Fontys University of Applied Sciences

Design Document for Pipelines

Coen Stange, Edgar Kruze, Wen Li

2015/12/02

Definitions and abbreviations

URS	User Requirements Specification document
SOLID	Single responsibility, Open-closed, Liskov substitution, Interface segregation and Dependency inversion
MVVM	Model View ViewModel
DTO	Data Transfer Object
WPF	Windows Presentation Foundation

Background and Context

This design document specifies the design decisions of the application "Pipelines in a flow network". It is recommended to read the URS first.

Contents

1 Introduction

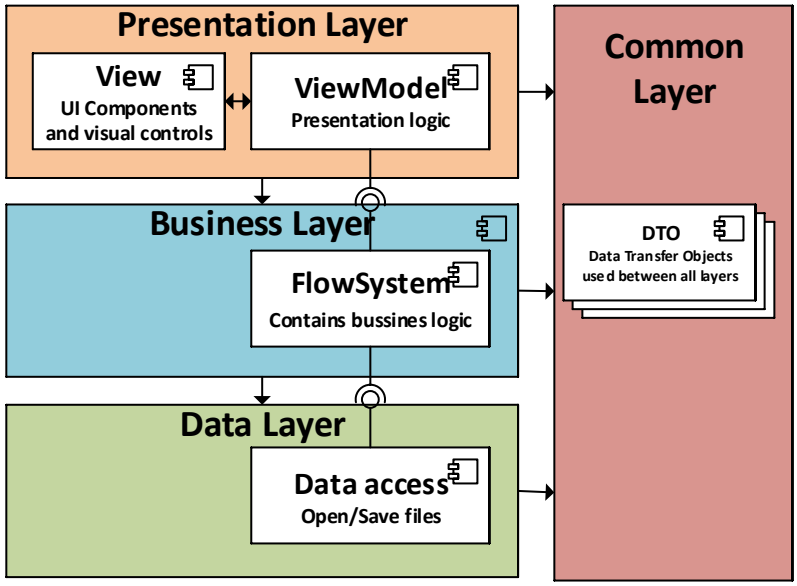

Figure 1.1: Overview of the system.

1.1 Abstraction layers

The system is divided in layers to make the software more modular. For example the "Data Layer" could be replaced to make it work with a Database, for now the application only works with the file system. The components of figure 1.1 know about each other through Dependency Injection whereby the components don't really know

about each other to prevent the code from being glued together. The code is more testable since it is modular.

1.1.1 Common Layer

The Common Layer is referenced by all the other layers. It contains DTO's which are objects transferred between all the layers. The objects only contain data and don't have any behavior, so the objects are really stupid and have no logic.

1.1.2 Presentation Layer

The presentation layer handles the UI of the application, this layer doesn't have any business logic. WPF is used for handling the graphics since it allows an MVVM pattern to be implemented see figure 1.2. The Presentation Layer only contains the View and the ViewModel, the model is in the Bussines Layer,

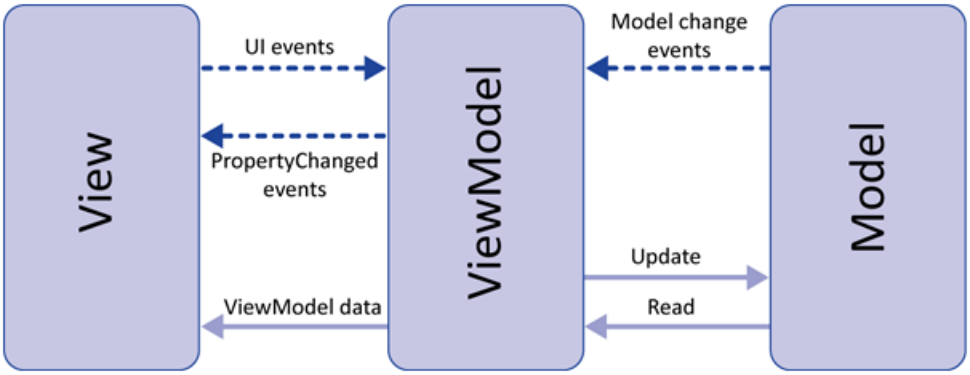

Figure 1.2: Model View ViewModel

1.1.3 Business Layer

The business layer contains the business logic of the program. Business logic include calculating the flow through the network.

1.1.4 Data Layer

The data layer contains the logic to read and write the flow network to a file.

2 | Class diagram

2.1 Common Layer

The class diagram shown in figure 2.2 gives an overview of all the classes (but doesn't show all the relations!). This overview is to clarify that the *FlowNetworkEntity* have a "has-a" relationship with the components (*MergerEntity, PumpEntity...*).

In figure 2.3 the class diagram gives a more detailed view of the components and in more detail how the pipes are connected. Very noticeable are the interfaces *IFlowOutput* and *IFlowInput*. Since a component can have more than 1 input/output an array is used, whereby the pipe also needs to understand to which output it is connected hence the pipe also has the index pipe.

2.2 Presentation Layer

TODO

2.3 Business Layer

TODO

2.4 Data Access layer

Figure 2.1 shows the class diagram of the Data Access layer, the Data Access Layer is small and it could be considered to move this to the business layer. When you open a file you get a dumb model back (objects from common) because otherwise the system is failing to follow Separation of Concerns in SOLID principals.

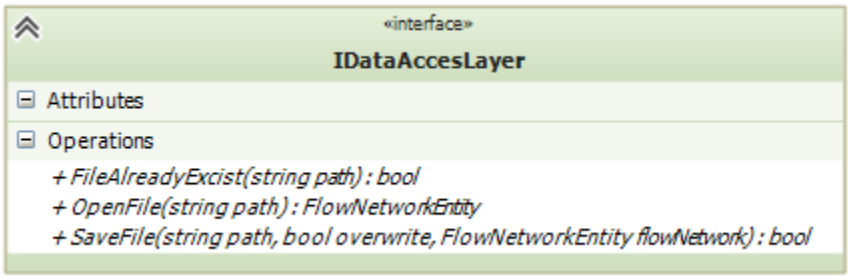

Figure 2.1: Class Diagram Data Access Layer

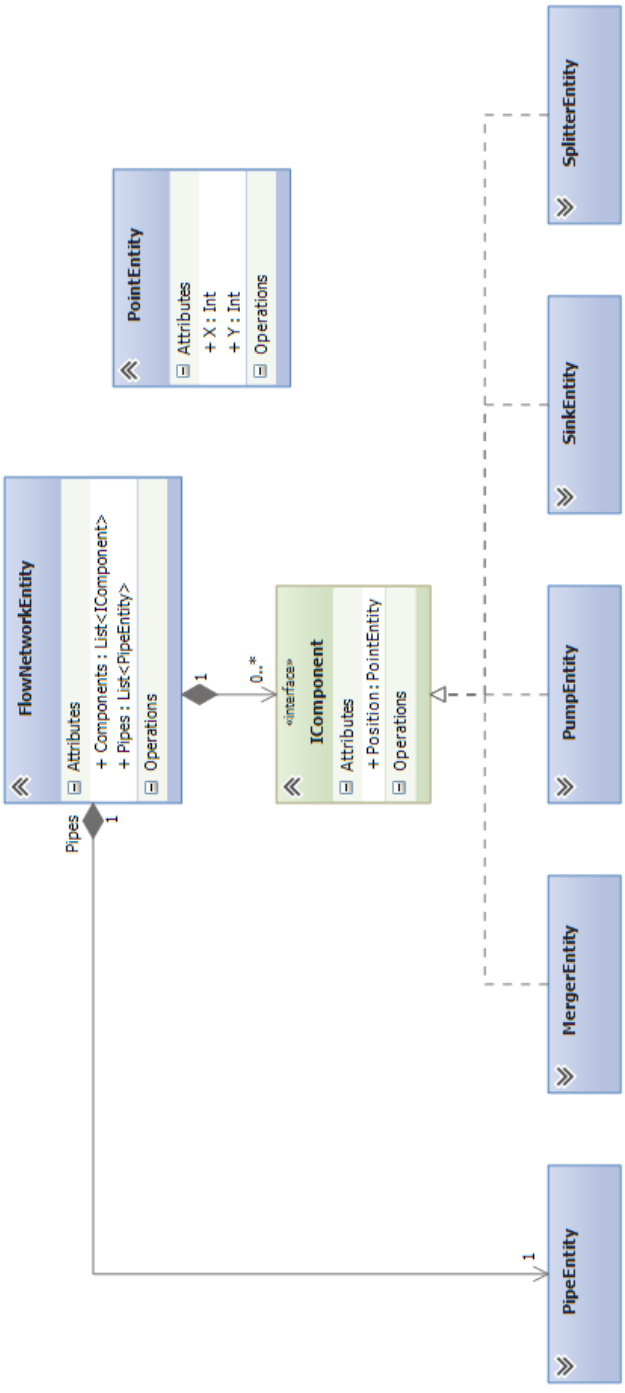

Figure 2.2: Class Diagram Common overview

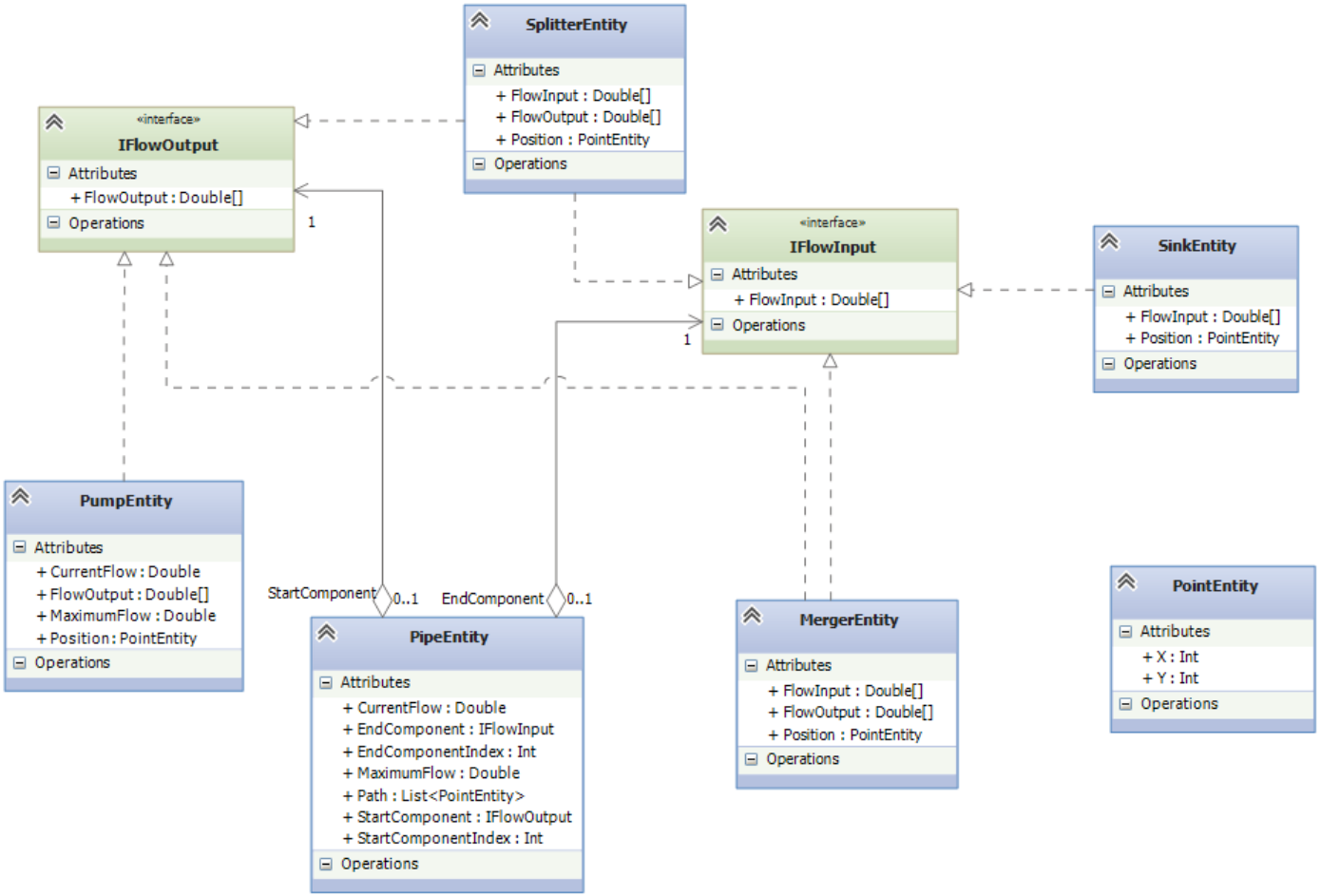

Figure 2.3: Class Diagram Common Components

3 | Sequence diagrams

TODO