# Architectural representation

## Abstract context overview

In Figure 1 in section 2.1.1, the abstract context overview of the system has already been shown and discussed. Let's have a brief look at it again.

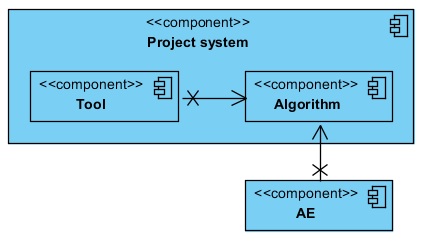


Figure 10 - Abstract context overview of the system

As can be seen in Figure 10, the system consists of two major components. The first one being an analysis tool and the second one being the algorithm library.

The analysis tool will primarily be used to analyze the prediction results of algorithms and retrain them if deemed necessary. This tool uses the algorithm library to call the prediction and (re)train functionality of the algorithm.

The algorithm library offers the functionality to create predictions with an algorithm and (re)train it.

The Analysis Environment (AE) is a system that's currently being used by the company 8vance to scrape and process profile data. The AE will also be using the algorithm library.

*<High level overview of the communication between the systems.>*

## Abstract component overview

### Tool component

In Figure 11, an abstract component overview of the Tool component can be seen. As can be seen, it's a layered architecture consisting of three layers.

The presentation layer contains the MVVM design pattern. The MVVM pattern consists of three components: the View, View-Model and Model. In this case, the View represents the GUI the user interacts with. The View-Model contains the logic to handle events of the View and update the View with data from the Model. The Model contains all the information that's relevant for the View in question.

The business layer contains the core functionality of the system and encapsulates the relevant business logic. This layer will contain the functionality to select algorithms, execute algorithms, train algorithms; in other words: all the user functionality.

The data access layer contains the logic required to access the underlying data stores, such as databases and (configuration) files. It offers a common data access functionality in order to make the application easier to configure and maintain.

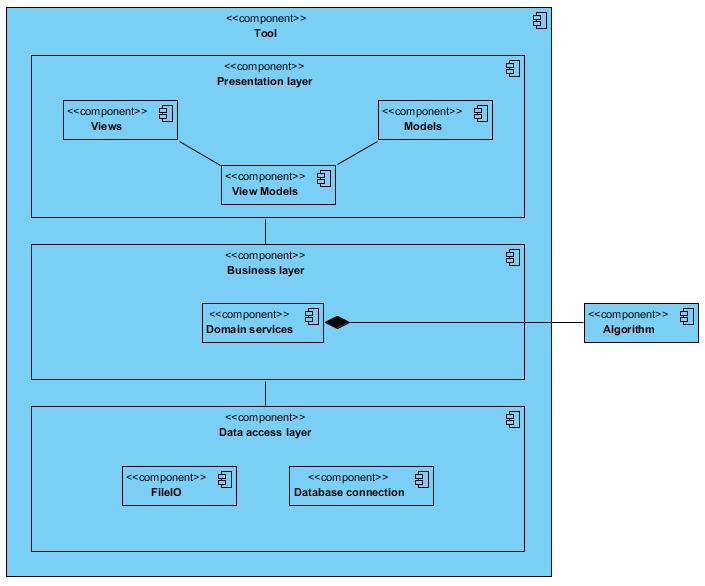


Figure 11 - Abstract component overview for the tool

### Algorithm component

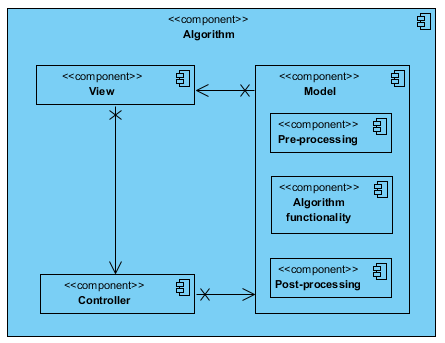


Figure 12 - Abstract component overview for the algorithm library

Figure 12 shows the abstract component overview for the Algorithm library component. The MVC design pattern is implemented in this component.

The View is used as an interface for the input and output data for the algorithm. The View calls operations in the Controller and passes the input data to the Controller.

The Controller is used to call operations in the Model to pre-process the input data received from the View, call algorithm-specific operations with the pre-processed data, and post-process the result of the algorithm-specific operations. So basically, the Controller creates a sort of pipeline that processes all the required steps to get the result the View is asking for.

The Model contains all of the following data-related operations:

* Pre-processing. The Model pre-processes the data by comparing the input data to external data models to find similar data. This similar data is translated to one common interpretation. After this translation, some data may need to be converted to another format to make it usable for the algorithm.
* Algorithm-related functionality. The Model can call the algorithm-related functionality such as making predictions and training the algorithm.
* Post-processing. The Model post-processes the data by converting the necessary data back to the original format.

Depending on the request of the Controller, the Model forwards a response to the View after executing specific algorithm-related functionality (like retraining) or post-processing the data.

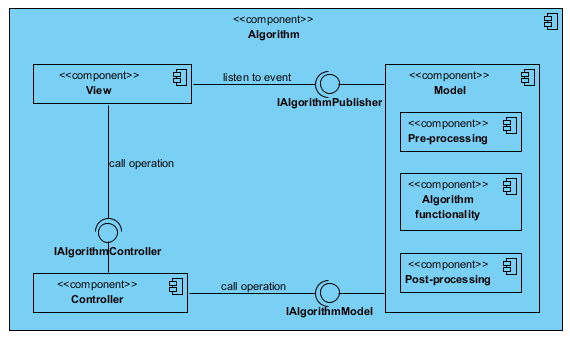
*<High level overview of the important components in the systems.>*

## Detailed component overview

### Tool component

...

### Algorithm component



<Low level overview of the communication between the components of the system.>

## Detailed implementation overview

<Lowest level overview of the implementation of the components of the system (class diagram level).>