Thomas Mühlematter

**User API and Examples**

**Overview**

The goal is to have a simple interface that can be used without being an expert at programming. The inspiration for this is the Matlab interface. The user calls functions but is not aware of how the results are computed or how the indexing and clustering is done.

The user interacts with the system using commands. Each command is followed by the reply from the system. This reply can be a message of success or error or the result of a certain function.

**High-level API**

Console

It handles the communications between the user and the system. More precisely, it checks that the inputs are correct and send the user’s commands to the system. It also shows him the messages (success/error,result) sent back by the system.

Operations

Receives the command of the user sent by the console and process them. It communicates with Timeseries,Console,Transformations and Storage.

Timeseries

It regroups all informations pertaining to the timeseries and stores the current temporary data fetched from file or online.

Transformations

It contains all the transformations that can be applied on the time series. I is directly called by the class Operations.

Storage

The I/O is handled by this class. This is the class that is called by operations each team data need to be fetched or stored. This is also the class that handles indexing, clustering and replication.

**Examples (C:command,R:reply)**

Find range of a certain time series

C: T=Timeseries(“id of the timeseries to load”);

R: T created successfully

C: a=range(1,10000);

R: a=5486

C: exit;

Loading a non-existing time series

C: T=Timeseries(“id of a non-existing ts”);

R: The Timeseries “id of a non-existing ts” does not exist

C: exit;

Storing time series

C: T=Timeseries(“id of the timeseries to load”);

R: T created successfully

C: movingAverageSmoother(1,10000,2);

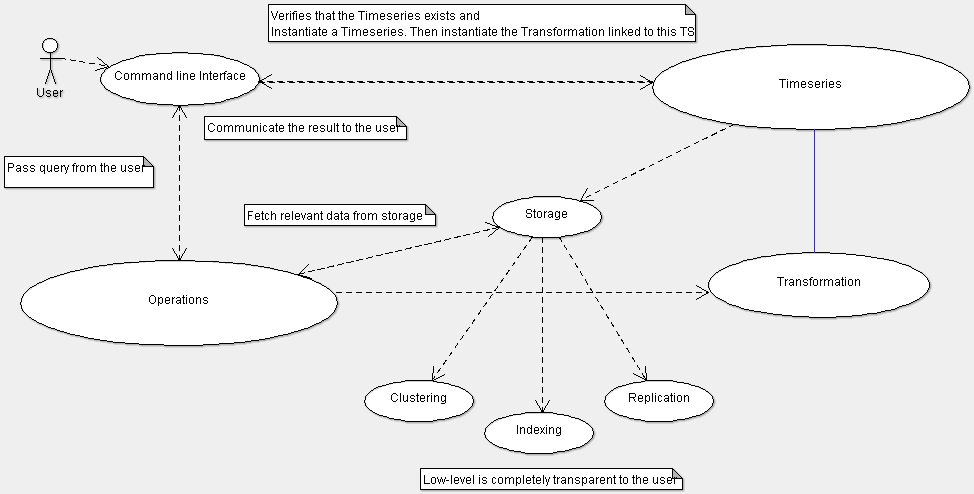
R: moving average smoother applied

C: store(T)

R: T successfully stored

C: exit;

**UML Usage diagram**



**UML Class diagram**