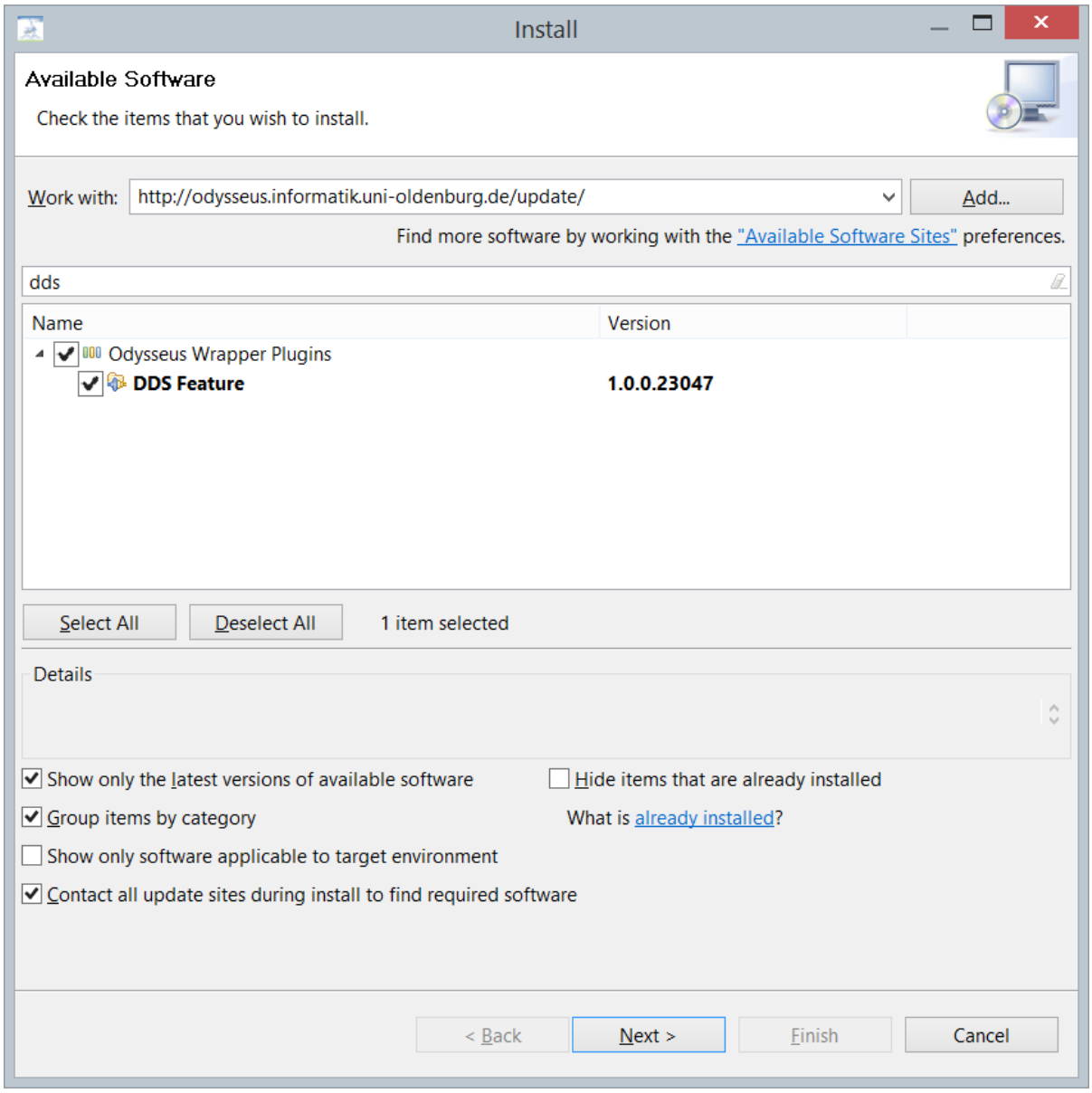
DDS Example with OpenICE

In this example I will show how to connect data from a DDS (Data distribution service for Real-Time Systems). Be aware, that this is very alpha 😉

First of all you have to install the DDS Feature:

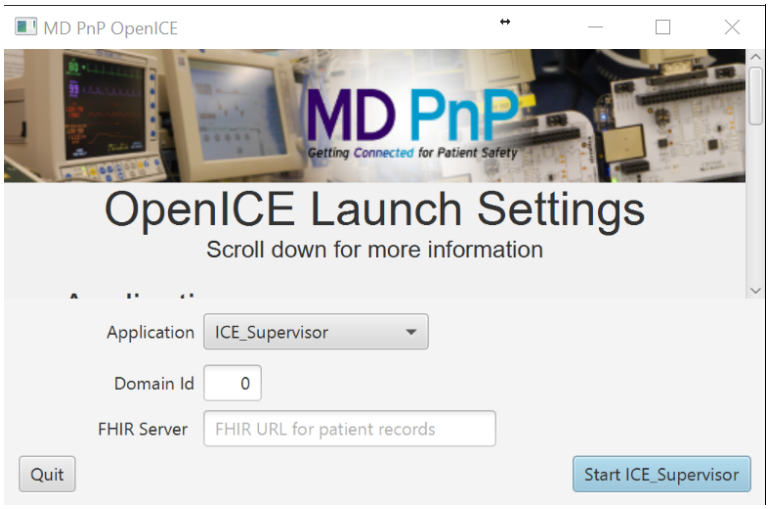


After that, you can download the MD PnP Software from here:

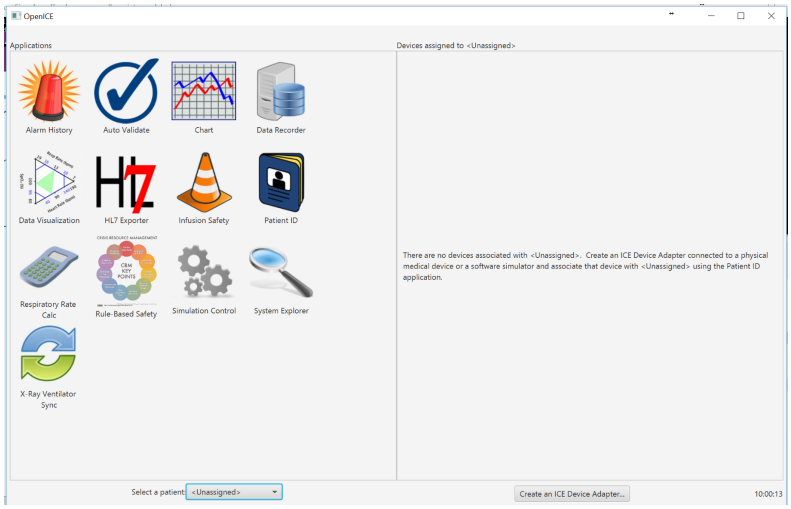
~~http://sourceforge.net/projects/mdpnp/~~

https://github.com/mdpnp/mdpnp

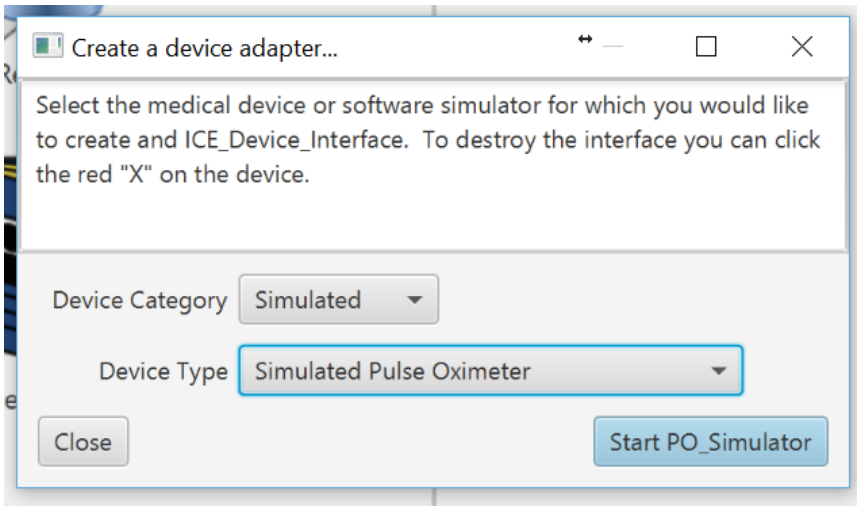
Start the application



Choose Domain Id 0 (or remeber the choosen value) and run "Start ICE_Supervisor".

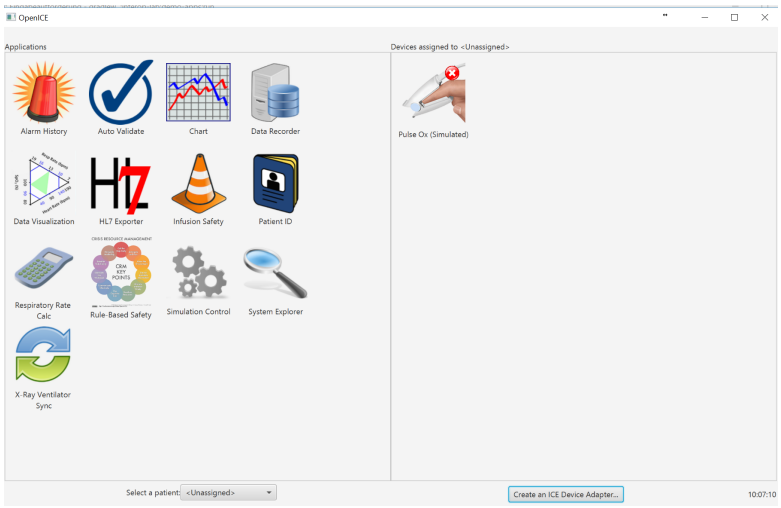


You can now "Create an ICE Device Adapter"

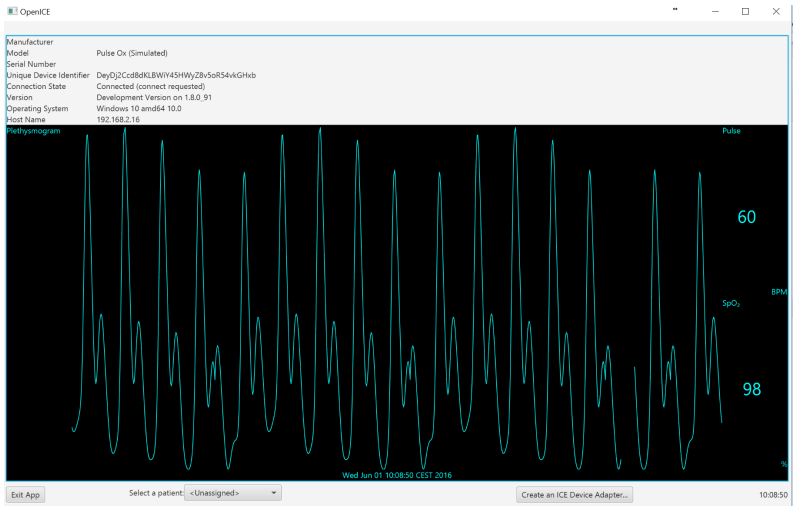


And choose e.g. Simulated Pulse Oximeter. Call "Start Simlulated Pulse Oximeter" and call "Start PO_Simulator".

Now you will get to the main screen:



When you double click on Puls Ox (Simulated) you will see:



Now back to Odysseus:

Create a new Project and add the following script:

```
#PARSER PQL
#RUNQUERY
sampleArray := DDSSOURCE({
                    source = 'SampleArray',
                    qosfile='${WORKSPACEPROJECT}/ice_library.xml',
                    idlfile='${WORKSPACEPROJECT}/ice.idl',
                    topic='SampleArray',
                    topictype='ice::SampleArray',
                    qoslibrary='ice_library',
                    qosprofile='waveform_data',

domain=0

                  }
                )
```

There are two important files:

`USER_QOS_PROFILES.xml`

`ice.idl`

They contain information necessary for DDS. You can download the files from the following locations:

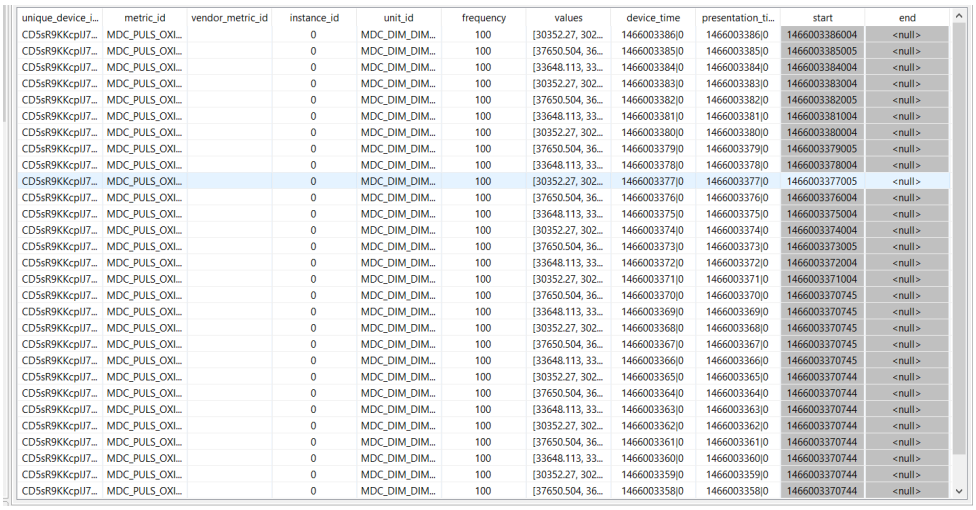
https://github.com/mdpnp/mdpnp/blob/master/data-types/x73-idl/src/main/idl/ice/ice.idl

https://github.com/mdpnp/mdpnp/blob/master/data-types/x73-idl-rti-dds/USER_QOS_PROFILES.xml

Copy them to the workspace.

If you changed the domain, please update domain value in the options above.

Now run the query, and show as table. You should see something like:



unique_device_i...	metric_id	vendor_metric_id	instance_id	unit_id	frequency	values	device_time	presentation_ti...	start	end
CD5sR9KKcpIJ7...	MDC_PULS_OXI...		0	MDC_DIM_DIM...	100	[30352.27, 302...	1466003386|0	1466003386|0	1466003386004	<null>
CD5sR9KKcpIJ7...	MDC_PULS_OXI...		0	MDC_DIM_DIM...	100	[37650.504, 36...	1466003385|0	1466003385|0	1466003385005	<null>
CD5sR9KKcpIJ7...	MDC_PULS_OXI...		0	MDC_DIM_DIM...	100	[33648.113, 33...	1466003384|0	1466003384|0	1466003384004	<null>
CD5sR9KKcpIJ7...	MDC_PULS_OXI...		0	MDC_DIM_DIM...	100	[30352.27, 302...	1466003383|0	1466003383|0	1466003383004	<null>
CD5sR9KKcpIJ7...	MDC_PULS_OXI...		0	MDC_DIM_DIM...	100	[37650.504, 36...	1466003382|0	1466003382|0	1466003382005	<null>
CD5sR9KKcpIJ7...	MDC_PULS_OXI...		0	MDC_DIM_DIM...	100	[33648.113, 33...	1466003381|0	1466003381|0	1466003381004	<null>
CD5sR9KKcpIJ7...	MDC_PULS_OXI...		0	MDC_DIM_DIM...	100	[30352.27, 302...	1466003380|0	1466003380|0	1466003380004	<null>
CD5sR9KKcpIJ7...	MDC_PULS_OXI...		0	MDC_DIM_DIM...	100	[37650.504, 36...	1466003379|0	1466003379|0	1466003379005	<null>
CD5sR9KKcpIJ7...	MDC_PULS_OXI...		0	MDC_DIM_DIM...	100	[33648.113, 33...	1466003378|0	1466003378|0	1466003378004	<null>
CD5sR9KKcpIJ7...	MDC_PULS_OXI...		0	MDC_DIM_DIM...	100	[30352.27, 302...	1466003377|0	1466003377|0	1466003377005	<null>
CD5sR9KKcpIJ7...	MDC_PULS_OXI...		0	MDC_DIM_DIM...	100	[37650.504, 36...	1466003376|0	1466003376|0	1466003376004	<null>
CD5sR9KKcpIJ7...	MDC_PULS_OXI...		0	MDC_DIM_DIM...	100	[33648.113, 33...	1466003375|0	1466003375|0	1466003375004	<null>
CD5sR9KKcpIJ7...	MDC_PULS_OXI...		0	MDC_DIM_DIM...	100	[30352.27, 302...	1466003374|0	1466003374|0	1466003374004	<null>
CD5sR9KKcpIJ7...	MDC_PULS_OXI...		0	MDC_DIM_DIM...	100	[37650.504, 36...	1466003373|0	1466003373|0	1466003373005	<null>
CD5sR9KKcpIJ7...	MDC_PULS_OXI...		0	MDC_DIM_DIM...	100	[33648.113, 33...	1466003372|0	1466003372|0	1466003372004	<null>
CD5sR9KKcpIJ7...	MDC_PULS_OXI...		0	MDC_DIM_DIM...	100	[30352.27, 302...	1466003371|0	1466003371|0	1466003371004	<null>
CD5sR9KKcpIJ7...	MDC_PULS_OXI...		0	MDC_DIM_DIM...	100	[37650.504, 36...	1466003370|0	1466003370|0	1466003370745	<null>
CD5sR9KKcpIJ7...	MDC_PULS_OXI...		0	MDC_DIM_DIM...	100	[33648.113, 33...	1466003369|0	1466003369|0	1466003370745	<null>
CD5sR9KKcpIJ7...	MDC_PULS_OXI...		0	MDC_DIM_DIM...	100	[30352.27, 302...	1466003368|0	1466003368|0	1466003370745	<null>
CD5sR9KKcpIJ7...	MDC_PULS_OXI...		0	MDC_DIM_DIM...	100	[37650.504, 36...	1466003367|0	1466003367|0	1466003370745	<null>
CD5sR9KKcpIJ7...	MDC_PULS_OXI...		0	MDC_DIM_DIM...	100	[33648.113, 33...	1466003366|0	1466003366|0	1466003370745	<null>
CD5sR9KKcpIJ7...	MDC_PULS_OXI...		0	MDC_DIM_DIM...	100	[30352.27, 302...	1466003365|0	1466003365|0	1466003370744	<null>
CD5sR9KKcpIJ7...	MDC_PULS_OXI...		0	MDC_DIM_DIM...	100	[37650.504, 36...	1466003364|0	1466003364|0	1466003370744	<null>
CD5sR9KKcpIJ7...	MDC_PULS_OXI...		0	MDC_DIM_DIM...	100	[33648.113, 33...	1466003363|0	1466003363|0	1466003370744	<null>
CD5sR9KKcpIJ7...	MDC_PULS_OXI...		0	MDC_DIM_DIM...	100	[30352.27, 302...	1466003362|0	1466003362|0	1466003370744	<null>
CD5sR9KKcpIJ7...	MDC_PULS_OXI...		0	MDC_DIM_DIM...	100	[37650.504, 36...	1466003361|0	1466003361|0	1466003370744	<null>
CD5sR9KKcpIJ7...	MDC_PULS_OXI...		0	MDC_DIM_DIM...	100	[33648.113, 33...	1466003360|0	1466003360|0	1466003370744	<null>
CD5sR9KKcpIJ7...	MDC_PULS_OXI...		0	MDC_DIM_DIM...	100	[30352.27, 302...	1466003359|0	1466003359|0	1466003370744	<null>
CD5sR9KKcpIJ7...	MDC_PULS_OXI...		0	MDC_DIM_DIM...	100	[37650.504, 36...	1466003358|0	1466003358|0	1466003370744	<null>

As you can see, the values attribute contains a list, and the time attributes a tuple.

Storing to a relational database

If you want to store this data in a database this is not possible directly. You have to "flatten" the data. This can be done as follows:

```
mappedValues = MAP({
                    expressions = [
                     'unique_device_identifier',
                     ['elementAt(time,0)','seconds'],
                     ['elementAt(time,1)','nanoseconds'],
                     'values'
                    ]
                  },
                  sampleArray
                )
table1 = UNNEST({
              attribute = 'values'
            },
            mappedValues
          )

table1_WithNumbers = MAP({
                          expressions = [
                              'unique_device_identifier',
                              'seconds',
                              'nanoseconds',
                              ['counter()','pos'],
                              ['values','value']
                               ]
                        },
                        table1
                      )
table2 = MAP({
              expressions = [
                'unique_device_identifier',
                ['elementAt(time,0)','seconds'],
                ['elementAt(time,1)','nanoseconds'],
                'metric_id',
                'instance_id',
                'unit_id',
                'frequency'
              ]
            },
            sampleArray
          )
```

table1_WithNumbers is only necessary if you want to keep the order of the measurements.

The output will be as:

unique_device_identifier	seconds	nanoseconds	pos	value	Metadata
CIMZHWsBALPediHAqjXB9paW...	1420625250	0	2901	33970.785	1420625250003|oo
CIMZHWsBALPediHAqjXB9paW...	1420625250	0	2900	33648.113	1420625250003|oo
CIMZHWsBALPediHAqjXB9paW...	1420625249	0	2899	33228.88	1420625249004|oo
CIMZHWsBALPediHAqjXB9paW...	1420625249	0	2898	32764.953	1420625249004|oo
CIMZHWsBALPediHAqjXB9paW...	1420625249	0	2897	32337.092	1420625249004|oo
CIMZHWsBALPediHAqjXB9paW...	1420625249	0	2896	32039.045	1420625249004|oo
CIMZHWsBALPediHAqjXB9paW...	1420625249	0	2895	31951.002	1420625249004|oo
CIMZHWsBALPediHAqjXB9paW...	1420625249	0	2894	32121.809	1420625249004|oo
CIMZHWsBALPediHAqjXB9paW...	1420625249	0	2893	32551.34	1420625249004|oo
CIMZHWsBALPediHAqjXB9paW...	1420625249	0	2892	33194.156	1420625249004|oo
CIMZHWsBALPediHAqjXB9paW...	1420625249	0	2891	33980.664	1420625249004|oo
CIMZHWsBALPediHAqjXB9paW...	1420625249	0	2890	34844.254	1420625249004|oo
CIMZHWsBALPediHAqjXB9paW...	1420625249	0	2889	35742.566	1420625249004|oo
CIMZHWsBALPediHAqjXB9paW...	1420625249	0	2888	36651.773	1420625249004|oo
CIMZHWsBALPediHAqjXB9paW...	1420625249	0	2887	37568.79	1420625249004|oo
CIMZHWsBALPediHAqjXB9paW...	1420625249	0	2886	38489.113	1420625249004|oo
CIMZHWsBALPediHAqjXB9paW...	1420625249	0	2885	39390.277	1420625249004|oo

and

unique_device_identifier	seconds	nanoseconds	metric_id	instance_id	unit_id	frequency	Metadata
CIMZHWsBALPediHAqjX...	1420625252	0	MDC_PULS_OXIM_PLETH	0	MDC_DIM_DIMLESS	100	1420625252003|oo
CIMZHWsBALPediHAqjX...	1420625251	0	MDC_PULS_OXIM_PLETH	0	MDC_DIM_DIMLESS	100	1420625251002|oo
CIMZHWsBALPediHAqjX...	1420625250	0	MDC_PULS_OXIM_PLETH	0	MDC_DIM_DIMLESS	100	1420625250003|oo
CIMZHWsBALPediHAqjX...	1420625249	0	MDC_PULS_OXIM_PLETH	0	MDC_DIM_DIMLESS	100	1420625249004|oo
CIMZHWsBALPediHAqjX...	1420625248	0	MDC_PULS_OXIM_PLETH	0	MDC_DIM_DIMLESS	100	1420625248003|oo
CIMZHWsBALPediHAqjX...	1420625247	0	MDC_PULS_OXIM_PLETH	0	MDC_DIM_DIMLESS	100	1420625247002|oo
CIMZHWsBALPediHAqjX...	1420625246	0	MDC_PULS_OXIM_PLETH	0	MDC_DIM_DIMLESS	100	1420625246003|oo
CIMZHWsBALPediHAqjX...	1420625245	0	MDC_PULS_OXIM_PLETH	0	MDC_DIM_DIMLESS	100	1420625245003|oo

Storing to a document store

You can store complex objects to a document store like MongoDB. For this, you have to convert the tuple to a KeyValueObject. This can be done with the TupleToKeyValueOperator.

See MongoDB Feature for more information on the Mongo DB feature.

```
#RUNQUERY
kv = TUPLETOKEYVALUE(sampleArray)
mongo = MONGODBSINK({PORT = port, HOST = 'host', COLLECTIONNAME = 'collectionname', DATABASE = 'database'}, kv)
```