requisiti	Codice Use case
	SWEDesigner::Client::Model::PackageToolbar SWEDesigner::Client::Model::PackageDiagram
R0F14 hline R0F14.1	SWEDesigner::Client::Model SWEDesigner::Client::Model::DAO SWEDesigner::Client::Model SWEDesigner::Client::Model::DAO
R0F14.1.1	SWEDesigner::Client::Model SWEDesigner::Client::Model::DAO
R0F14.2	SWEDesigner::Client::Model SWEDesigner::Client::Model::DAO
R0F14.3	SWEDesigner::Client::Model SWEDesigner::Client::Model::DAO
R0F14.3.1	SWEDesigner::Client::Model SWEDesigner::Client::Model::DAO
R0F14.4	SWEDesigner::Client::Model SWEDesigner::Client::Model::State
R0F14.5	SWEDesigner::Client::Model SWEDesigner::Client::Model::State
R0F14.6	SWEDesigner::Client::Model::RequestHandler SWEDesigner::Server::RequestHandler SWEDesigner::Server::CodeGenerator::CoderGenerator
R0F14.7	SWEDesigner::Client::Model::RequestHandler SWEDesigner::Server::RequestHandler SWEDesigner::Server::CodeGenerator::CoderGenerator
R0F14.8	SWEDesigner::Client::Model SWEDesigner::Client::Model::DAO
R0F14.8.1	SWEDesigner::Client::Model SWEDesigner::Client::Model::DAO
R0F14.8.2	SWEDesigner::Client::Model SWEDesigner::Client::Model::DAO
R0F15	SWEDesigner::Client::View SWEDesigner::Client::Model::TitleBarModel SWEDesigner::Client::Model::ClassToolbar SWEDesigner::Client::Model::EditPanelModel SWEDesigner::Client::Model::ClassDiagram
R0F15.1	SWEDesigner::Client::View SWEDesigner::Client::Model::TitleBarModel

Codice Requisiti	Codice Use case
	SWEDesigner::Client::Model::ClassToolbar SWEDesigner::Client::Model::ClassDiagram
R0F15.2	SWEDesigner::Client::View SWEDesigner::Client::Model::TitleBarModel SWEDesigner::Client::Model::EditPanelModel SWEDesigner::Client::Model::ClassDiagram
R0F15.2.1	SWEDesigner::Client::View SWEDesigner::Client::Model::TitleBarModel SWEDesigner::Client::Model::EditPanelModel SWEDesigner::Client::Model::ClassDiagram
R0F15.2.2	SWEDesigner::Client::View SWEDesigner::Client::Model::TitleBarModel SWEDesigner::Client::Model::ClassToolbar SWEDesigner::Client::Model::ClassDiagram
R0F15.2.3	SWEDesigner::Client::View SWEDesigner::Client::Model::TitleBarModel SWEDesigner::Client::Model::EditPanelModel SWEDesigner::Client::Model::ClassDiagram
R0F15.2.4	SWEDesigner::Client::View SWEDesigner::Client::Model::TitleBarModel SWEDesigner::Client::Model::EditPanelModel SWEDesigner::Client::Model::ClassDiagram
R0F15.2.4.1	SWEDesigner::Client::View SWEDesigner::Client::Model::TitleBarModel SWEDesigner::Client::Model::EditPanelModel SWEDesigner::Client::Model::ClassDiagram
R0F15.2.4.2	SWEDesigner::Client::View SWEDesigner::Client::Model::TitleBarModel SWEDesigner::Client::Model::EditPanelModel SWEDesigner::Client::Model::ClassDiagram
R0F15.2.4.3	SWEDesigner::Client::View SWEDesigner::Client::Model::TitleBarModel SWEDesigner::Client::Model::EditPanelModel SWEDesigner::Client::Model::ClassDiagram
R0F15.2.5	SWEDesigner::Client::View SWEDesigner::Client::Model::AddressModel SWEDesigner::Client::Model::ClassDiagram

Codice Requisiti	Codice Use case
R0F15.3	SWEDesigner::Client::View SWEDesigner::Client::Model::ClassDiagram
R0F15.4	SWEDesigner::Client::View SWEDesigner::Client::Model::ClassToolbar SWEDesigner::Client::Model::ClassDiagram
R0F15.4.1	SWEDesigner::Client::View SWEDesigner::Client::Model::ClassToolbar SWEDesigner::Client::Model::ClassDiagram
R0F15.4.2	SWEDesigner::Client::View SWEDesigner::Client::Model::ClassToolbar SWEDesigner::Client::Model::ClassDiagram
R0F15.4.3	SWEDesigner::Client::View SWEDesigner::Client::Model::ClassToolbar SWEDesigner::Client::Model::ClassDiagram
R0F15.4.4	SWEDesigner::Client::View SWEDesigner::Client::Model::ClassToolbar SWEDesigner::Client::Model::ClassDiagram
R0F15.4.5	SWEDesigner::Client::View SWEDesigner::Client::Model::ClassToolbar SWEDesigner::Client::Model::ClassDiagram
R0F15.4.6	SWEDesigner::Client::View SWEDesigner::Client::Model::ClassToolbar SWEDesigner::Client::Model::ClassDiagram
R0F15.4.7	SWEDesigner::Client::View SWEDesigner::Client::Model::ClassToolbar SWEDesigner::Client::Model::ClassDiagram
R0F15.5	SWEDesigner::Client::View SWEDesigner::Client::Model::EditPanelModel SWEDesigner::Client::Model::ClassDiagram
R0F15.5.1	SWEDesigner::Client::View SWEDesigner::Client::Model::EditPanelModel SWEDesigner::Client::Model::ClassDiagram
R0F15.5.2	SWEDesigner::Client::View SWEDesigner::Client::Model::EditPanelModel SWEDesigner::Client::Model::ClassDiagram
R0F15.5.3	SWEDesigner::Client::View SWEDesigner::Client::Model::EditPanelModel

Codice Requisiti	Codice Use case
	SWEDesigner::Client::Model::ClassDiagram
R0F15.5.4	SWEDesigner::Client::View SWEDesigner::Client::Model::EditPanelModel SWEDesigner::Client::Model::ClassDiagram
R0F15.5.5	SWEDesigner::Client::View SWEDesigner::Client::Model::EditPanelModel SWEDesigner::Client::Model::ClassDiagram
R0F15.5.6	SWEDesigner::Client::View SWEDesigner::Client::Model::EditPanelModel SWEDesigner::Client::Model::ClassDiagram
R0F15.5.7	SWEDesigner::Client::View SWEDesigner::Client::Model::EditPanelModel SWEDesigner::Client::Model::ClassDiagram
R0F15.6	SWEDesigner::Client::View SWEDesigner::Client::Model::ClassDiagram
R0F15.7	SWEDesigner::Client::View SWEDesigner::Client::Model::TitleBarModel CT SWEDesigner::Client::Model::ClassDiagram
R0F15.8	SWEDesigner::Client::View SWEDesigner::Client::Model::TitleBarModel SWEDesigner::Client::Model::EditPanelModel SWEDesigner::Client::Model::ClassDiagram
R0F15.8.1	SWEDesigner::Client::View SWEDesigner::Client::Model::TitleBarModel SWEDesigner::Client::Model::EditPanelModel SWEDesigner::Client::Model::ClassDiagram
R0F15.8.2	SWEDesigner::Client::View SWEDesigner::Client::Model::TitleBarModel SWEDesigner::Client::Model::EditPanelModel SWEDesigner::Client::Model::ClassDiagram
R0F15.8.2.1	SWEDesigner::Client::View SWEDesigner::Client::Model::TitleBarModel SWEDesigner::Client::Model::EditPanelModel SWEDesigner::Client::Model::ClassDiagram
R0F15.8.2.2	SWEDesigner::Client::View SWEDesigner::Client::Model::TitleBarModel



Codice Requisiti	Codice Use case
	SWEDesigner::Client::Model::EditPanelModel SWEDesigner::Client::Model::ClassDiagram
R0F15.8.2.3	SWEDesigner::Client::View SWEDesigner::Client::Model::TitleBarModel SWEDesigner::Client::Model::EditPanelModel SWEDesigner::Client::Model::ClassDiagram
R0F15.8.3	SWEDesigner::Client::View SWEDesigner::Client::Model::EditPanelModel SWEDesigner::Client::Model::ClassDiagram
R0F15.9	SWEDesigner::Client::View SWEDesigner::Client::Model::ClassDiagram
R0F15.10	SWEDesigner::Client::View CT SWEDesigner::Client::Model::EditPanelModel SWEDesigner::Client::Model::ClassDiagram
R0F15.11	SWEDesigner::Client::View SWEDesigner::Client::Model::ClassDiagram
R0F15.12	SWEDesigner::Client::View SWEDesigner::Client::Model::ClassDiagram
R0F15.13	SWEDesigner::Client::View SWEDesigner::Client::Model::ClassDiagram
R0F15.14	SWEDesigner::Client::View SWEDesigner::Client::Model::ClassDiagram
R0F15.14.1	SWEDesigner::Client::View SWEDesigner::Client::Model::ClassDiagram
R0F15.14.2	SWEDesigner::Client::View SWEDesigner::Client::Model::ClassDiagram
R0F15.15	SWEDesigner::Client::View SWEDesigner::Client::Model::AddressModel SWEDesigner::Client::Model::PackageDiagram SWEDesigner::Client::Model::ClassDiagram
R0F15.16	SWEDesigner::Client::View SWEDesigner::Client::Model::ClassDiagram
R0F16	SWEDesigner::Client::View SWEDesigner::Client::Model::EditPanelModel SWEDesigner::Client::Model::ClassDiagram
R0F16.1	SWEDesigner::Client::View

Codice Requisiti	Codice Use case
	SWEDesigner::Client::Model::EditPanelModel SWEDesigner::Client::Model::ClassDiagram
R0F16.2	SWEDesigner::Client::View SWEDesigner::Client::Model::EditPanelModel SWEDesigner::Client::Model::ClassDiagram
R0F16.2.1	SWEDesigner::Client::View SWEDesigner::Client::Model::EditPanelModel SWEDesigner::Client::Model::ClassDiagram
R0F16.2.2	SWEDesigner::Client::View SWEDesigner::Client::Model::EditPanelModel SWEDesigner::Client::Model::ClassDiagram
R0F16.2.3	SWEDesigner::Client::View SWEDesigner::Client::Model::EditPanelModel SWEDesigner::Client::Model::ClassDiagram
R0F16.2.4	SWEDesigner::Client::View SWEDesigner::Client::Model::EditPanelModel SWEDesigner::Client::Model::ClassDiagram
R0F16.3	SWEDesigner::Client::View SWEDesigner::Client::Model::EditPanelModel SWEDesigner::Client::Model::ClassDiagram
R0F16.4	SWEDesigner::Client::View SWEDesigner::Client::Model::EditPanelModel SWEDesigner::Client::Model::ClassDiagram
R0F16.5	SWEDesigner::Client::View SWEDesigner::Client::Model::EditPanelModel SWEDesigner::Client::Model::ClassDiagram
R0F16.5.1	SWEDesigner::Client::View SWEDesigner::Client::Model::EditPanelModel SWEDesigner::Client::Model::ClassDiagram
R0F16.5.2	SWEDesigner::Client::View SWEDesigner::Client::Model::EditPanelModel SWEDesigner::Client::Model::ClassDiagram
R0F16.5.3	SWEDesigner::Client::View SWEDesigner::Client::Model::EditPanelModel SWEDesigner::Client::Model::ClassDiagram
R0F16.5.3.1	SWEDesigner::Client::View SWEDesigner::Client::Model::EditPanelModel

Codice Requisiti	Codice Use case
	SWEDesigner::Client::Model::ClassDiagram
R0F16.5.3.2	SWEDesigner::Client::View SWEDesigner::Client::Model::EditPanelModel SWEDesigner::Client::Model::ClassDiagram
R0F16.5.3.2.1	SWEDesigner::Client::View SWEDesigner::Client::Model::EditPanelModel SWEDesigner::Client::Model::ClassDiagram
R0F16.5.3.2.2	SWEDesigner::Client::View SWEDesigner::Client::Model::EditPanelModel SWEDesigner::Client::Model::ClassDiagram
R0F16.5.3.2.3	SWEDesigner::Client::View SWEDesigner::Client::Model::EditPanelModel SWEDesigner::Client::Model::ClassDiagram
R0F16.5.3.2.4	SWEDesigner::Client::View SWEDesigner::Client::Model::EditPanelModel SWEDesigner::Client::Model::ClassDiagram
R0F16.5.3.3	SWEDesigner::Client::View SWEDesigner::Client::Model::EditPanelModel SWEDesigner::Client::Model::ClassDiagram
R0F16.5.4	SWEDesigner::Client::View SWEDesigner::Client::Model::EditPanelModel SWEDesigner::Client::Model::ClassDiagram
R0F16.6	SWEDesigner::Client::View SWEDesigner::Client::Model::EditPanelModel SWEDesigner::Client::Model::ClassDiagram
R0F16.7	SWEDesigner::Client::View SWEDesigner::Client::Model::EditPanelModel SWEDesigner::Client::Model::ClassDiagram
R0F16.8	SWEDesigner::Client::View SWEDesigner::Client::Model::EditPanelModel SWEDesigner::Client::Model::ClassDiagram
R0F16.9	SWEDesigner::Client::View SWEDesigner::Client::Model::ClassDiagram
R0F16.10	SWEDesigner::Client::View SWEDesigner::Client::Model::EditPanelModel SWEDesigner::Client::Model::ClassDiagram
R0F16.10.1	SWEDesigner::Client::View

Codice Requisiti	Codice Use case
	SWEDesigner::Client::Model::EditPanelModel SWEDesigner::Client::Model::ClassDiagram
R0F16.10.2	SWEDesigner::Client::View SWEDesigner::Client::Model::EditPanelModel SWEDesigner::Client::Model::ClassDiagram
R0F16.10.3	SWEDesigner::Client::View SWEDesigner::Client::Model::EditPanelModel SWEDesigner::Client::Model::ClassDiagram
R0F16.10.4	SWEDesigner::Client::View SWEDesigner::Client::Model::EditPanelModel SWEDesigner::Client::Model::ClassDiagram
R0F16.10.5	SWEDesigner::Client::View SWEDesigner::Client::Model::EditPanelModel SWEDesigner::Client::Model::ClassDiagram
R0F16.10.6	SWEDesigner::Client::View SWEDesigner::Client::Model::EditPanelModel SWEDesigner::Client::Model::ClassDiagram
R0F16.10.7	SWEDesigner::Client::View SWEDesigner::Client::Model::EditPanelModel SWEDesigner::Client::Model::ClassDiagram
R0F17	SWEDesigner::Client::View SWEDesigner::Client::Model::EditPanelModel SWEDesigner::Client::Model::ClassDiagram
R0F17.1	SWEDesigner::Client::View SWEDesigner::Client::Model::EditPanelModel SWEDesigner::Client::Model::ClassDiagram
R0F17.2	SWEDesigner::Client::View SWEDesigner::Client::Model::EditPanelModel SWEDesigner::Client::Model::ClassDiagram
R0F17.2.1	SWEDesigner::Client::View SWEDesigner::Client::Model::EditPanelModel SWEDesigner::Client::Model::ClassDiagram
R0F17.2.2	SWEDesigner::Client::View SWEDesigner::Client::Model::EditPanelModel SWEDesigner::Client::Model::ClassDiagram
R0F17.2.3	SWEDesigner::Client::View SWEDesigner::Client::Model::EditPanelModel

Codice Requisiti	Codice Use case
	SWEDesigner::Client::Model::ClassDiagram
R0F17.2.3.1	SWEDesigner::Client::View SWEDesigner::Client::Model::EditPanelModel SWEDesigner::Client::Model::ClassDiagram
R0F17.2.3.2	SWEDesigner::Client::View SWEDesigner::Client::Model::EditPanelModel SWEDesigner::Client::Model::ClassDiagram
R0F17.2.3.2.1	SWEDesigner::Client::View SWEDesigner::Client::Model::EditPanelModel SWEDesigner::Client::Model::ClassDiagram
R0F17.2.3.2.2	SWEDesigner::Client::View SWEDesigner::Client::Model::EditPanelModel SWEDesigner::Client::Model::ClassDiagram
R0F17.2.3.2.3	SWEDesigner::Client::View SWEDesigner::Client::Model::EditPanelModel SWEDesigner::Client::Model::ClassDiagram
R0F17.2.3.2.4	SWEDesigner::Client::View SWEDesigner::Client::Model::EditPanelModel SWEDesigner::Client::Model::ClassDiagram
R0F17.2.3.3	SWEDesigner::Client::View SWEDesigner::Client::Model::EditPanelModel SWEDesigner::Client::Model::ClassDiagram
R0F17.2.4	SWEDesigner::Client::View SWEDesigner::Client::Model::EditPanelModel SWEDesigner::Client::Model::ClassDiagram
R0F17.2.5	SWEDesigner::Client::View SWEDesigner::Client::Model::EditPanelModel SWEDesigner::Client::Model::ClassDiagram
R0F17.2.6	SWEDesigner::Client::View SWEDesigner::Client::Model::EditPanelModel SWEDesigner::Client::Model::ClassDiagram SWEDesigner::Client::Model::ActivityDiagram
R0F17.3	SWEDesigner::Client::View SWEDesigner::Client::Model::EditPanelModel SWEDesigner::Client::Model::ClassDiagram
R0F17.4	SWEDesigner::Client::View SWEDesigner::Client::Model::EditPanelModel

Codice Requisiti	Codice Use case
	SWEDesigner::Client::Model::ClassDiagram
R0F17.5	SWEDesigner::Client::View SWEDesigner::Client::Model::EditPanelModel SWEDesigner::Client::Model::ClassDiagram
R0F17.6	SWEDesigner::Client::View SWEDesigner::Client::Model::EditPanelModel SWEDesigner::Client::Model::ClassDiagram
R0F17.7	SWEDesigner::Client::View SWEDesigner::Client::Model::ClassDiagram
R0F17.8 SWEDesigner::Client::View	SWEDesigner::Client::Model::EditPanelModel SWEDesigner::Client::Model::ClassDiagram
R1F18	SWEDesigner::Client::View SWEDesigner::Client::Model::PackageToolbar SWEDesigner::Client::Model::EditPanelModel SWEDesigner::Client::Model::PackageDiagram
R1F18.1	SWEDesigner::Client::View SWEDesigner::Client::Model::PackageToolbar SWEDesigner::Client::Model::PackageDiagram
R1F18.2	SWEDesigner::Client::View SWEDesigner::Client::Model::PackageToolbar SWEDesigner::Client::Model::EditPanelModel SWEDesigner::Client::Model::PackageDiagram
R1F18.2.1	SWEDesigner::Client::View SWEDesigner::Client::Model::EditPanelModel SWEDesigner::Client::Model::PackageDiagram
R1F18.2.2	SWEDesigner::Client::View SWEDesigner::Client::Model::EditPanelModel SWEDesigner::Client::Model::PackageDiagram
R1F18.2.3	SWEDesigner::Client::View SWEDesigner::Client::Model::EditPanelModel SWEDesigner::Client::Model::PackageDiagram
R1F18.2.4	SWEDesigner::Client::View SWEDesigner::Client::Model::EditPanelModel SWEDesigner::Client::Model::PackageDiagram
R1F18.3	SWEDesigner::Client::View SWEDesigner::Client::Model::EditPanelModel

Codice Requisiti	Codice Use case
	SWEDesigner::Client::Model::PackageDiagram
R1F18.4	SWEDesigner::Client::View SWEDesigner::Client::Model::AddressModel SWEDesigner::Client::Model::PackageDiagram
R1F18.5	SWEDesigner::Client::View SWEDesigner::Client::Model::PackageToolBar SWEDesigner::Client::Model::PackageDiagram
R1F18.6	SWEDesigner::Client::View SWEDesigner::Client::Model::PackageDiagram
R1F18.7	SWEDesigner::Client::View SWEDesigner::Client::Model::PackageDiagram
R0F19	SWEDesigner::Client::View SWEDesigner::Client::Model::TitleBarModel SWEDesigner::Client::Model::ActivityToolBar SWEDesigner::Client::Model::EditPanelModel SWEDesigner::Client::Model::ActivityDiagram
R0F19.1	SWEDesigner::Client::View SWEDesigner::Client::Model::TitleBarModel SWEDesigner::Client::Model::ActivityToolBar SWEDesigner::Client::Model::ActivityDiagram
R0F19.2	SWEDesigner::Client::View SWEDesigner::Client::Model::EditPanelModel SWEDesigner::Client::Model::ActivityDiagram
R0F19.2.1	SWEDesigner::Client::View SWEDesigner::Client::Model::EditPanelModel SWEDesigner::Client::Model::ActivityDiagram
R0F19.2.2	SWEDesigner::Client::View SWEDesigner::Client::Model::EditPanelModel SWEDesigner::Client::Model::ActivityDiagram
R0F19.2.3	SWEDesigner::Client::View SWEDesigner::Client::Model::EditPanelModel SWEDesigner::Client::Model::ActivityDiagram
R0F19.2.4	SWEDesigner::Client::View SWEDesigner::Client::Model::EditPanelModel SWEDesigner::Client::Model::ActivityDiagram
R0F19.2.4.1	SWEDesigner::Client::View SWEDesigner::Client::Model::EditPanelModel

Codice Requisiti	Codice Use case
	SWEDesigner::Client::Model::ActivityDiagram
R0F19.2.4.2	SWEDesigner::Client::View SWEDesigner::Client::Model::EditPanelModel SWEDesigner::Client::Model::ActivityDiagram
R0F19.2.4.3	SWEDesigner::Client::View SWEDesigner::Client::Model::EditPanelModel SWEDesigner::Client::Model::ActivityDiagram
R0F19.2.4.4	SWEDesigner::Client::View SWEDesigner::Client::Model::EditPanelModel SWEDesigner::Client::Model::ActivityDiagram
R0F19.2.5	SWEDesigner::Client::View SWEDesigner::Client::Model::EditPanelModel SWEDesigner::Client::Model::ActivityDiagram
R0F19.2.6	SWEDesigner::Client::View SWEDesigner::Client::Model::EditPanelModel SWEDesigner::Client::Model::ActivityDiagram SWEDesigner::Client::Model::BubbleDiagram
R0F19.3	SWEDesigner::Client::View SWEDesigner::Client::Model::ActivityDiagram
R0F19.4	SWEDesigner::Client::View SWEDesigner::Client::Model::ActivityToolbar SWEDesigner::Client::Model::ActivityDiagram
R0F19.5	SWEDesigner::Client::View SWEDesigner::Client::Model::EditPanelModel SWEDesigner::Client::Model::ActivityDiagram
R0F19.6	SWEDesigner::Client::View SWEDesigner::Client::Model::ActivityDiagram
R0F19.7	SWEDesigner::Client::View SWEDesigner::Client::Model::ActivityToolbar SWEDesigner::Client::Model::ActivityDiagram
R0F19.8	SWEDesigner::Client::View SWEDesigner::Client::Model::EditPanelModel SWEDesigner::Client::Model::ActivityDiagram
R0F19.8.1	SWEDesigner::Client::View SWEDesigner::Client::Model::EditPanelModel SWEDesigner::Client::Model::ActivityDiagram
R0F19.8.2	SWEDesigner::Client::View



Codice Requisiti	Codice Use case
	SWEDesigner::Client::Model::EditPanelModel SWEDesigner::Client::Model::ActivityDiagram
R0F19.8.2.1	SWEDesigner::Client::View SWEDesigner::Client::Model::EditPanelModel SWEDesigner::Client::Model::ActivityDiagram
R0F19.8.2.2	SWEDesigner::Client::View SWEDesigner::Client::Model::EditPanelModel SWEDesigner::Client::Model::ActivityDiagram
R0F19.8.2.3	SWEDesigner::Client::View SWEDesigner::Client::Model::EditPanelModel SWEDesigner::Client::Model::ActivityDiagram
R0F19.8.3	SWEDesigner::Client::View SWEDesigner::Client::Model::ActivityDiagram
R0F19.9	SWEDesigner::Client::View SWEDesigner::Client::Model::ActivityDiagram
R0F19.10	SWEDesigner::Client::View SWEDesigner::Client::Model::ActivityDiagram
R0F19.11	SWEDesigner::Client::View SWEDesigner::Client::Model::ActivityToolbar SWEDesigner::Client::Model::ActivityDiagram
R0F19.12	SWEDesigner::Client::View SWEDesigner::Client::Model::EditPanelModel SWEDesigner::Client::Model::ActivityDiagram
R0F19.13	SWEDesigner::Client::View SWEDesigner::Client::Model::ActivityDiagram
R0F19.14	SWEDesigner::Client::View SWEDesigner::Client::Model::ActivityToolbar SWEDesigner::Client::Model::ActivityDiagram
R0F19.15	SWEDesigner::Client::View SWEDesigner::Client::Model::EditPanelModel SWEDesigner::Client::Model::ActivityDiagram
R0F19.16	SWEDesigner::Client::View SWEDesigner::Client::Model::ActivityDiagram
R0F19.17	SWEDesigner::Client::View SWEDesigner::Client::Model::AddressModel SWEDesigner::Client::Model::ClassDiagram SWEDesigner::Client::Model::ActivityDiagram

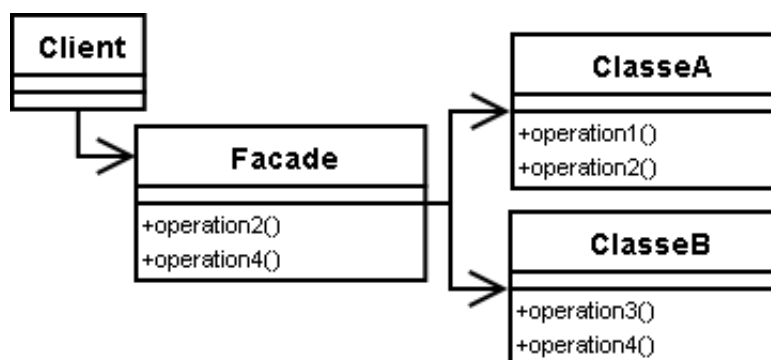
Codice Requisiti	Codice Use case
R0F20	SWEDesigner::Client::View SWEDesigner::Client::Model::TitleBarModel SWEDesigner::Client::Model::BubbleToolbar SWEDesigner::Client::Model::EditPanelModel SWEDesigner::Client::Model::BubbleDiagram
R0F20.1	SWEDesigner::Client::View SWEDesigner::Client::Model::BubbleToolbar SWEDesigner::Client::Model::BubbleDiagram
R0F20.2	SWEDesigner::Client::View SWEDesigner::Client::Model::EditPanelModel SWEDesigner::Client::Model::BubbleDiagram
R0F20.3	SWEDesigner::Client::View SWEDesigner::Client::Model::BubbleDiagram
R0F20.4	SWEDesigner::Client::View SWEDesigner::Client::Model::BubbleToolbar SWEDesigner::Client::Model::BubbleDiagram
R0F20.5	SWEDesigner::Client::View SWEDesigner::Client::Model::EditPanelModel SWEDesigner::Client::Model::BubbleDiagram
R0F20.6	SWEDesigner::Client::View SWEDesigner::Client::Model::BubbleDiagram
R0F20.7	SWEDesigner::Client::View SWEDesigner::Client::Model::AddressModel SWEDesigner::Client::Model::ActivityDiagram SWEDesigner::Client::Model::BubbleDiagram
R0F20.8	SWEDesigner::Client::View SWEDesigner::Client::Model::BubbleDiagram
R0F21	SWEDesigner::Client::Model::State
R0F22	SWEDesigner::Client::Model::DAO SWEDesigner::Client::Model::RequestHandler SWEDesigner::Server::RequestHandler SWEDesigner::Server::CodeGenerator::CoderGenerator

**Tabella 2:** Tracciamento Requisiti-Use case

## A Design pattern utilizzati

### A.1 Strutturali

#### A.1.1 Facade



**Figura 2:** Esempio pattern Facade

#### Scopo

Fornire un'interfaccia per l'accesso ad uno o più sottosistemi complessi nascondendone la complessità all'esterno.

#### Problema

Una parte del client necessita di una interfaccia semplificata alla funzionalità di un sottosistema complesso.

#### Struttura

La principale componente individuabile è la classe Facade che si interpone tra il sottosistema e l'esterno, ed associa ogni richiesta ad una classe del sottosistema, delegando la risposta.

#### Utilizzo nel progetto

Utilizzato, ad esempio, lato server nel CodeGenerator per interfacciarsi con le diverse componenti.

## A.2 Creazionali

### A.2.1 Singleton

Singleton
<u>- singleton : Singleton</u>
- Singleton() <u>+ getInstance() : Singleton</u>

**Figura 3:** Esempio pattern Singleton

#### Scopo

Garantire che venga creata una ed una sola istanza di una determinata classe e di fornirne un punto di accesso globale.

#### Problema

Il sistema richiede una ed una istanza di un oggetto; inoltre, sono necessari un accesso globale e la possibilità di effettuare un'inizializzazione pigra dell'istanza.

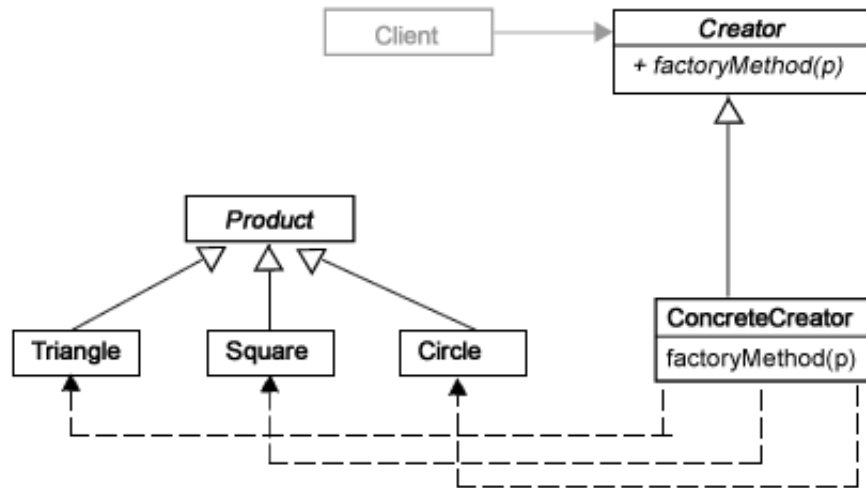
#### Struttura

L'unica componente è la classe Singleton che definisce i metodi opportuni per permettere la creazione di un'unica istanza e l'accesso a quest'ultima.

#### Utilizzo nel progetto

Utilizzato, ad esempio, lato client per fornire un'unica esistenza della classe State interna al Model.

### A.2.2 Factory



**Figura 4:** Esempio pattern Factory

#### Scopo

Permettere la creazione di oggetti senza esporre la logica creazionale al client mediante l'utilizzo di un'interfaccia.

#### Problema

Un'applicazione per essere portabile ha bisogno di incapsulare le dipendenze delle componenti del sistema in modo efficace ed efficiente.

#### Struttura

Sono individuabili tre componenti principali:

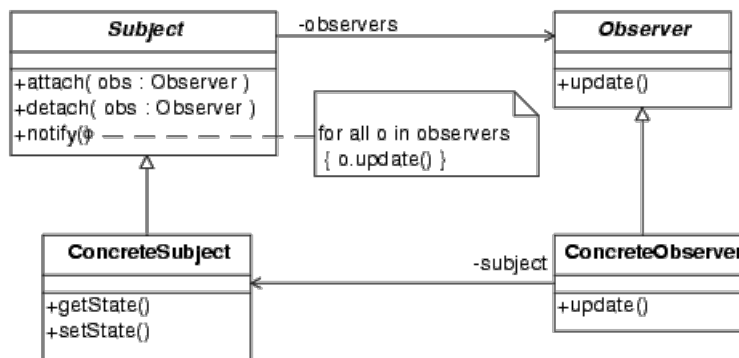
- Product: interfaccia di creazione dei prodotti;
- ConcreteProduct: implementa l'interfaccia secondo i metodi definiti;
- ConcreteCreator: definisce il factory method che ritornerà l'oggetto appropriato.

#### Utilizzo nel progetto

Utilizzato, ad esempio, lato server nella componente Coder per separare la costruzione degli oggetti generati.

## A.3 Comportamentali

### A.3.1 Observer



**Figura 5:** Esempio pattern Observer

#### Scopo

Definire una dipendenza "1..n" fra oggetti, riflettendo le modifiche dell'oggetto sui suoi dipendenti.

#### Problema

È necessario implementare un sistema di gestione di eventi provenienti da diversi oggetti.

#### Struttura

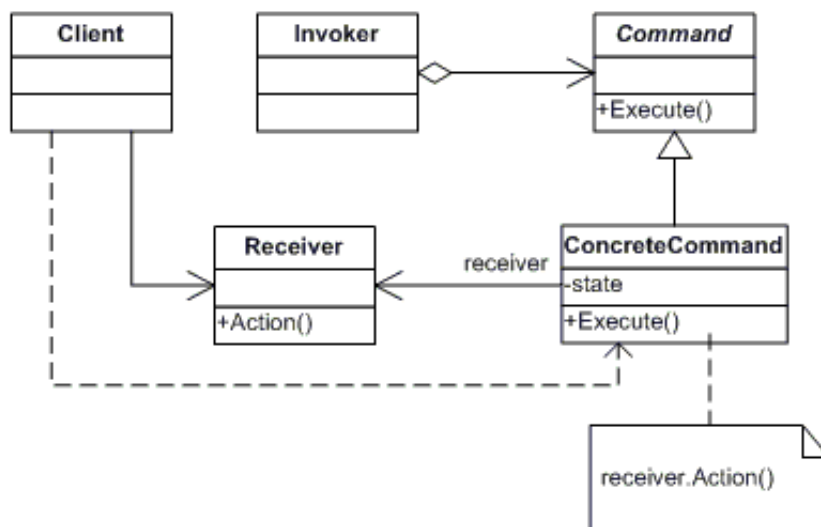
Sono individuabili quattro componenti principali:

- Subject: interfaccia per permettere agli osservatori di sottoscrivere e cancellarsi, avendo un riferimento ad ognuno di quelli iscritti;
- ConcreteSubject: mantiene lo stato di un oggetto concreto, notificando gli osservatori concreti in caso di cambiamenti;
- Observer: interfaccia per consentire agli osservatori di aggiornarsi al cambiamento di stato dell'oggetto osservato;
- ConcreteObserver: implementa l'interfaccia definita dall'Observer esplicitando le azioni da eseguire qualora si verifichi un cambio di stato dell'oggetto osservato.

#### Utilizzo nel progetto

Utilizzato, ad esempio, lato client dalla View per osservare i cambiamenti del Model; la sua implementazione è fornita da Backbone.js.

### A.3.2 Command



**Figura 6:** Esempio pattern Command

#### Scopo

Incapsulare il codice che effettua un'azione separandolo dall'oggetto che ne richiede l'esecuzione.

#### Problema

È necessario inviare richieste agli oggetti senza conoscere l'operazione richiesta o il destinatario della richiesta.

#### Struttura

Sono individuabili cinque componenti principali:

- **Client**: effettua la richiesta del comando e imposta il **Receiver**;
- **Receiver**: conosce come effettuare il comando;
- **Invoker**: effettua l'invocazione del comando;
- **Command**: interfaccia dei comandi;
- **ConcreteCommand**: implementa il comando e invoca l'operazione sul **Receiver**.

#### Utilizzo nel progetto

Utilizzato, ad esempio, lato client dove ogni intervento della View sul Model sarà trasmesso a quest'ultimo attraverso un command avente metodi di `"exec()"` e `"undo()"`.