Creating a new Wrapper for Odysseus

Although, there is already a large set of wrappers available for Odysseus, there is often the need to create a new specific adapter for a specific source. To ease the creation of new wrapper, we created an adapter framework.

There are two kinds of generic wrapper:

- **GenericPull**: In this approach, data is pulled from the source, i.e. the source is asked if it provides more information and then the information is delivered. A typical example is a file or a web source. The Odysseus scheduler initiates the pull request.
- **GenericPush**: In this approach, the source actively sends the data to Odysseus, so each time new data is available it will directly send through Odysseus. A typical example is a tcp server. No scheduler is needed in this case.

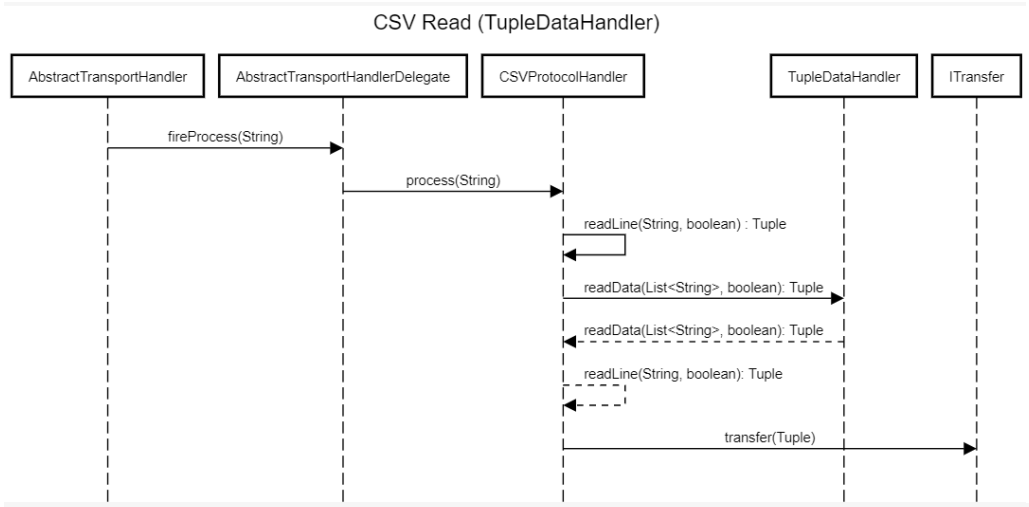
To avoid redundant coding, the wrappers are built from basic elements (handler), each representing a specific function in the wrapping process.

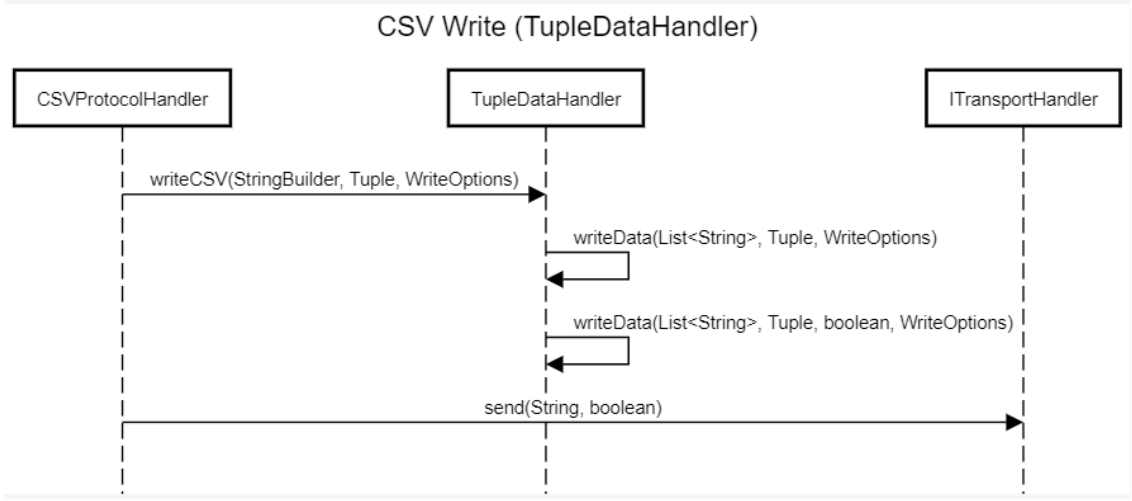
- **Transport**: Die physical bridge between the external systems and Odysseus is given by this handler. It is responsible for the communication, this could be e.g. a file access, a tcp client/server or a message bus handler.
  Hint: There are scenarios in which it is not feasible to separate transport and protocol layer. In such cases, one can implement the combination as a transport handler and use it in combination with the "None" protocol handler.
- **Protocol**: While the transport handler handles the connections, the protocol handler is responsible for the interpretation of the given input coming from the transport handler. I.e. the protocol handler translates the incoming data into a format understandable by Odysseus
- **DataHandler**: Finally, the internal format can be defined. This is the data type, which is used by the internal operators of Odysseus. There are different kind of operators for different kinds of datatypes. DataHandler are e.g. Tuple or KeyValue.

In a source, the information is send from the transport handler, via the protocol handler to the datahandler.

The wrapper can also be used at sink side, i.e. to send data from Odysseus to other systems. In this case, the information is send from the datahandler via the protocol handler to the transport handler. Each handler can provide both ways, i.e. retrieving and sending of data but is not required to. So some handler may only be used in sources, while other only in sinks.

Here are two sequence diagramms that show via an example how the different handler interoperate:





This document explains how to write new wrappers using this generic wrappers.