



Manuale Sviluppatore

7DOS - 12 Aprile 2019

Informazioni sul documento

Versione	1.0.0
Responsabile	Andrea Trevisin
Verifica	Lorenzo Busin Nicolò Tartaggia
Redazione	Giacomo Barzon Marco Costantino Michele Roverato Giovanni Sorice
Stato	Approvato
Uso	Esterno
Destinato a	Prof. Tullio Vardanega Prof. Riccardo Cardin 7DOS
Email	7dos.swe@gmail.com

Descrizione

Questo documento contiene il manuale sviluppatore relativo al prodotto G&B del gruppo 7DOS.

Diario delle modifiche

Modifica	Autore	Ruolo	Data	Versione
<i>Approvazione del documento</i>	Andrea Trevisin	Responsabile	2018-12-11	1.0.0
<i>Verifica del documento</i>	Nicolò Tartaggia	Verificatore	2018-12-10	0.7.0
<i>Verifica del documento</i>	Lorenzo Busin	Verificatore	2018-12-9	0.6.0
<i>Stesura capitolato C4</i>	Giovanni Sorice	Analista	2018-12-08	0.5.0
<i>Stesura capitolati C1 e C2</i>	Giacomo Barzon	Analista	2018-12-07	0.4.0
<i>Stesura capitolato C3</i>	Giovanni Sorice	Analista	2018-12-06	0.3.0
<i>Stesura capitolato C6</i>	Michele Roverato	Analista	2018-12-06	0.2.0
<i>Stesura capitolato C5</i>	Marco Costantino	Analista	2018-12-05	0.1.0
<i>Stesura della sezione Introduzione</i>	Giovanni Sorice	Analista	2018-12-05	0.0.2
<i>Stesura dello scheletro del documento</i>	Giovanni Sorice	Analista	2018-12-05	0.0.1

Indice

1	Introduzione	3
1.1	Scopo del documento	3
1.2	Scopo del prodotto	3
1.3	Riferimenti	3
1.3.1	Informativi	3
1.3.2	Installazione	3
1.3.3	Legali	4
1.4	Glossario	4
2	Requisiti di sistema	5
2.1	Requisiti hardware	5
2.2	Browser compatibili	5
3	Installazione	6
3.1	Node.js	6
3.2	Installazione Grafana	6
3.3	Installazione plugin	6
3.4	Installazione e configurazione InfluxDB	6
4	Configurazione dell'ambiente di lavoro	6
4.1	Obiettivo del capitolo	6
4.2	Requisiti	6
4.2.1	WebStorm	6
4.2.2	VSCode	6
4.2.3	TSLint e ESLint	6
5	Test	7
5.1	Test sul codice TypeScript	7
5.2	Test sul codice HTML/CSS	7
5.3	Code Coverage	7
6	Architettura	8
6.1	Persistence Layer	9
6.1.1	NetworkAdapter	9
6.1.2	NetBuilder	9
6.1.3	NetParser	10
6.2	Buisness Layer	10
6.3	Database Layer	10
6.4	Presentation Layer	10

1 Introduzione

1.1 Scopo del documento

Questo documento rappresenta il Manuale dello Sviluppatore relativo al *prodotto software_g* G&B sviluppato dal gruppo 7DOS. Il suo scopo principale è fornire tutte le informazioni necessarie per usufruire delle funzionalità fino ad ora implementate ed, eventualmente, per estendere e migliorare il prodotto.

1.2 Scopo del prodotto

Lo scopo del prodotto è la creazione di un *plug-in_g* per il software di monitoraggio *Grafana_g* in grado di collegare dati provenienti da una specifica *datasource_g* ad un sistema probabilistico definito in una *rete Bayesiana_g*. In seguito, i risultati ottenuti vengono memorizzati in un database, nel nostro caso *influxDB_g*, e letti attraverso un *panel_g* standard di Grafana. In questo modo è possibile evidenziare eventi non visibili ma con alta *likelihood_g*.

1.3 Riferimenti

1.3.1 Informativi

- Rete Bayesiana
https://en.wikipedia.org/wiki/Bayesian_network

1.3.2 Installazione

- TypeScript
<https://www.typescriptlang.org/index.html#download-links>
- Node.js
<https://nodejs.org/it/>
- Grafana
<http://docs.grafana.org/installation/>
- Grafana Plug-in
<http://docs.grafana.org/plugins/installation/>
- InfluxDB
<https://portal.influxdata.com/downloads/>
- VSCode
<https://code.visualstudio.com/>
- JetBrains WebStorm
<https://www.jetbrains.com/webstorm/>
- JetBrains WebStorm for students
<https://www.jetbrains.com/student/>

1.3.3 Legali

- Licenza MIT

<https://opensource.org/licenses/MIT>

1.4 Glossario

Per rendere la lettura del documento più semplice, chiara e comprensibile viene allegato il *Glossario_v3.0.0* nel quale sono contenute le definizioni dei termini tecnici, dei vocaboli ambigui, degli acronimi e delle abbreviazioni. La presenza di un termine all'interno del Glossario è segnalata con una 'g' posta come pedice (esempio: *Glossario_g*).

2 Requisiti di sistema

2.1 Requisiti hardware

Per poter utilizzare il plug-in e tutto ciò che permette la sua esecuzione sono presenti dei requisiti hardware da soddisfare:

- RAM
- Memoria interna
- Processore
- Altro?

2.2 Browser compatibili

Di seguito vengono riportate le versioni minime dei browser sui quali è garantito il funzionamento del nostro plug-in:

- Google Chrome v.73 (quella nel pc di tartizz)
- Mozilla Firefox v.66 (quella nel pc di tartizz)
- Safari v.
- Microsoft Internet Explorer v.11 (quella nel pc di tartizz)
- Microsoft Edge v.41 (quella nel pc di tartizz)

Affinché tutte le funzionalità offerte dal plug-in vengano eseguite correttamente, è necessario che *JavaScript_g* sia abilitato.

3 Installazione

3.1 Node.js

3.2 Installazione Grafana

3.3 Installazione plugin

3.4 Installazione e configurazione InfluxDB

4 Configurazione dell'ambiente di lavoro

4.1 Obbiettivo del capitolo

Il seguente capitolo si impone l'obbiettivo di spiegare al lettore come configurare il proprio ambiente di lavoro in modo tale che sia lo stesso dei membri del team 7DOS.

4.2 Requisiti

4.2.1 WebStorm

Per lo sviluppo del plugin il team ha scelto di utilizzare l'IDE WebStorm sviluppato da JetBrains. L'IDE è a pagamento, tuttavia esso può essere scaricato gratuitamente dal loro sito ufficiale connettendo al proprio account un'e-mail universitaria. Il software è disponibile per Microsoft Windows, Linux e MacOS.

4.2.2 VSCode

VsCode è una valida alternativa a WebStorm, che il team consiglia di utilizzare nel caso in cui quest'ultimo non fosse reperibile. Il software può essere scaricato molto facilmente dal sito ufficiale ed è disponibile per Microsoft Windows, Linux e MacOS.

4.2.3 TSLint e ESLint

TSLint e ESLint verranno automaticamente installati con l'esecuzione del comando:

```
npm install
```

Una volta installati correttamente WebStorm li rileverà automaticamente tra le dipendenze presenti all'interno del package.json e procederà a segnalare tutti gli errori relativi all'analisi statica rilevati senza la necessità di dover eseguire il comando:

```
npm run build
```

5 Test

5.1 Test sul codice TypeScript

5.2 Test sul codice HTML/CSS

Verrà fatto per la RA

5.3 Code Coverage

6 Architettura

All'interno della seguente sezione verrà descritta l'architettura dell'intero sistema. Per la realizzazione dell'architettura del plugin il team ha deciso di adottare uno stile architetturale a layer.

I principali layer che sono stati individuati nel corso dello sviluppo dell'applicazione sono i seguenti:

- **Presentation layer:** si occupa di gestire la componente grafica del panel;
- **Persistence layer:** si occupa di mantenere la struttura della rete;
- **Buisness layer:** incorpora tutta la logica relativa all'intero processo di ricalcolo delle probabilità;
- **Database layer:** composto dall'insieme di tutti i database esterni forniti dall'utente per il reperimento delle informazioni.

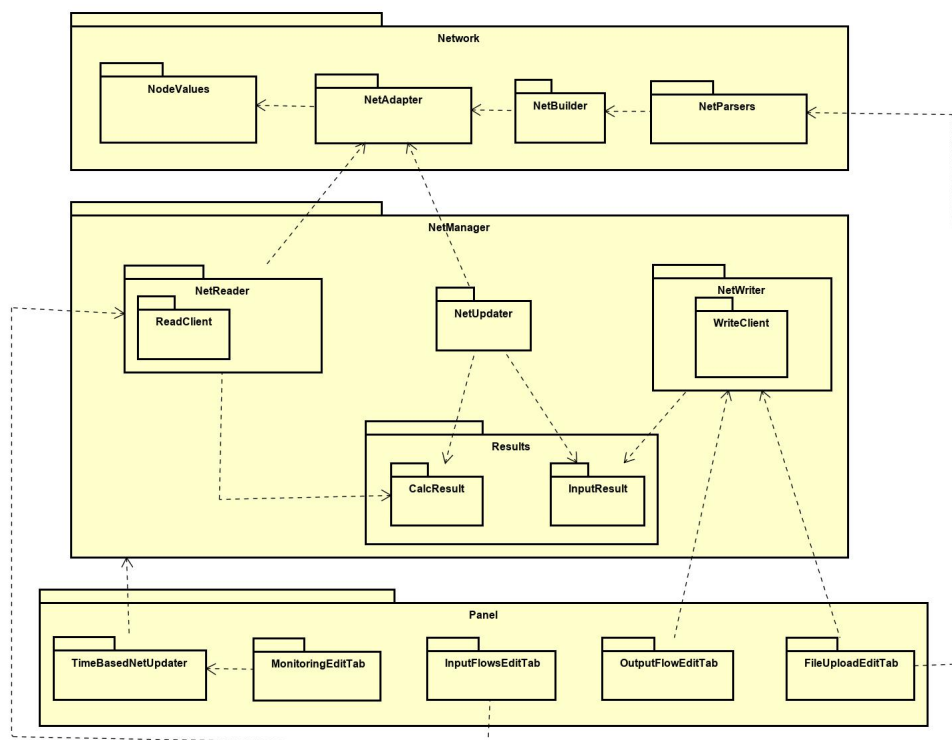


Figura 1: Diagramma dei package

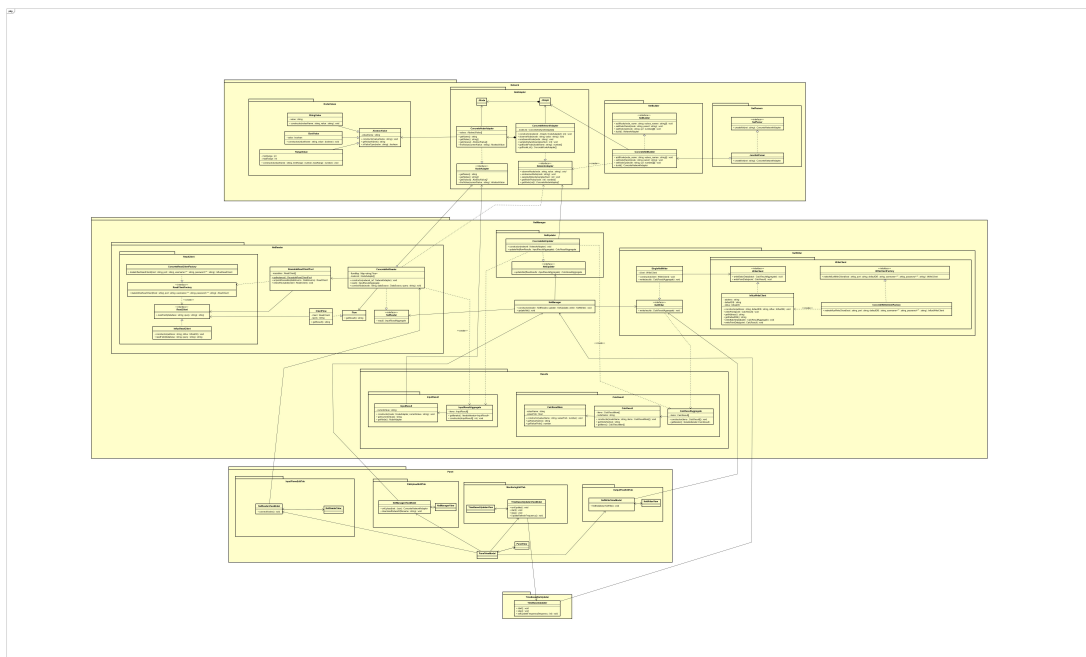


Figura 2: Diagramma delle classi complessivo

Analizziamo ora più in dettaglio ciascuno dei layer precedentemente descritti.

6.1 Persistence Layer

L'intera nostra applicazione è basata sulla libreria Javascript JSBayes consigliata dal proponente, la quale permette di mantenere l'intera struttura della rete ed esegue su di essa il calcolo delle probabilità di ogni stato associato ad ogni nodo della rete. Questa libreria tuttavia presenta non pochi problemi, tra i quali una pressochè totale assenza della gestione degli errori, una difficile interazione con altri oggetti, ed una scarsa estendibilità. Il persistence layer è composto principalmente da un insieme di classi che permettono di ovviare ai problemi precedentemente descritti.

6.1.1 NetworkAdapter

La classe NetworkAdapter ha il compito di schermare tutte le operazioni non sicure di JSBayes, cioè quelle che vanno a modificare la struttura della rete rischiando quindi di minare la potenziale integrità dei dati. Essa espone solamente i metodi necessari per effettuare il ricalcolo delle probabilità e che quindi non modificano in alcun modo lo stato.

Figura 3: Diagramma UML NetworkAdapter

6.1.2 NodeAdapter

Per lo sviluppo delle funzionalità del plugin il team ha avuto la necessità di aggiungere informazioni ai singoli nodi della rete bayesiana memorizzata da JSBayes. Estendere dalle

classi offerte dalla libreria era impensabile a causa dei molteplici problemi descritti all'inizio di questa sezione. Per questo motivo abbiamo deciso di realizzare un adapter anche per i singoli nodi di JSBayes ed includere all'interno di questi tutte le informazioni aggiuntive necessarie, come gli `AbstractValue` che verranno descritti in dettaglio all'interno della sezione seguente.

Figura 4: Diagramma UML NodeAdapter

6.1.3 NetBuilder

Per garantire l'integrità dei dati abbiamo deciso di esternalizzare il processo di creazione della rete JSBayes all'interno di un builder esterno il quale ha il compito di effettuare tutti i controlli necessari e ritornare un riferimento ad un `NetworkAdapter` solo al termine dell'intero processo di creazione invocando il metodo `build()`.

Figura 5: Diagramma UML NetBuilder

6.1.4 NetParser

Il `NetParser` ha il compito di realizzare un `NetAdapter`, sfruttando le funzionalità esposte dal `NetBuilder`, a partire dal contenuto di un file di testo. Il team fino ad ora ha realizzato una unica implementazione, quella relativa a file di tipo JSON in quanto era l'unico formato richiesto dal proponente. Nel caso in cui un giorno fosse necessario utilizzare formati differenti è sarà sufficiente realizzare una differente implementazione dell'interfaccia `NetParser`.

Figura 6: Diagramma UML NetParser

6.2 Buisness Layer

6.3 Database Layer

6.4 Presentation Layer