Università degli Studi di Padova

KaleidosCode

SWEDesigner

Software per diagrammi UML

Specifica tecnica v2.0.0

Informazioni sul documento

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

kaleidos.codec6@gmail.com

Diario delle Modifiche

Versione	Data	Autore	Descrizione
1.0.1	30/05/2017	Sanna Giovanni	Correzione appendice A (Design pattern utilizzati) come da valutazione post RP
1.0.0	08/05/2017	Bonato Enrico	Approvazione Documento
0.1.0	07/05/2017	Sanna Giovanni	Verifica Documento
0.0.8	06/05/2017	Pace Giulio	Stesura Sezione Tracciamento
0.0.7	06/05/2017	Sovilla Matteo	Stesura Sezione Tracciamento
0.0.6	05/05/2017	Pace Giulio	Stesura sezione Componenti e Classi Principali
0.0.5	05/05/2017	Bonolo Marco	Stesura sezione Componenti e Classi Principali
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

Elenco delle tabelle

Elenco delle figure

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$_G$ utilizzati.

1.2 Scopo del prodotto

Lo scopo del progetto è la realizzazione di un software di costruzione di diagrammi UML$_G$ con la relativa generazione di codice Java$_G$ e Javascript$_G$ 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 v3.0.0*.
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$_G$ d'appalto**:
  http://www.math.unipd.it/~tullio/IS-1/2016/Progetto/C6.pdf (09/03/2017);
- **Norme di progetto**: *Norme di progetto v3.0.0*;
- **Analisi dei requisiti**: *Analisi dei requisiti v3.0.0*;
- **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.
- **Jointjs**: https://www.jointjs.com/opensource - 02/05/2017
- **jQuery**: https://jquery.com/ - 02/05/2017
- **Lodash**: https://lodash.com/ - 02/05/2017
- **Backbone.js**: http://backbonejs.org/ - 02/05/2017
- **Node.js**: https://nodejs.org/it/ - 02/05/2017
- **RequireJS**: http://requirejs.org/ - 02/05/2017
- **MySQL**: https://www.mysql.com/ - 02/05/2017

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;
- Il suo corretto utilizzo permette di separare struttura e contenuti delle pagine web dalla loro presentazione e comportamento. Questi ultimi vengono realizzati con altri linguaggi, aumentando quindi la manutenibilità del prodotto.

Principali svantaggi

- Linguaggio non ancora pienamente supportato da tutti i browser$_G$.

2.2 CSS

Linguaggio per la formattazione e presentazione delle pagine HTML$_G$, come richiesto dal Proponente.

Principali vantaggi

- Il suo corretto utilizzo permette di separare totalmente la presentazione della struttura 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$_G$, 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; di conseguenza il server non è sfruttato più del dovuto;
- Permette di gestire le azioni dell'utente attraverso la gestione degli eventi;
- Rende possibile interagire con il DOM$_G$.

Principali svantaggi

- Il linguaggio non è fortemente tipizzato.

2.4 JointJS$_G$

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

Principali vantaggi

- Fornisce elementi grafici di diagrammi UML;
- Elementi e collegamenti interattivi;
- Serializzazione/de-serializzazione da/a formato JSON$_G$;
- Supporto a Node.js$_G$.

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$_G$ 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 utilizzanti AJAX;
- Facilita la realizzazione di animazioni a livello base.

Principali svantaggi

- Non tutta la libreria è sviluppata rispettando uno standard comune, rendendo eventualmente necessario manipolarne il codice per i propri bisogni;

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 utili per la manipolazione degli oggetti;
- Rendono più semplice la valutazione e la manipolazione dei test.

2.7 Backbone.js

Framework$_G$ 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$_G$ 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$_G$ di terze parti.

Principali svantaggi

- Non supporta il data binding$_G$ bidirezionale;
- È difficile eseguire test di unità sulle views scrivendo poco codice per mocking$_G$.

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$_G$, 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$_G$, ecc.

Principali svantaggi

- Rende complessa la gestione di database$_G$ relazionali.

2.9 Bootstrap

Framework HTML, CSS e javaScript che facilita lo sviluppo di front-end.
(http://getbootstrap.com/ - 30/05/2017)

Principali vantaggi

- Librerie semplici e complete;
- Ben documentato;
- Utilizza la tecnologia Less, che permette a bootstrap di essere al passo con i tempi più velocemente rispetto ad altre librerie.

Principali svantaggi

- le pagine create con Bootstrap hanno un aspetto simile tra loro
- va inclusa tutta la libreria anche se ne viene usata una parte ridotta

2.10 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$_G$. 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 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.

2.11 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$_G$ 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.

2.12 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 che richiede uno studio molto approfondito in quanto non di apprendimento immediato.

2.13 MySQL

$DBMS_G$ 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 $big\ data_G$;
- Supporta moli di dati che possono essere anche molto grandi senza compromettere le prestazioni.

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$_G$) è una Single Page Application (SPA$_G$) scritta con i linguaggi HTML5, CSS$_G$ 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;

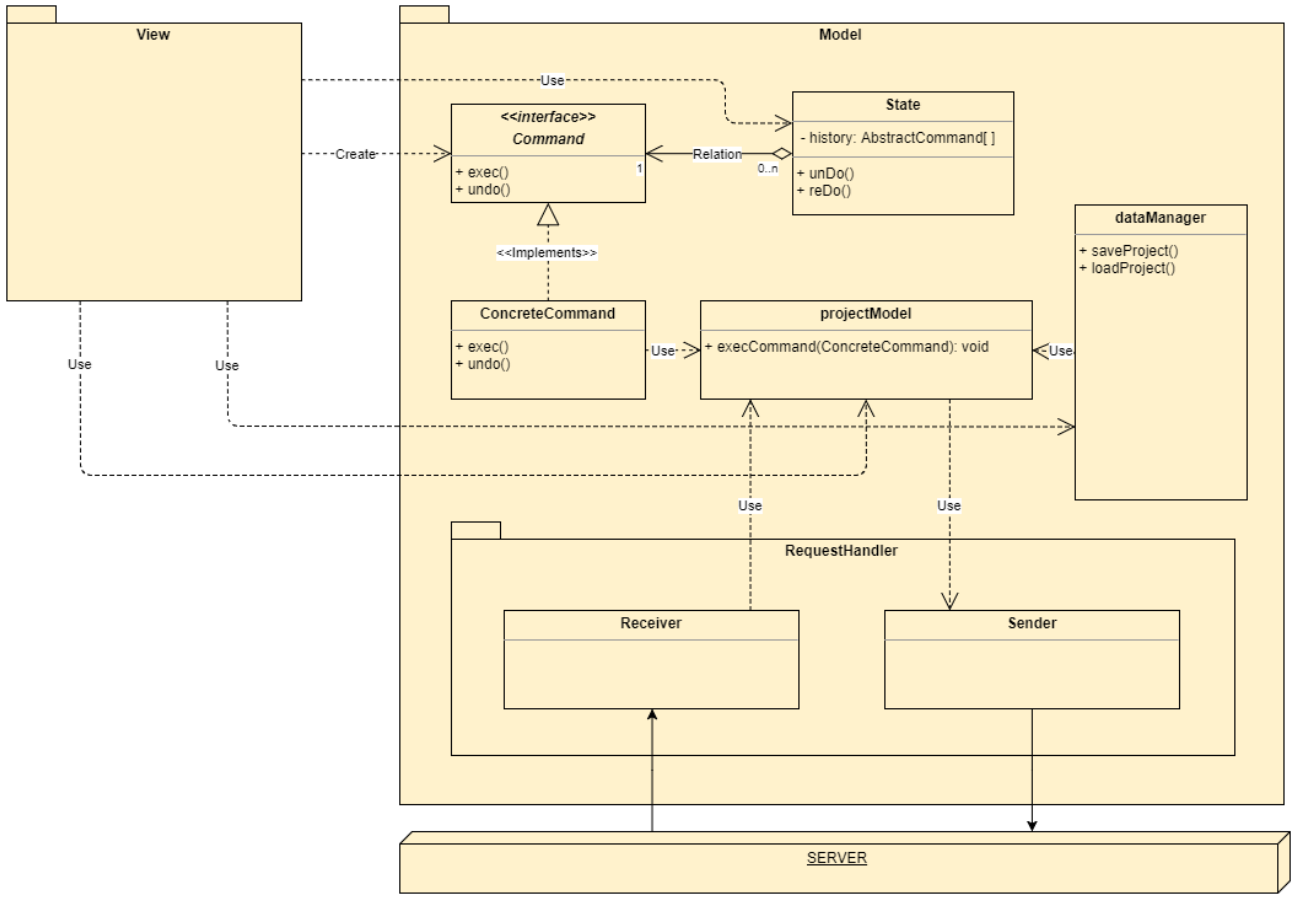

Figura 1: Architettura del client

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$_G$;
- 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$_G$) 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).

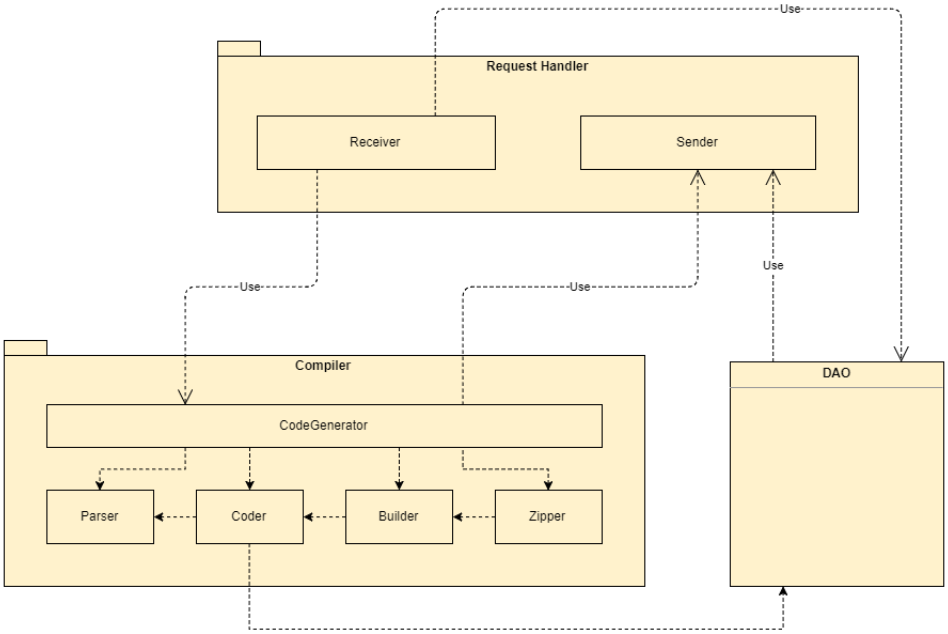

Figura 2: Architettura del server

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.

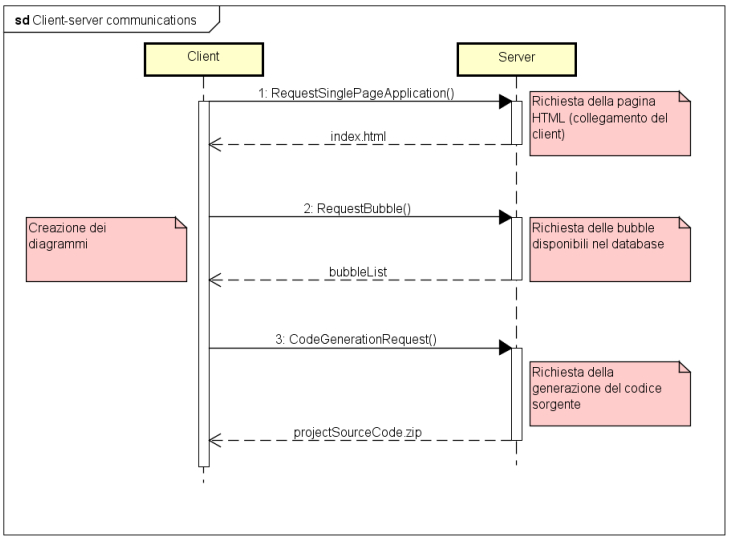

Figura 3: Esempi delle possibili comunicazioni client-server

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

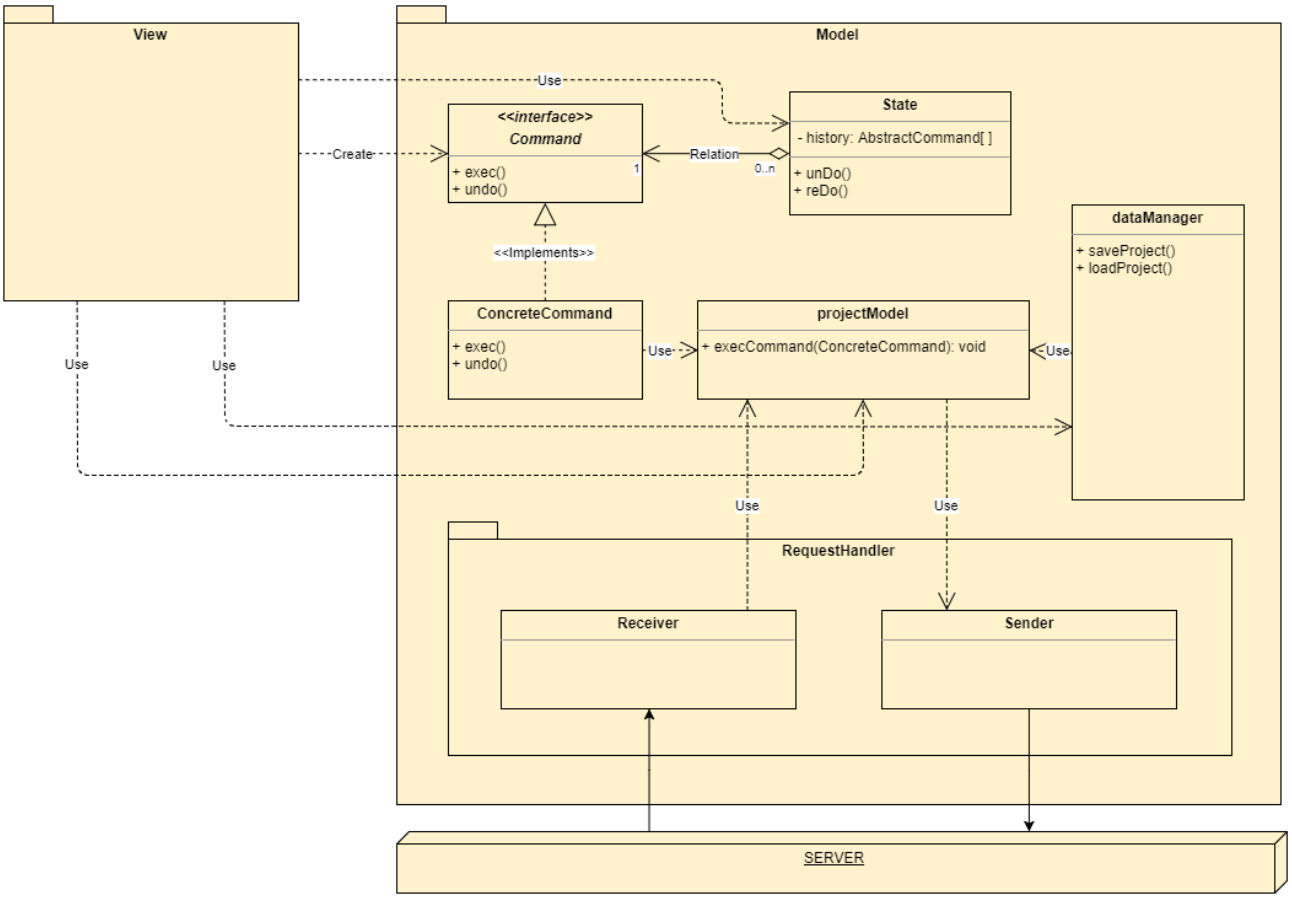

Figura 4: Architettura del client

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: implementa l'interfaccia Command per la rappresentazione concreta dei singoli comandi impartiti dai moduli View ai Model;
- View: il componente del programma che si occupa di gestire l'interfaccia grafica;
- State: gestisce la cronologia delle operazioni svolte permettendo le operazioni di unDo e reDo.

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.
FAN-OUT:

- Command: è l'interfaccia che rappresenta un generico comando impartito dai moduli View ai Model;
- projectModel: si occupa di gestire la parte logica dell'editor.

4.3.3 SWEDesigner::Client::Model::State

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

- View: il componente del programma che si occupa di gestire l'interfaccia grafica.

FAN-OUT:

- Command: è l'interfaccia che rappresenta un generico comando impartito dai moduli View ai Model.

.

4.3.4 SWEDesigner::Client::Model::dataManager

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

- View: il componente del programma che si occupa di gestire l'interfaccia grafica.

FAN-OUT:

- projectModel: si occupa di gestire la parte logica dell'editor;
- project: si occupa di gestire gli elementi contenuti nel diagramma.

4.3.5 SWEDesigner::Client::Model::projectModel

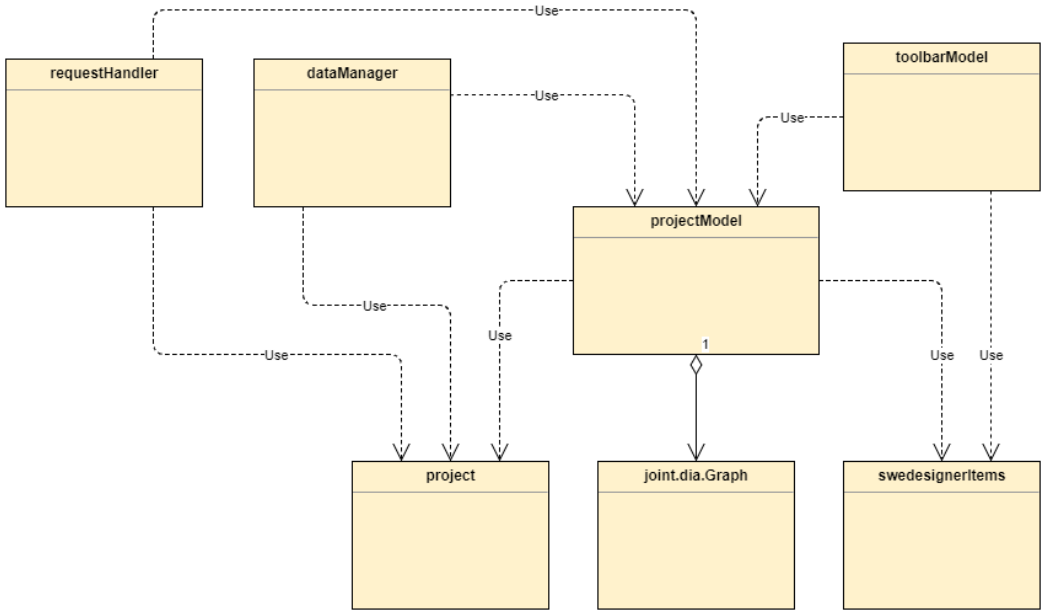

Figura 5: Architettura di Model

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

- ConcreteCommand: rappresenta i comandi inviati dalle View ed eseguiti poi da Model;
- dataManager: si occupa della persistenza dei dati, in particolare del salvataggio su file system locale del progetto e del caricamento di un progetto già esistente;
- View: invoca il metodo execCommand;
- Client::RequestHandler::Receiver: si occupa di gestire i dati ricevuti dal server.

FAN-OUT:

- Client::RequestHandler::Sender: si occupa di gestire le comunicazioni in uscita verso il server.

4.3.6 SWEDesigner::Client::Model::toolbarModel

È il componente del programma che si occupa di gestire la parte logica della toolbar.
FAN-IN:
Non ci sono dipendenze IN.
FAN-OUT:

- projectModel: si occupa di gestire la parte logica dell'editor;
- swedesignerItems: definisce il comportamento degli oggetti contenuti nel diagramma.

4.3.7 SWEDesigner::Client::Model::project

si occupa di gestire gli elementi contenuti nel diagramma.
FAN-IN:

- projectModel: si occupa di gestire la parte logica dell'editor;
- dataManager: Si occupa della persistenza dei dati, in particolare del salvataggio su file system locale del progetto già esistente;
- requestHandler: Si occupa di gestire le comunicazioni con il server.

FAN-OUT:
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:

- View: ne invoca i metodi.

FAN-OUT:

- projectModel: si occupa di gestire la parte logica dell'editor;
- project: si occupa di gestire gli elementi contenuti nel diagramma;
- Server::RequestHandler::Receiver: si occupa di gestire le comunicazioni in entrata dal Client.

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

Si occupa di gestire le comunicazioni in entrata dal server.
FAN-IN:

- Server::RequestHandler::Sender: si occupa di gestire le comunicazioni in uscita verso il Client.

FAN-OUT:

- projectModel: si occupa di gestire la parte logica dell'editor;
- project: si occupa di gestire gli elementi contenuti nel diagramma.

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::projectView

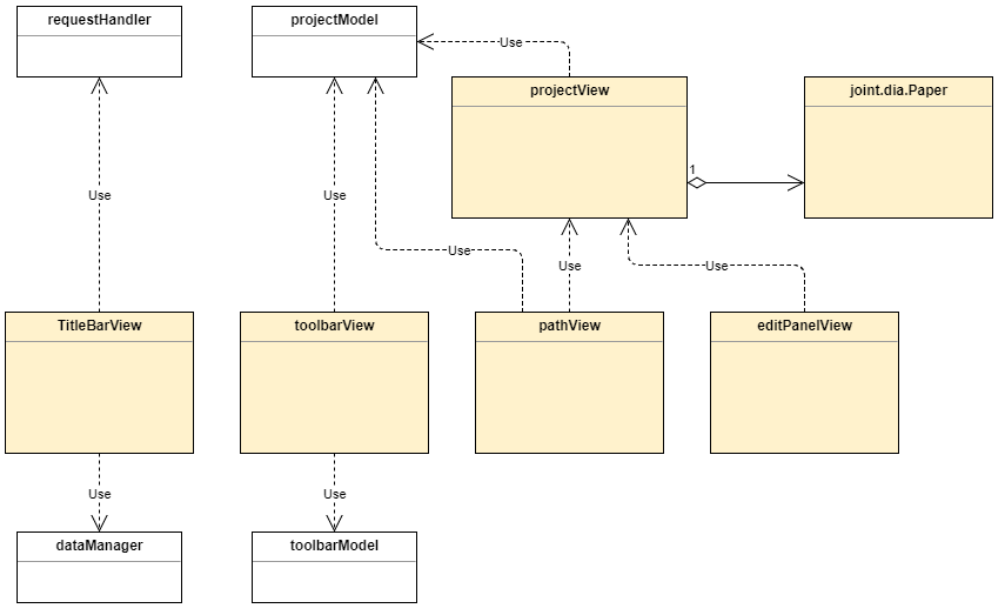

Figura 6: Architettura di projectView

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.
FAN-IN:

- pathView: gestisce l'interfaccia grafica della barra di indirizzo;
- editPanelView: gestisce l'interfaccia grafica del pannello di editing.

FAN-OUT:

- projectModel: si occupa di gestire la parte logica dell'editor;
- Paper: gestisce l'interfaccia grafica dell'area dei diagrammi.

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:
Non ci sono dipendenze IN.
FAN-OUT:

- requestHandler: gestisce la comunicazione con il server;
- dataManager: gestisce la persistenza dei dati su file system.

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:
Non ci sono dipendenze IN.
FAN-OUT:

- projectModel: si occupa di gestire la parte logica dell'editor;
- toolbarModel: si occupa di gestire la parte logica della toolbar.

4.5.4 SWEDesigner::Client::View::pathView

È il componente del programma che fa la funzione di view per il cosiddetto breadcrumb dove viene inserita la posizione corrente.
FAN-IN: Non ci sono dipendenze IN.
FAN-OUT:

- projectModel: si occupa di gestire la parte logica dell'editor;
- projectView: si occupa di gestire la parte grafica del model.

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:
Non ci sono dipendenze IN.
FAN-OUT:

- projectView: si occupa di gestire la parte grafica del model.

4.6 SWEDesigner::Server

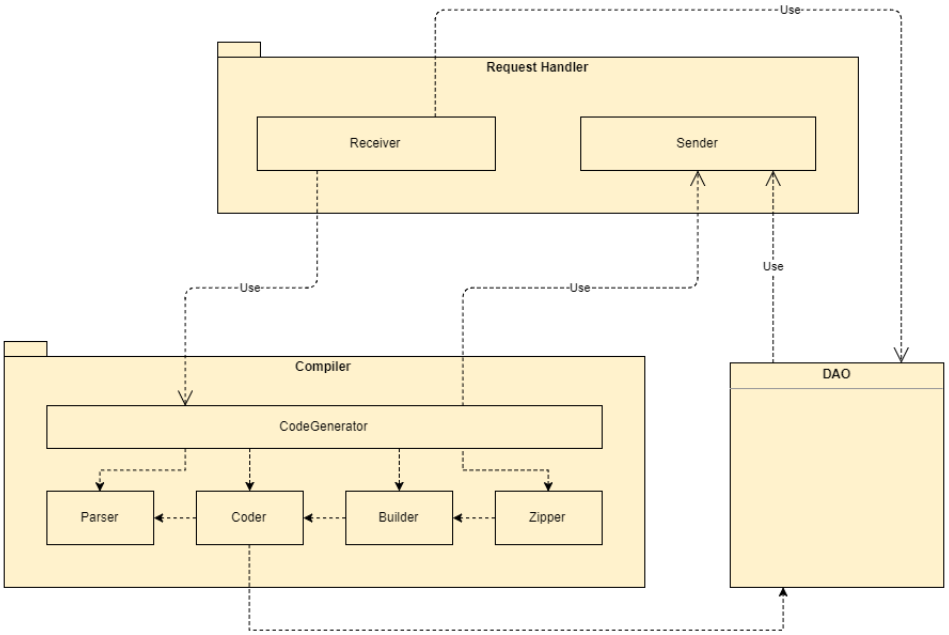

Figura 7: Architettura del server

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:

- Server::RequestHandler::Receiver: si occupa di gestire le comunicazioni in entrata dal client.

FAN-OUT:

- Server::RequestHandler::Sender: si occupa di gestire le comunicazioni in uscita verso il client;
- Parser: si occupa di creare un oggetto che contiene le informazioni ricevute in input;
- Coder: si occupa della traduzione in codice dell'oggetto ottenuto dal Parser;
- Builder: si occupa di organizzare in maniera organica il codice generato dal Coder;
- Zipper: si occupa di creare un archivio .zip contenente in codici sorgente precedentemente creati.

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.
FAN-IN:

- Zipper: si occupa di creare un archivio .zip contenente in codici sorgente precedentemente creati.

FAN-OUT:

- Coder: componente che funge da interfaccia alle operazioni di codifica di una stringa permettendo quindi di trasformare le informazioni del file in formato JSON in codice sorgente.

4.9 SWEDesigner::Server::CodeGenerator::Coder

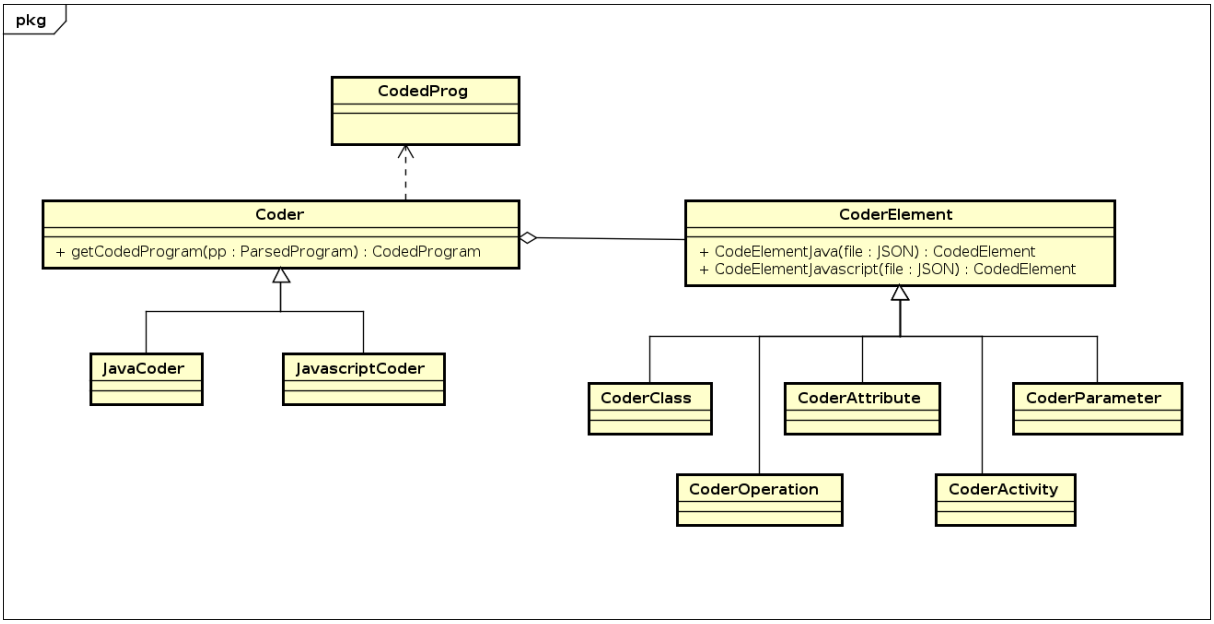

Figura 8: Architettura di Coder

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

4.9.1 SWEDesigner::Server::CodeGenerator::Coder::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.
FAN-IN:

- JavaCoder: si occupa di trasformare un oggetto JSON ricevuto in input in un oggetto contenente il codice sorgente scritto in java;
- JavaScriptCoder: si occupa di trasformare un oggetto JSON ricevuto in input in un oggetto contenente il codice sorgente scritto in javascript.

FAN-OUT:

- CodedProg: componente che contiene il codice prodotto dal Coder;
- CoderElement: componente astratto che offre la funzionalità che permette di associare ad ogni stringa contenuta nel file JSON il corrispondente codice sorgente.

4.9.2 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.
FAN-OUT:

- Coder: componente che funge da interfaccia alle operazioni di codifica di una stringa permettendo quindi di trasformare le informazioni del file in formato JSON in codice sorgente.

4.9.3 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.
FAN-OUT:

- Coder: componente che funge da interfaccia alle operazioni di codifica di una stringa permettendo quindi di trasformare le informazioni del file in formato JSON in codice sorgente.

4.9.4 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.
FAN-OUT:

- CoderElement: componente astratto che offre la funzionalità che permette di associare ad ogni stringa contenuta nel file JSON il corrispondente codice sorgente.

4.9.5 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.
FAN-OUT:

- CoderElement: componente astratto che offre la funzionalità che permette di associare ad ogni stringa contenuta nel file JSON il corrispondente codice sorgente.

4.9.6 SWEDesigner::Server::CodeGenerator::Coder::CoderParameter

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.
FAN-OUT:

- CoderElement: componente astratto che offre la funzionalità che permette di associare ad ogni stringa contenuta nel file JSON il corrispondente codice sorgente.

4.9.7 SWEDesigner::Server::CodeGenerator::Coder::CoderAttribute

È il componente che mette a disposizione la funzionalità, data una stringa in input in formato JSON che rappresenta un attributo valido, di ottenere il corrispondente codice sorgente di tale attributo. È possibile scegliere fra la codifica in Java o Javascript.
Non ci sono dipendenze IN.
FAN-OUT:

- CoderElement: componente astratto che offre la funzionalità che permette di associare ad ogni stringa contenuta nel file JSON il corrispondente codice sorgente.

4.9.8 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.
FAN-OUT:

- CoderElement: componente astratto che offre la funzionalità che permette di associare ad ogni stringa contenuta nel file JSON il corrispondente codice sorgente;
- DAO: si occupa di gestire il database delle bubble.

4.9.9 SWEDesigner::Server::CodeGenerator::Coder::CodedProg

È il componente che contiene il codice sorgente prodotto dal Coder.
FAN-IN:

- Coder: componente che funge da interfaccia alle operazioni di codifica di una stringa permettendo quindi di trasformare le informazioni del file in formato JSON in codice sorgente.

Non ci sono dipendenze OUT.

4.9.10 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: componente che funge da interfaccia alle operazioni di codifica di una stringa permettendo quindi di trasformare le informazioni del file in formato JSON in codice sorgente;
- CoderClass: componente che permette data una stringa in input in formato JSON che rappresenta un diagramma delle classi valido, di ottenere il corrispondente codice sorgente di tale classe;
- CoderOperations: componente che permette data una stringa in input in formato JSON che rappresenta un'operazione valida, di ottenere il corrispondente codice sorgente di tale operazione;
- CoderAttributes: componente che permette data una stringa in input in formato JSON che rappresenta un attributo valido, di ottenere il corrispondente codice sorgente di tale attributo;
- CoderActivity: componente che permette data una stringa in input in formato JSON che rappresenta un diagramma delle attività valido, di ottenere il corrispondente codice sorgente di tale attività;
- CoderParameter: componente che permette data una stringa in input in formato JSON che rappresenta un parametro valido, di ottenere il corrispondente codice sorgente di tale parametro.

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: si occupa di restituire in output un archivio zip contenente i codici sorgenti generati a partire dal file JSON ricevuto in input;
- Coder: componente che funge da interfaccia alle operazioni di codifica di una stringa permettendo quindi di trasformare le informazioni del file in formato JSON in codice sorgente.

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.
FAN-IN:

- CodeGenerator: si occupa di restituire in output un archivio zip contenente i codici sorgenti generati a partire dal file JSON ricevuto in input.

FAN-OUT:

- Builder: componente che si occupa di creare un oggetto contenitore con il codice sorgente, partendo dalle informazioni prese dal file JSON ricevuto in input che rappresenta un programma.

4.11.2 SWEDesigner::Server::DAO

Questa classe si occupa di gestire il database delle bubble.
FAN-IN:

- Coder: componente che funge da interfaccia alle operazioni di codifica di una stringa permettendo quindi di trasformare le informazioni del file in formato JSON in codice sorgente.

Non ci sono dipendenze OUT.

4.12 SWEDesigner::Server::RequestHandler

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

4.12.1 SWEDesigner::Server::RequestHandler::Sender

Si occupa di gestire le comunicazioni in uscita verso il client.
FAN-IN:

- CodeGenerator: si occupa di restituire in output un archivio zip contenente i codici sorgenti generati a partire dal file JSON ricevuto in input.

FAN-OUT:

- Client::Model::RequestHandler::Receiver: si occupa di gestire le comunicazioni in entrata dal server.

4.12.2 SWEDesigner::Server::RequestHandler::Receiver

Si occupa di gestire le comunicazioni in entrata dal client.
FAN-IN:

- Client::Model::RequestHandler::Sender: si occupa di gestire le comunicazioni in uscita verso il server.

FAN-OUT:

- CodeGenerator: si occupa di restituire in output un archivio zip contenente i codici sorgenti generati a partire dal file JSON ricevuto in input.

5 Tracciamento

Seguono le tabelle 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 Componenti
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 Componenti
	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 Componenti
	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 Componenti
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 Componenti
	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 SWEDesigner::Client::Model::ClassToolbar 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 Componenti
	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 SWEDesigner::Client::Model::ClassToolbar 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 Componenti
	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 Componenti
	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 Componenti
	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 Componenti
	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 Componenti
	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 Componenti
	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 Componenti
	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 Componenti
	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 Componenti
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-Componenti

5.2 Tracciamento componenti-requisiti

Codice Componenti	Codice Requisiti
SWEDesigner::Client::View	R0F1
	R0F2
	R0F3
	R0F4
	R0F4.1
	R1F8
	R0F11
	R0F15
	R0F15.1
	R0F15.2
	R0F15.2.1
	R0F15.2.2
	R0F15.2.3
	R0F15.2.4
	R0F15.2.4.1
	R0F15.2.4.2
	R0F15.2.4.3
	R0F15.2.5
	R0F15.3
	R0F15.4
	R0F15.4.1
	R0F15.4.2
	R0F15.4.3
	R0F15.4.4
	R0F15.4.5
	R0F15.4.6
	R0F15.4.7
	R0F15.5
	R0F15.5.1
	R0F15.5.2
	R0F15.5.3
	R0F15.5.4
	R0F15.5.5
	R0F15.5.6
	R0F15.5.7
	R0F15.6

Codice Componenti	Codice Requisiti
	R0F15.7
	R0F15.8
	R0F15.8.1
	R0F15.8.2
	R0F15.8.2.1
	R0F15.8.2.2
	R0F15.8.2.3
	R0F15.8.3
	R0F15.9
	R0F15.10
	R0F15.11
	R0F15.12
	R0F15.13
	R0F15.14
	R0F15.14.1
	R0F15.14.2
	R0F15.15
	R0F15.16
	R0F16
	R0F16.1
	R0F16.2
	R0F16.2.1
	R0F16.2.2
	R0F16.2.3
	R0F16.2.4
	R0F16.3
	R0F16.4
	R0F16.5
	R0F16.5.1
	R0F16.5.2
	R0F16.5.3
	R0F16.5.3.1
	R0F16.5.3.2
	R0F16.5.3.2.1
	R0F16.5.3.2.2
	R0F16.5.3.2.3
	R0F16.5.3.2.4
	R0F16.5.3.3

Codice Componenti	Codice Requisiti
	R0F16.5.4
	R0F16.6
	R0F16.7
	R0F16.8
	R0F16.9
	R0F16.10
	R0F16.10.1
	R0F16.10.2
	R0F16.10.3
	R0F16.10.4
	R0F16.10.5
	R0F16.10.6
	R0F16.10.7
	R0F17
	R0F17.1
	R0F17.2
	R0F17.2.1
	R0F17.2.2
	R0F17.2.3
	R0F17.2.3.1
	R0F17.2.3.2
	R0F17.2.3.2.1
	R0F17.2.3.2.2
	R0F17.2.3.2.3
	R0F17.2.3.2.4
	R0F17.2.3.3
	R0F17.2.4
	R0F17.2.5
	R0F17.2.6
	R0F17.3
	R0F17.4
	R0F17.5
	R0F17.6
	R0F17.7
	R0F17.8
	R1F18
	R1F18.1
	R1F18.2

Codice Componenti	Codice Requisiti
	R1F18.2.1
	R1F18.2.2
	R1F18.2.3
	R1F18.2.4
	R1F18.3
	R1F18.4
	R1F18.5
	R1F18.6
	R1F18.7
	R0F19
	R0F19.1
	R0F19.2
	R0F19.2.1
	R0F19.2.2
	R0F19.2.3
	R0F19.2.4
	R0F19.2.4.1
	R0F19.2.4.2
	R0F19.2.4.3
	R0F19.2.4.4
	R0F19.2.5
	R0F19.2.6
	R0F19.3
	R0F19.4
	R0F19.5
	R0F19.6
	R0F19.7
	R0F19.8
	R0F19.8.1
	R0F19.8.2
	R0F19.8.2.1
	R0F19.8.2.2
	R0F19.8.2.3
	R0F19.8.3
	R0F19.9
	R0F19.10
	R0F19.11
	R0F19.12

Codice Componenti	Codice Requisiti
	R0F19.13
	R0F19.14
	R0F19.15
	R0F19.16
	R0F19.17
	R0F20
	R0F20.1
	R0F20.2
	R0F20.3
	R0F20.4
	R0F20.5
	R0F20.6
	R0F20.7
	R0F20.8
SWEDesigner::Client::Model	R0F1
	R0F14
	R0F14.1
	R0F14.1.1
	R0F14.2
	R0F14.3
	R0F14.3.1
	R0F14.4
	R0F14.5
	R0F14.8
	R0F14.8.1
	R0F14.8.2
SWEDesigner::Client::Model::State	
	R0F14.4
	R0F14.5
	R0F21
SWEDesigner::Client::Model::DAO	R0F11
	R0F14
	R0F14.1
	R0F14.1.1
	R0F14.2
	R0F14.3
	R0F14.3.1

Codice Componenti	Codice Requisiti
	R0F14.8 R0F14.8.1 R0F14.8.2 R0F22
SWEDesigner::Client::Model::TitleBarModel	R0F2 R0F3 R0F4 R0F4.1 R1F8 R0F11 R0F15 R0F15.1 R0F15.2 R0F15.2.1 R0F15.2.2 R0F15.2.3 R0F15.2.4 R0F15.2.4.1 R0F15.2.4.2 R0F15.2.4.3 R0F15.7 R0F15.8 R0F15.8.1 R0F15.8.2 R0F15.8.2.1 R0F15.8.2.2 R0F15.8.2.3 R0F19 R0F19.1 R0F20
SWEDesigner::Client::Model::PackageToolbar	R0F11 R1F18 R1F18.1 R1F18.2 R1F18.5
SWEDesigner::Client::Model::ClassToolbar	R0F2 R0F4

Codice Componenti	Codice Requisiti
	R0F4.1 R0F15 R0F15.1 R0F15.2.2 R0F15.4 R0F15.4.1 R0F15.4.2 R0F15.4.3 R0F15.4.4 R0F15.4.5 R0F15.4.6 R0F15.4.7 R0F15.7 R0F15.10
SWEDesigner::Client::Model::ActivityToolbar	R0F3 R0F19 R0F19.1 R0F19.4 R0F19.7 R0F19.11 R0F19.14
SWEDesigner::Client::Model::BubbleToolbar	R1F8 R0F20 R0F20.1 R0F20.4
SWEDesigner::Client::Model::AddressModel	R0F15.2.5 R0F15.15 R1F18.4 R0F19.17 R0F20.7
SWEDesigner::Client::Model::EditPanelModel	R0F2 R0F4 R0F4.1 R0F15 R0F15.2 R0F15.2.1 R0F15.2.3

Codice Componenti	Codice Requisiti
	R0F15.2.4
	R0F15.2.4.1
	R0F15.2.4.2
	R0F15.2.4.3
	R0F15.5
	R0F15.5.1
	R0F15.5.2
	R0F15.5.3
	R0F15.5.4
	R0F15.5.5
	R0F15.5.6
	R0F15.5.7
	R0F15.8
	R0F15.8.1
	R0F15.8.2
	R0F15.8.2.1
	R0F15.8.2.2
	R0F15.8.2.3
	R0F15.8.3
	R0F15.10
	R0F16.1
	R0F16.2
	R0F16.2.1
	R0F16.2.2
	R0F16.2.3
	R0F16.2.4
	R0F16.3
	R0F16.4
	R0F16.5
	R0F16.5.1
	R0F16.5.2
	R0F16.5.3
	R0F16.5.3.1
	R0F16.5.3.2
	R0F16.5.3.2.1
	R0F16.5.3.2.2
	R0F16.5.3.2.3
	R0F16.5.3.2.4

Codice Componenti	Codice Requisiti
	R0F16.5.3.3
	R0F16.5.4
	R0F16.6
	R0F16.7
	R0F16.8
	R0F16.10
	R0F16.10.1
	R0F16.10.2
	R0F16.10.3
	R0F16.10.4
	R0F16.10.5
	R0F16.10.6
	R0F16.10.7
	R0F17
	R0F17.1
	R0F17.2
	R0F17.2.1
	R0F17.2.2
	R0F17.2.3
	R0F17.2.3.1
	R0F17.2.3.2
	R0F17.2.3.2.1
	R0F17.2.3.2.2
	R0F17.2.3.2.3
	R0F17.2.3.2.4
	R0F17.2.3.3
	R0F17.2.4
	R0F17.2.5
	R0F17.2.6
	R0F17.3
	R0F17.4
	R0F17.5
	R0F17.6
	R0F17.8
	R1F18
	R1F18.2
	R1F18.2.1
	R1F18.2.2

Codice Componenti	Codice Requisiti
	R1F18.2.3
	R1F18.2.4
	R1F18.3
	R0F19
	R0F19.2
	R0F19.2.1
	R0F19.2.2
	R0F19.2.3
	R0F19.2.4
	R0F19.2.4.1
	R0F19.2.4.2
	R0F19.2.4.3
	R0F19.2.4.4
	R0F19.2.5
	R0F19.2.6
	R0F19.5
	R0F19.8
	R0F19.8.1
	R0F19.8.2
	R0F19.8.2.1
	R0F19.8.2.2
	R0F19.8.2.3
	R0F19.12
	R0F19.15
	R0F20
	R0F20.2
	R0F20.5
SWEDesigner::Client::Model::PackageDiagram	R0F11
	R0F15.15
	R1F18
	R1F18.1
	R1F18.2
	R1F18.2.1
	R1F18.2.2
	R1F18.2.3
	R1F18.2.4
	R1F18.3

Codice Componenti	Codice Requisiti
	R1F18.4
	R1F18.5
	R1F18.6
	R1F18.7
SWEDesigner::Client::Model::ClassDiagram	R0F2
	R0F4
	R0F4.1
	R0F15
	R0F15.1
	R0F15.2
	R0F15.2.1
	R0F15.2.2
	R0F15.2.3
	R0F15.2.4
	R0F15.2.4.1
	R0F15.2.4.2
	R0F15.2.4.3
	R0F15.2.5
	R0F15.3
	R0F15.4
	R0F15.4.1
	R0F15.4.2
	R0F15.4.3
	R0F15.4.4
	R0F15.4.5
	R0F15.4.6
	R0F15.4.7
	R0F15.5
	R0F15.5.1
	R0F15.5.2
	R0F15.5.3
	R0F15.5.4
	R0F15.5.5
	R0F15.5.6
	R0F15.5.7
	R0F15.6
	R0F15.7

Codice Componenti	Codice Requisiti
	R0F15.8
	R0F15.8.1
	R0F15.8.2
	R0F15.8.2.1
	R0F15.8.2.2
	R0F15.8.2.3
	R0F15.8.3
	R0F15.9
	R0F15.10
	R0F15.11
	R0F15.12
	R0F15.13
	R0F15.14
	R0F15.14.1
	R0F15.14.2
	R0F15.15
	R0F15.16
	R0F16
	R0F16.1
	R0F16.2
	R0F16.2.1
	R0F16.2.2
	R0F16.2.3
	R0F16.2.4
	R0F16.3
	R0F16.4
	R0F16.5
	R0F16.5.1
	R0F16.5.2
	R0F16.5.3
	R0F16.5.3.1
	R0F16.5.3.2
	R0F16.5.3.2.1
	R0F16.5.3.2.2
	R0F16.5.3.2.3
	R0F16.5.3.2.4
	R0F16.5.3.3
	R0F16.5.4

Codice Componenti	Codice Requisiti
	R0F16.6
	R0F16.7
	R0F16.8
	R0F16.9
	R0F16.10
	R0F16.10.1
	R0F16.10.2
	R0F16.10.3
	R0F16.10.4
	R0F16.10.5
	R0F16.10.6
	R0F16.10.7
	R0F17
	R0F17.1
	R0F17.2
	R0F17.2.1
	R0F17.2.2
	R0F17.2.3
	R0F17.2.3.1
	R0F17.2.3.2
	R0F17.2.3.2.1
	R0F17.2.3.2.2
	R0F17.2.3.2.3
	R0F17.2.3.2.4
	R0F17.2.3.3
	R0F17.2.4
	R0F17.2.5
	R0F17.2.6
	R0F17.3
	R0F17.4
	R0F17.5
	R0F17.6
	R0F17.7
	R0F17.8
	R0F19.17
SWEDesigner::Client::Model::ActivityDiagram R0F3	R0F17.2.6

Codice Componenti	Codice Requisiti
	R0F19
	R0F19.1
	R0F19.2
	R0F19.2.1
	R0F19.2.2
	R0F19.2.3
	R0F19.2.4
	R0F19.2.4.1
	R0F19.2.4.2
	R0F19.2.4.3
	R0F19.2.4.4
	R0F19.2.5
	R0F19.2.6
	R0F19.3
	R0F19.4
	R0F19.5
	R0F19.6
	R0F19.7
	R0F19.8
	R0F19.8.1
	R0F19.8.2
	R0F19.8.2.1
	R0F19.8.2.2
	R0F19.8.2.3
	R0F19.8.3
	R0F19.9
	R0F19.10
	R0F19.11
	R0F19.12
	R0F19.13
	R0F19.14
	R0F19.15
	R0F19.16
	R0F19.17
	R0F20.7
SWEDesigner::Client::Model::BubbleDiagram	R1F8

Codice Componenti	Codice Requisiti
	R0F19.2.6 R0F20 R0F20.1 R0F20.2 R0F20.3 R0F20.4 R0F20.5 R0F20.6 R0F20.7 R0F20.8
SWEDesigner::Client::Model::RequestHandler	R0F14.6 R0F14.7 R0F22
SWEDesigner::Server::RequestHandler	R0F14.6 R0F14.7 R0F22
SWEDesigner::Server::CodeGenerator::CoderGenerator	R0F5 R0F6 R0F7 R0F14.6 R0F14.7 R0F22
SWEDesigner::Server::DAO	

Tabella 3: Tracciamento Componenti-Requisiti

A Design pattern utilizzati

A.1 Architetturali

A.1.1 MVC

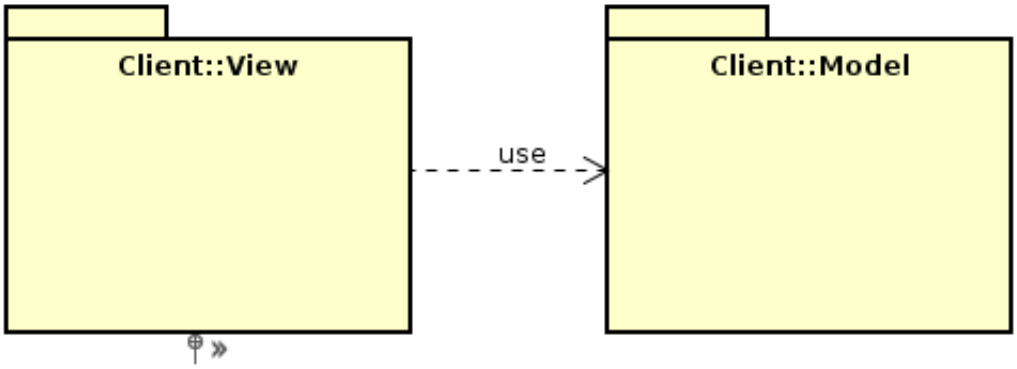

Figura 9: Esempio pattern MVC

Scopo

Pattern architetturale per la progettazione e strutturazione modulare di applicazioni software interattive. Consente di separare e disaccoppiare i componenti software che implementano il modello delle informazioni, dai componenti che implementano la logica di presentazione e la logica di controllo che gestiscono tali informazioni.

Problema

L'applicazione deve avere una natura modulare e basata sulle responsabilità, al fine di ottenere una vera e propria applicazione component - based. Questo è conveniente per poter più facilmente gestire la manutenzione dell'applicazione.
L'applicazione deve separare i componenti software che implementano il modello delle funzionalità di business, dai componenti che implementano la logica di presentazione e di controllo che utilizzano tali funzionalità.

Struttura

- Model : Definisce le regole di business per l'interazione con i dati, esponendo alla View ed al Controller rispettivamente le funzionalità per l'accesso e l'aggiornamento. Ha la responsabilità di notificare ai componenti della View eventuali aggiornamenti verificatisi in seguito a richieste del Controller, al fine di permettere alle View di presentare dati sempre aggiornati;
- View : Gestisce la logica di presentazione dei dati;

- Controller : implementa la logica di controllo dell'applicazione. Nella nostra particolare architettura, questa componente non è indipendente, ma bensì implementata dalla View, come in accordo con il framework utilizzato, ovvero Backbone.

Utilizzo nel progetto

Pattern architetturale utilizzato nella componente di front-end, ovvero SWEDesigner::Client. Tale pattern è stato adattato per rispecchiare il modello architetturale previsto da Backbone, il framework utilizzato per sviluppare la parte di front-end.

A.1.2 Data Access Object

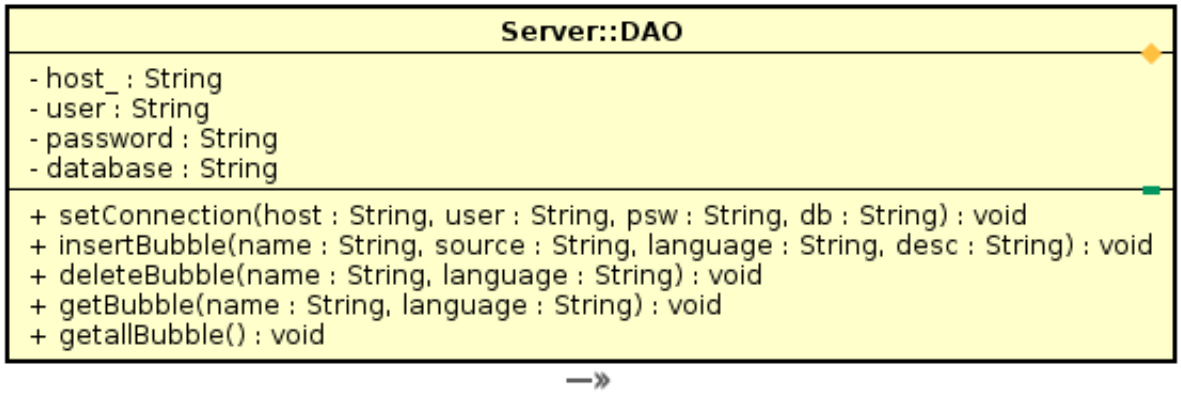

Figura 10: Esempio pattern DAO

Scopo

Pattern architetturale per la gestione della persistenza: si tratta fondamentalmente di una classe con relativi metodi che rappresenta un'entità tabellare di un RDBMS, usata per stratificare e isolare l'accesso ad una tabella tramite query (poste all'interno dei metodi della classe) ovvero al data layer da parte della business logic creando un maggiore livello di astrazione ed una più facile manutenibilità.

Problema

Necessità di separare le logiche di business da quelle di accesso ai dati; incapsulare l'accesso ai dati persistenti in un oggetto e fornire un'interfaccia per la manipolazione di tali dati.

Struttura

Composto da un'unica componente, DAO, che espone le funzionalità per accedere ai dati persistenti.

Utilizzo nel progetto

Utilizzato nella parte di back-end, per gestire la persistenza e gestione delle bubble.

A.2 Strutturali

A.2.1 Facade

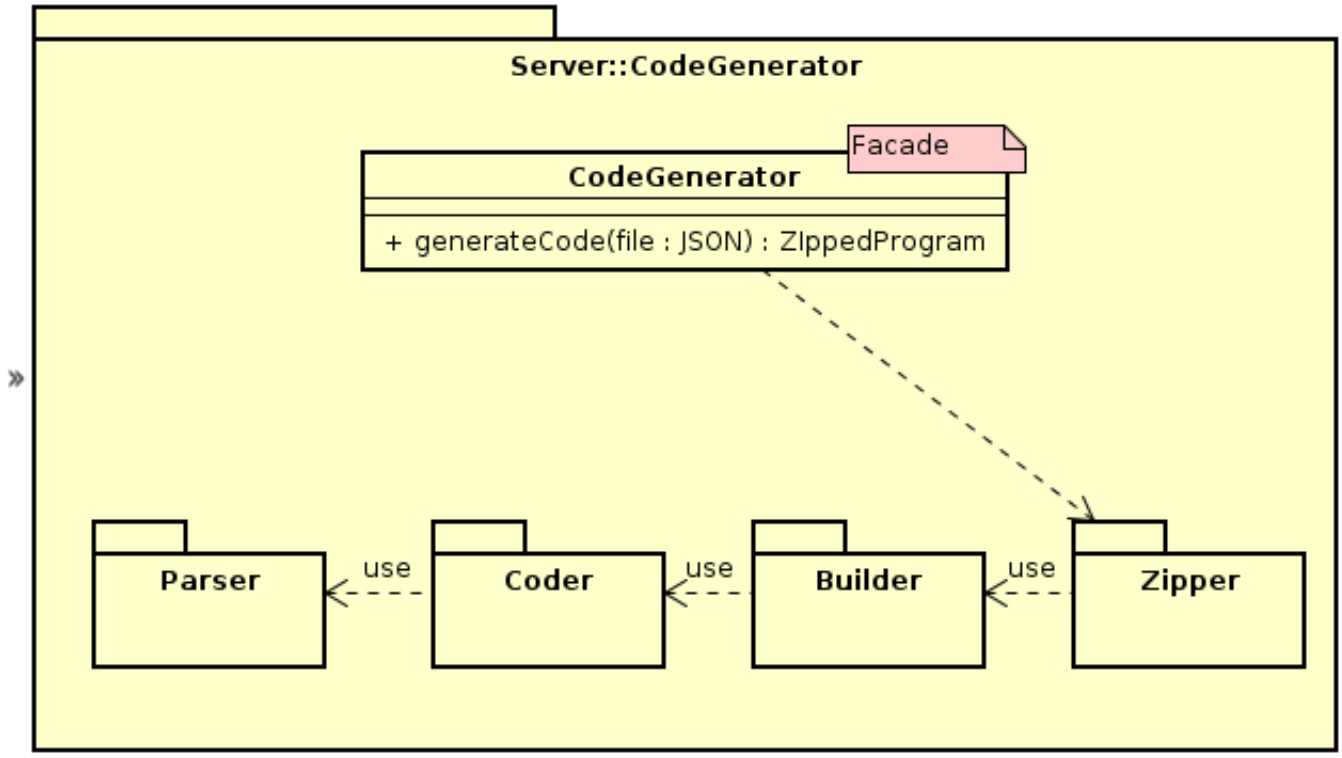

Figura 11: Esempio pattern Facade

Scopo

Fornire un'interfaccia unificata per l'accesso ad uno o più sottosistemi complessi nascondendone la complessità all'esterno. Strutturare un sistema in sottosistemi aiuta a ridurre la complessità; un obiettivo comune è minimizzare la comunicazione tra questi sottosistemi: una soluzione è introdurre un oggetto Facade che fornisca una singola, semplificata interfaccia alle funzioni più generali di un sottosistema.

Problema

I sottosistemi, man mano che evolvono, tendono a creare molte e piccole classi, al fine di aumentare la riusabilità e la facilità di personalizzazione, ma in questo modo diventano complesse da utilizzare e la maggior parte dei clients non necessitano di conoscere i dettagli del sottosistema; un Facade fornisce una semplice interfaccia, ma sufficiente alla maggior parte dei clients, di un sistema 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.

- Facade : espone le funzionalità di cui necessitano la maggior parte dei client, nascondendo la complessità del sottosistema; Nella figura, tale componente è rappresentata dalla classe CodeGenerator, che espone la funzionalità di creazione del programma richiesto dal client;
- Sottosistema: Implementano le funzionalità del sottosistema; gestiscono le attività assegnate dal Facade e non hanno nessun riferimento ad esso. Nella figura, il sottosistema è composto dai moduli Parser, Coder,Builder e Zipper, i quali si occupano di implementare la funzionalità di generazione del programma.

Utilizzo nel progetto

Utilizzato, ad esempio, lato server nel CodeGenerator in cui la classe CodeGenerator funge da Facade e espone la funzionalità che permette la generazione del codice sorgente, senza dover interfacciarsi con i vari moduli che implementano tale funzionalità.

A.3 Creazionali

A.3.1 Builder

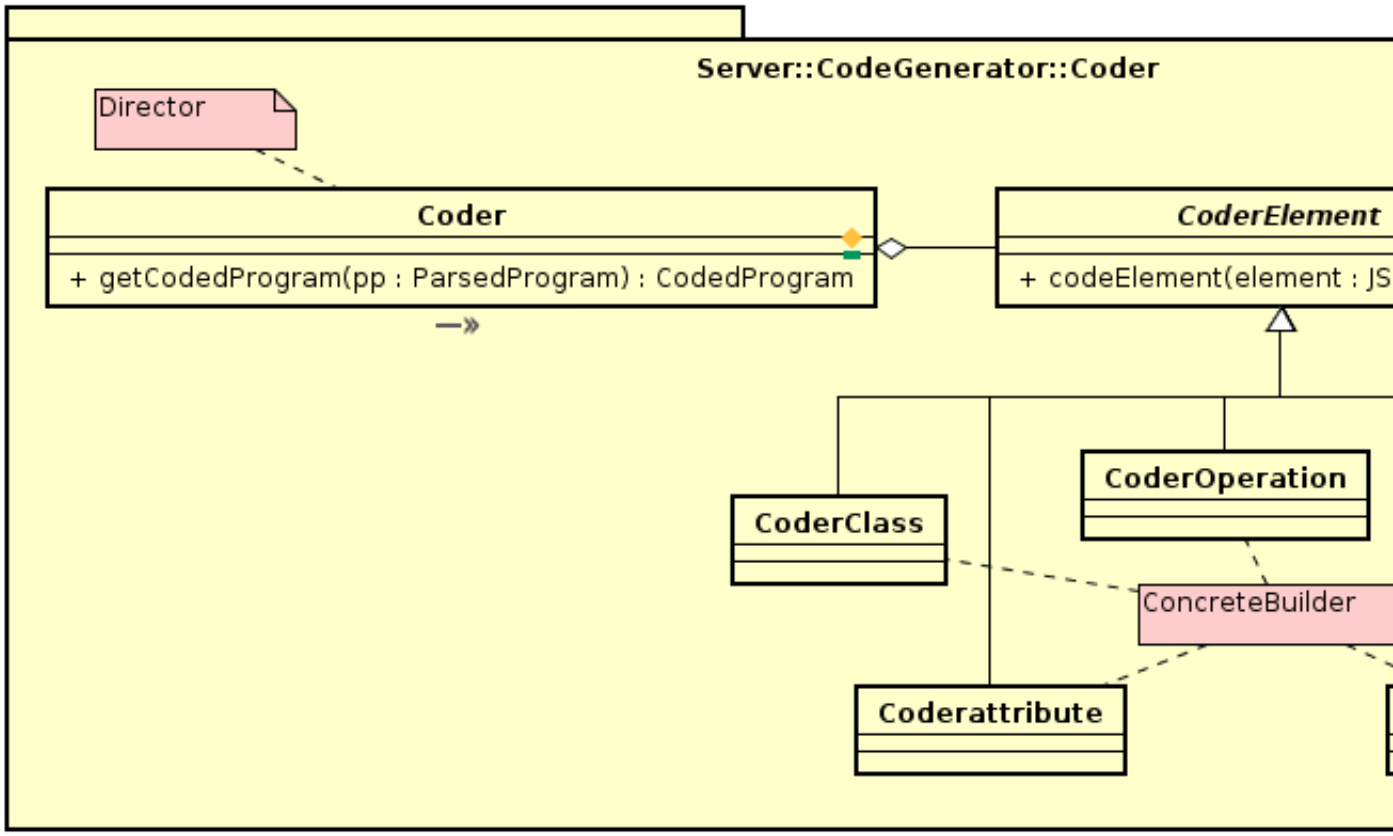

Figura 12: Esempio pattern Builder

Scopo

Separa la costruzione di un oggetto complesso dalla sua rappresentazione, in modo che lo stesso processo di costruzione consenta la creazione di diverse rappresentazioni.

Problema

L'algoritmo di creazione di un oggetto complesso dovrebbe essere indipendente dalle parti che compongono l'oggetto e da come queste sono assemblate.

Struttura

Il "Builder" pattern propone separare la "logica del processo di costruzione" dalla "costruzione stessa". Per fare ciò si utilizza un oggetto Director, che determina la logica di costruzione del prodotto, e che invia le istruzioni necessarie ad un oggetto Builder, incaricato della sua realizzazione. Siccome i prodotti da realizzare sono di diversa natura, ci saranno Builder particolari per ogni tipo di prodotto, ma soltanto un unico Director, che nel processo di costruzione invocherà i metodi del Builder scelto secondo il tipo di prodotto desiderato (i Builder dovranno implementare un'interfaccia comune per consentire al Director di interagire con tutti questi).
Sono individuabili quattro componenti principali:

- Director: Costruisce un oggetto utilizzando il builder (procedura). Nella figura, tale componente è rappresentata da Coder;
- Builder: Specifica l'interfaccia da utilizzare per assemblare il prodotto. Nella figura, tale componente è rappresentata da CoderElement;
- ConcreteBuilder: Costruisce e assembla le parti del prodotto. Tiene traccia dell'istanza costruita. Fornisce un'interfaccia per recuperare il prodotto finale. Nella figura, sono presenti diverse componenti di questo tipo: CoderClass, CoderOperation, CoderActivity, CoderAttribute, CoderParameter;
- Product: Prodotto da costruire. Nel nostro caso non si tratta di uno specifico oggetto, ma bensì della stringa che rappresenta il codice sorgente creato dalo specifico ConcreteBuilder.

Utilizzo nel progetto

Utilizzato, ad esempio, lato server nella componente Coder per costruire i vari elementi codificati di una classe.

A.4 Comportamentali

A.4.1 Observer

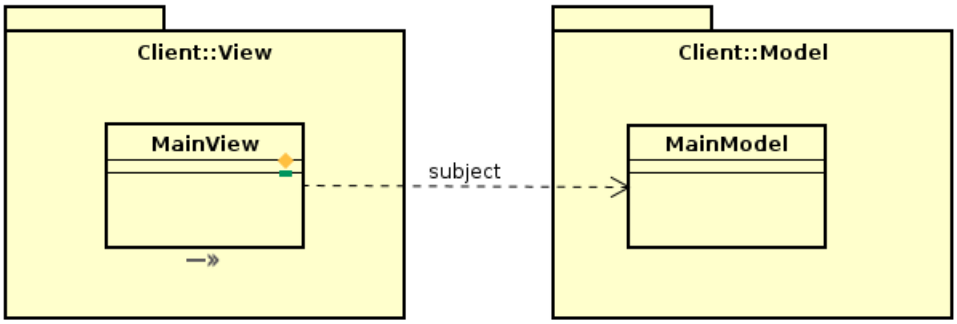

Figura 13: Esempio pattern Observer

Scopo

Definire una dipendenza "1..n" fra oggetti, riflettendo le modifiche dell'oggetto sui suoi dipendenti. Consente la definizione di associazioni di dipendenza di molti oggetti verso di uno, in modo che se quest'ultimo cambia il suo stato, tutti gli altri sono notificati e aggiornati automaticamente.

Problema

Il cambiamento si stato di un oggetto richiede la modifica dello stato di altri oggetti, senza conoscere quanti oggetti hanno effettivamente bisogno di cambiare il loro stato.
Un oggetto deve avere la capacità di notificare altri oggetti senza fare assunzioni su chi questi oggetti siano; in altre parole, non si vuole che questi oggetti siano fortemente accoppiati.

Struttura

Sono individuabili quattro componenti principali:

- ConcreteSubject: mantiene lo stato di un oggetto concreto, notificando gli osservatori concreti in caso di cambiamenti. Nella figura, tale componente è rappresentata da SWEDedigner::Client::Model::MainModel;
- ConcreteObserver: esplicita le azioni da eseguire qualora si verifichi un cambio di stato dell'oggetto osservato. Nella figura, tale componente è rappresentata da SWEDedigner::Client::View::MainView;

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.4.2 Command

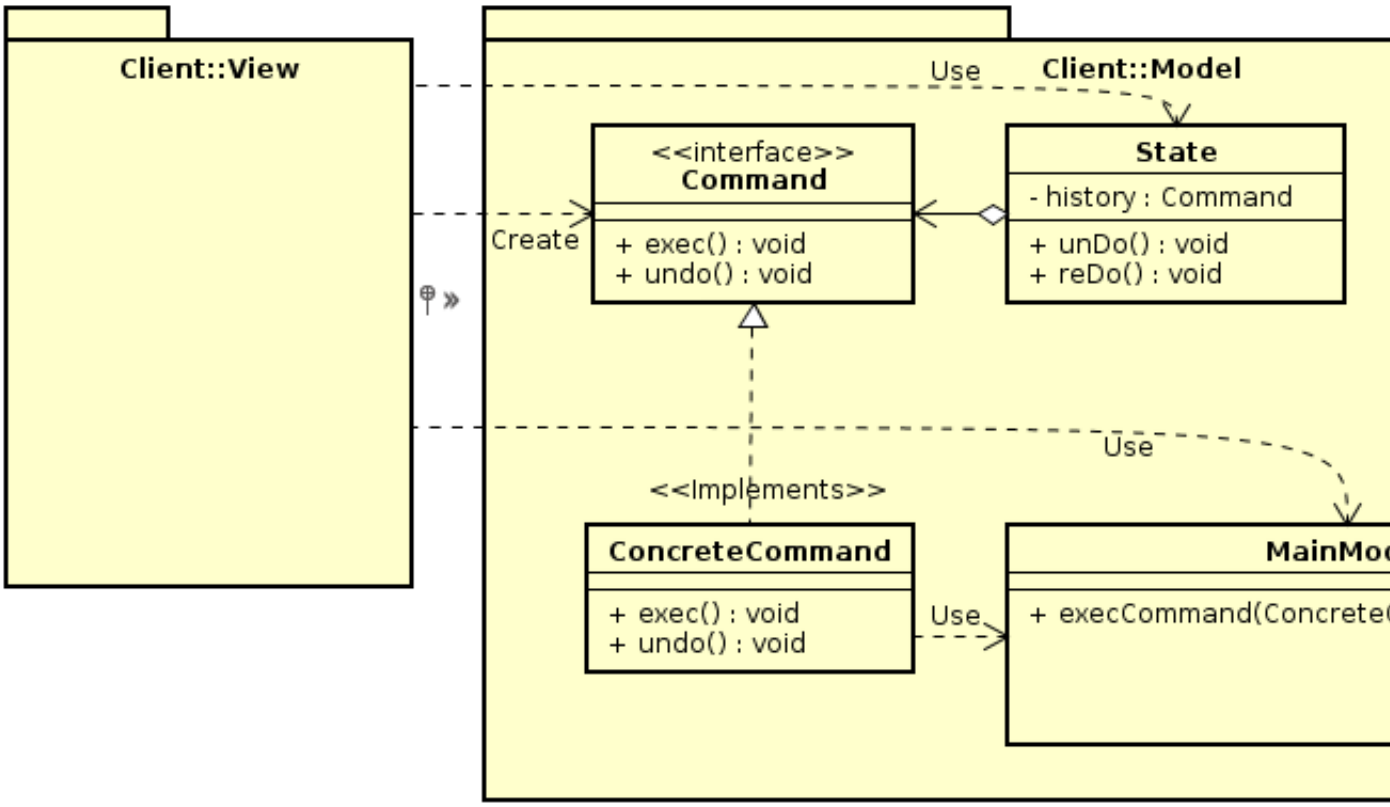

Figura 14: Esempio pattern Command

Scopo

Incapsulare il codice che effettua un'azione separandolo dall'oggetto che ne richiede l'esecuzione. Incapsula una richiesta all'interno di un oggetto, cosicché i client siano indipendenti dalle richieste; questo consente la parametrizzazione dei client con diverse tipologie di richieste, accodare o rintracciare le richieste, oppure supportare operazioni di undo.

Problema

Necessità di gestire richieste di cui non si conoscono i particolari; ovvero le effettive operazioni che verranno richieste e/o i ricevitori di tali richieste.
Necessità di fornire operazioni di undo.

Struttura

Sono individuabili quattro componenti principali:

- Receiver: conosce l'effettiva implementazione dell'operazione/i associata con la gestione di una richiesta. Nella figura, tale componente è rappresentata da SWEDesigner::Client::Model::MainModel;

- Invoker: effettua l'invocazione del comando necessario a portare a termine la richiesta. Nella figura, tale componente è rappresentata dal modulo SWEDesigner::Client::View;
- Command: dichiara un'interfaccia per l'esecuzione di un'operazione. Nella figura, tale componente è rappresentata da SWEDesigner::Client::Model::Command;
- ConcreteCommand: definisce un legame tra l'oggetto Receiver e un'azione. Implementa l'esecuzione del comando invocando la corrispondente operazione/i sull'oggetto Receiver. Nella figura, tale componente è rappresentata da SWEDesigner::Client::Model::ConcreteCommand.

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()".