Entwurfsdokumentation

Visuelle Programmiersprache für den Physikunterricht zur Datenerfassung auf einem Raspberry Pi

Version 0.0.0

David Gawron Stefan Geretschläger Leon Huck
Jan Küblbeck Linus Ruhnke

24. Juni 2019

Inhaltsverzeichnis

1 Ziel der Entwurfsdokumentation

2 Ablaufkommunikation

2.1 Klassenübersicht

2.2 Klassendokumentation

2.3 Sequenzdiagramme

3 Konfigurationsfeld

3.1 Klassenübersicht

3.2 Klassendokumentation

3.3 Sequenzdiagramme

4 Bausteinprototypen

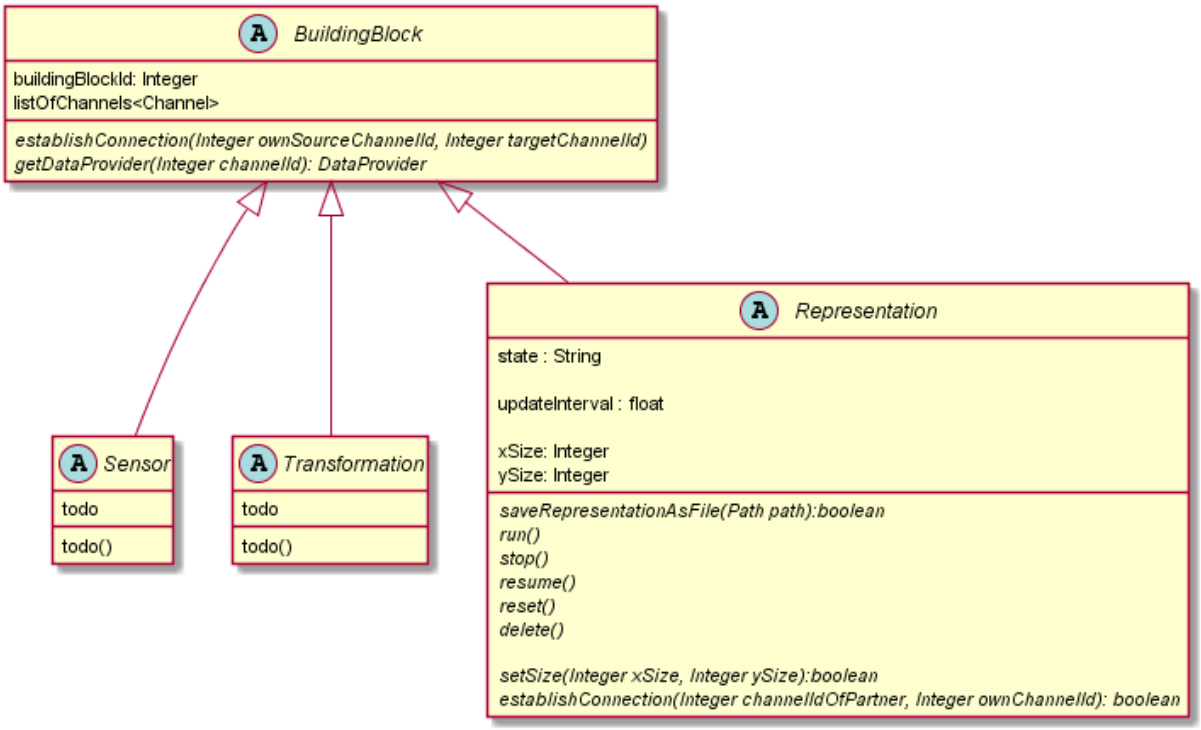

Abbildung 1: Klassendiagramm zur Übersicht der Struktur von Bausteinen

In Abbildung 1 ist das Klassendiagramm zur Übersicht der Struktur von Bausteinen dargestellt. Ein Baustein ist durch eine abstrakte Klasse repräsentiert. Weiter wird er durch eine der drei abstrakten Unterklassen als Sensor, Transformation oder Darstellung konkretisiert. Jeder Bausteinblock enthält eine List seiner Kanäle und kann durch diese mit anderen Bausteinen verbunden werden.

In Abbildung 2 ist das Klassendiagramm zur Kommunikation zwischen Bausteinen dargestellt. Durch diese Klasse ChannelList und Channel wird die Verbindung zwischen zwei Kanälen definiert. Ein Kanal kann mit einem anderen Kanal über die Methode setConnectionPartner(String id): boolean verbunden werden. Dabei ist nur eine Verbindung zwischen zwei Kanälen mit unterschiedlichem Typ(Eingang oder Ausgang) möglich. Einem Kanal wird auch ein DataProvider zugewiesen. Dabei handelt es sich um einen konkreten Datenstrom, z.B. von einem Sensor. Dabei kann ein Kanal nur einen DataProvider von einem Ausgang eines anderen Bausteins erhalten. Die Klasse DataProvider ist dabei hinter dem Interface IDataProvider versteckt, um später andere Arten von DataProvidern implementieren zu können.

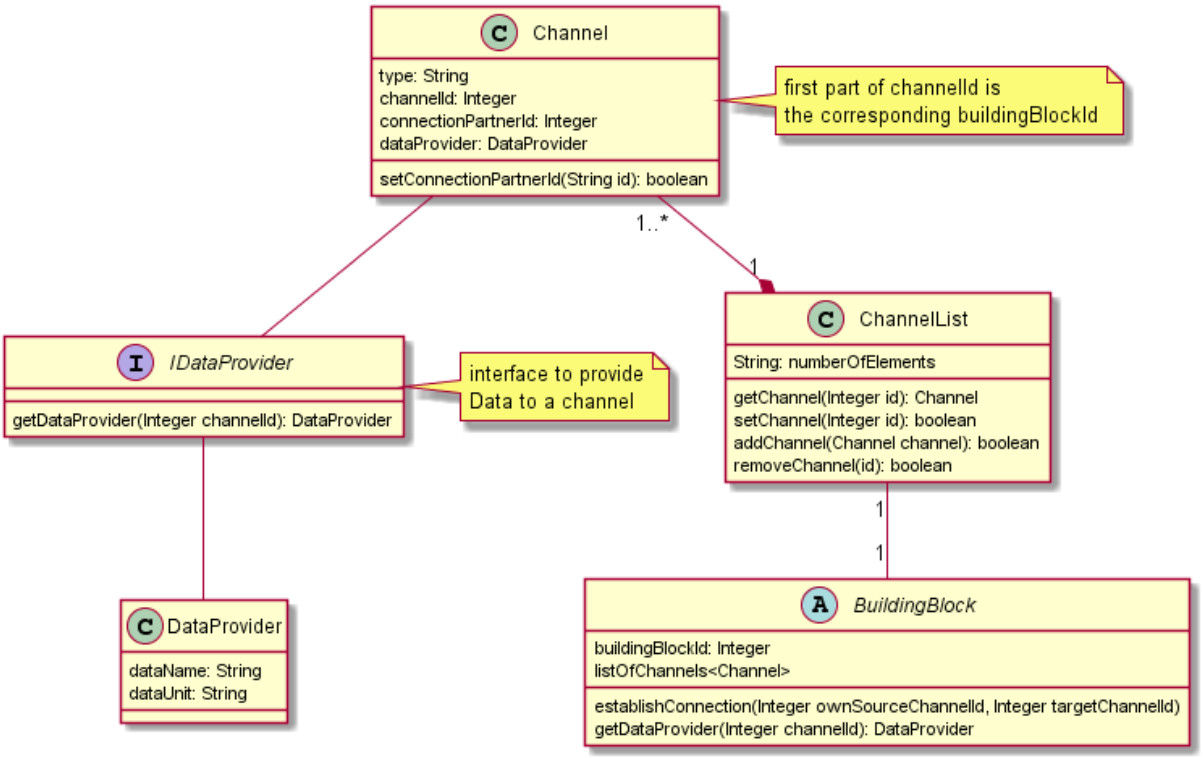

Abbildung 2: Klassendiagramm zur Übersicht der Kommunikation zwischen Bausteinen

4.1 Sensoren

to do

4.2 Transformationen

to do

4.3 Darstellungen

In Abbildung 3 ist das Klassendiagramm zur Übersicht der Darstellungen dargestellt. Die abstrakte Klasse Representation ist eine Konkretisierung eines Bausteins. Eine Darstellung hat einen Zustand(Idle, Running), abhängig davon, ob eine Messung läuft. Der Zustand wird über die Methoden run(), stop(), resume(), verwaltet. Die Darstellung kann auch resettet und gelöscht werden. Außerdem kann man ihre Größe in der UI verändern. Weiter lässt sich eine Darstellung speichern. Die Gesamtmenge der verfügbaren Darstellungen der Anwendung wird das Interface RepresentationManager verwaltet.

Die abstrakte Darstellung wird bisher durch die Klassen XYRepresentation und TableRepresentation implementiert. Bei der Klasse TableRepresentation handelt es sich bisher um die Menge der nicht grafische Darstellungen, die noch konkretisiert werden muss. Die Klasse XYRepresentation repräsentiert die grafische Darstellung von Messwerten. Mögliche Arten der Darstellungen sind durch die Unterklassen „Channel X vs Time“, „Channel X vs Channel Y“ und „N Channels vs Time“ dargestellt. Die Darstellung der Daten in einer XY-Repräsentation(z.B. Balken, Linie oder Bereich) wird durch die Klasse Trace festgelegt. Durch einen Trace wird der Datenstrom von mindestens einem DataProvider dargestellt.

Das eigentliche Plotten der Messwerte wird durch die Bibliothek *Nebula Visualization* durchgeführt.

In Abbildung 4 ist das Sequenzdiagramm zum Hinzufügen einer Darstellungen dargestellt. Dabei sind die Klassen IUserInterface und INotYetDefined bisher nur Lückenfüller, die noch nicht konkret entworfen oder synchronisiert worden sind.

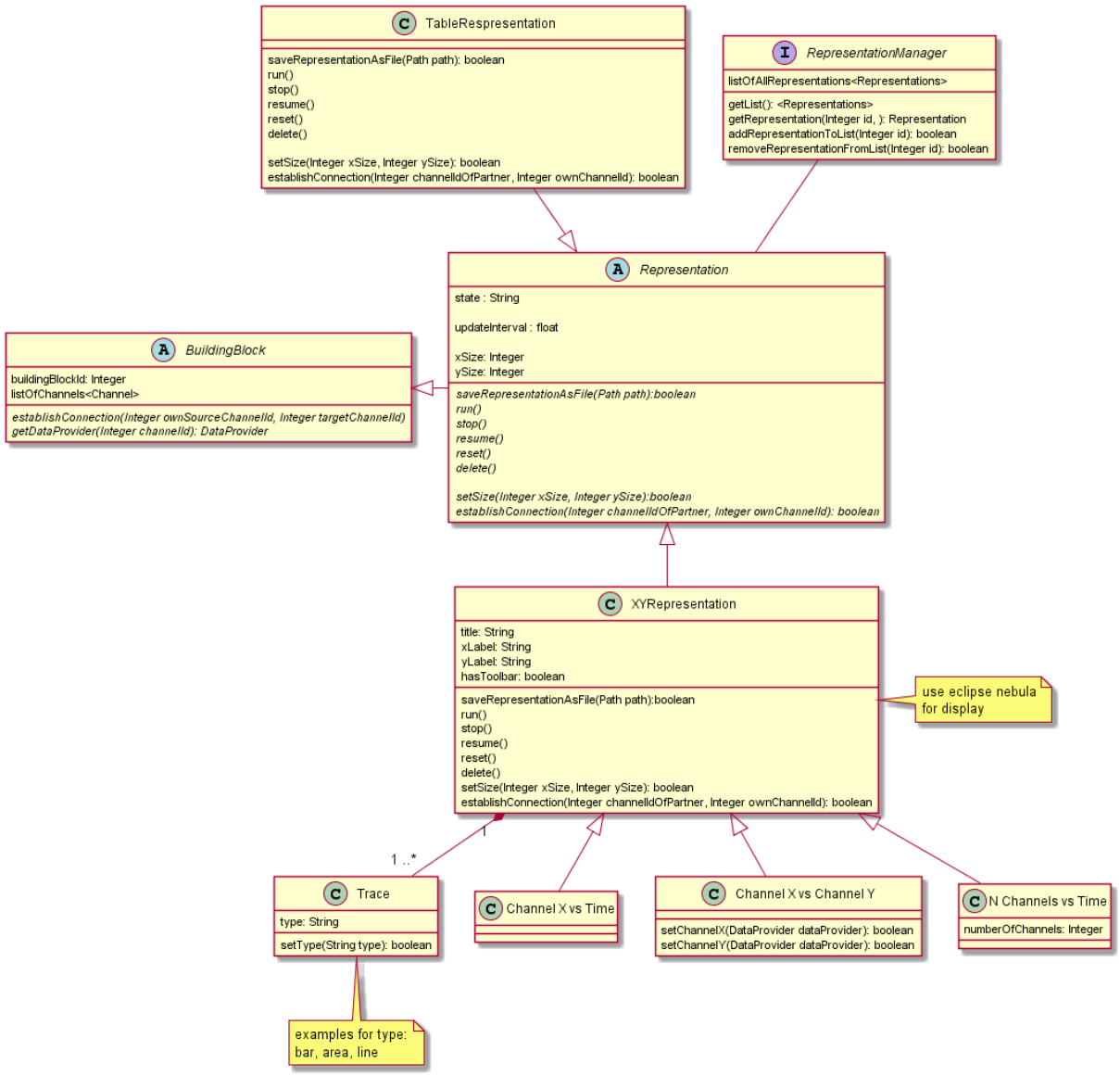

Abbildung 3: Klassendiagramm zur Übersicht der Darstellungen

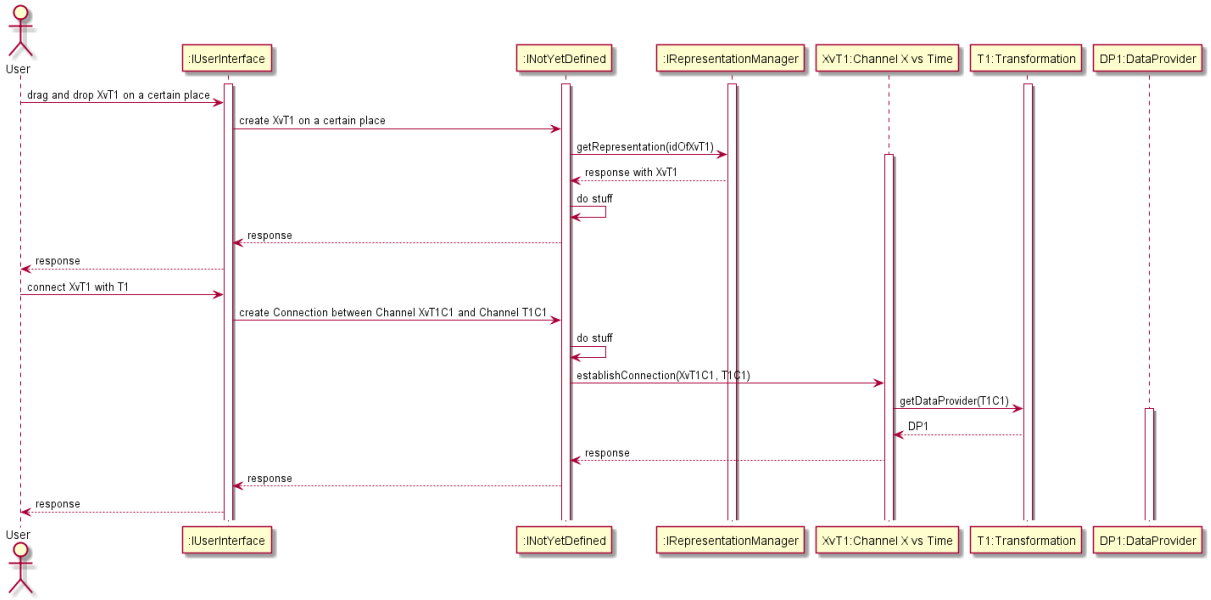

Abbildung 4: Sequenzdiagramm zum Hinzufügen einer Darstellungen

5 Messlauf

In diesem Kapitel wird der zentrale Ablauf des Messens dargestellt.

5.1 Klassenübersicht

5.2 Klassendokumentation

5.3 Sequenzdiagramme

6 Hilfe / Einstellungen

6.1 Ziel

6.2 Klassenübersicht

6.3 Klassendokumentation

6.4 Sequenzdiagramme

7 Glossar

Nebula Visualization Bei Nebula Visualization handelt es sich um eine Widget-Bibliothek zur Visualisierung von Daten. Sie ist über die Eclipse Public License 1.0 verfügbar: https://projects.eclipse.org/content/eclipse-public-license-1.0.