Using RRD4J with OPENREMOTE

This tutorial aims to give some more information about the parameters that are needed to setup a Round Robin Database (RRD) for monitoring, logging and plotting data coming from Openremote (OR) sensors.

I. THE XML CONFIGURATION FILE

The config file to set up a RRD on the OR Controller can be found under:

./webapps/controller/rrd / rrd4j-config.xml

In this file some parameters need to be given appropriate values in order to get the RRD properly functioning.

I.1. Configuration of the actual database

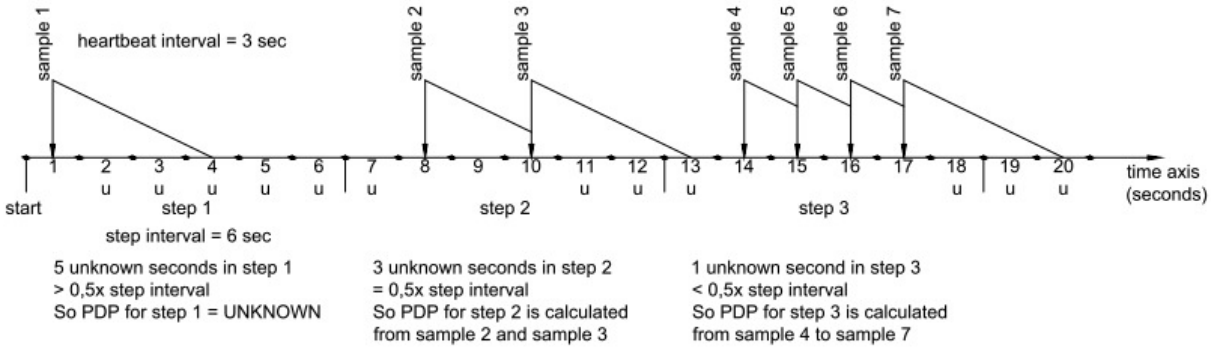
```
<rrdDB fileName = "test.rrd" step = "60">
  <datasource name = "Temp1Sensor" type = "GAUGE" heartbeat = "120" />
  <datasource name = "Weather sensor" type ="GAUGE" heartbeat = "120" />
      <archive function = "AVERAGE" xff = "0.5" steps = "1" rows = "300" />
      <archive function = "AVERAGE" xff = "0.5" steps = "10" rows ="144" />
      <archive function = "AVERAGE" xff = "0.5" steps = "15" rows = "672" />
      <archive function = "AVERAGE" xff = "0.5" steps = "60" rows = "744" />
      <archive function = "AVERAGE" xff = "0.5" steps = "120" rows = "4380" />
</rrdDB>
```

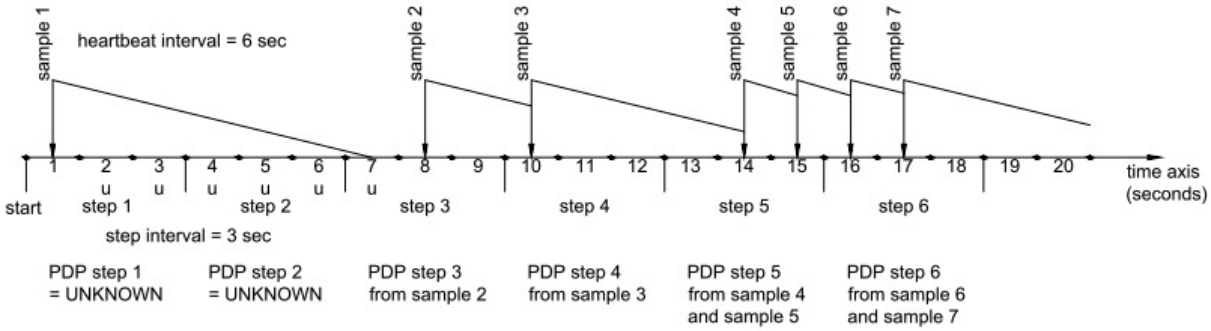
filename	Name of the database file including the file extension *".rrd "*.
step	Specifies the base interval in seconds for which a **Primary Data Point** (PDP) is calculated from the values (samples) that are sent to RRD.
datasource name	Name of the sensor defined in OR Designer wh values are sent to the RRD. **It must be 1 to maximum 19 characters long and only containing alphanumeric characters or "_" and "-".**

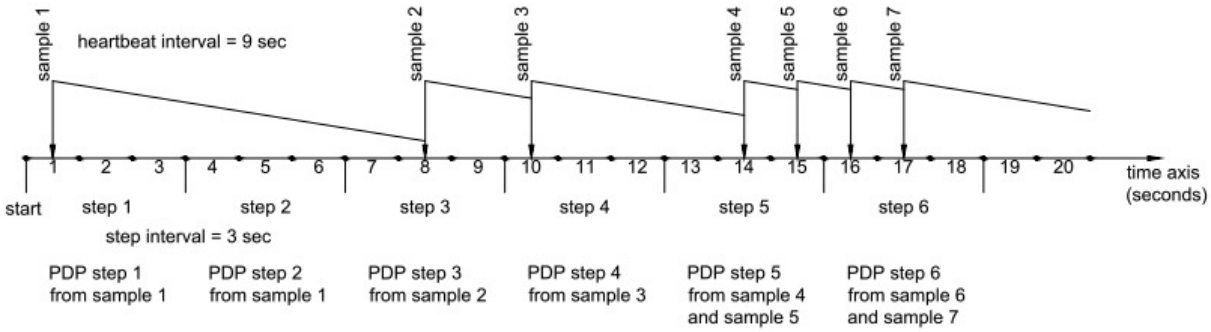
type	Possible datasource types (DST) are: ■ COUNTER **The rate of change** of the value is calculated (the difference between the current value and the previous value divided by the time interval between these two values). The values must always increasing (i.e. the difference between the current and the previous value is greater than 0). ■ DERIVE Same as COUNTER, but negative values are allowed as well. ■ ABSOLUTE Also calculates the rate of change, but assumes that the previous value is set to 0. Thus the current value divided by the time interval is calculated. ■ GAUGE The actual value stays unchanged.
heartbeat	Defines the maximum acceptable interval in seconds between two value sendings (samples). If the interval between two samples is greater than the heartbeat interval, that entire time interval will be considered as being "UNKNOWN". The samples received are used to calculate an average rate (or value in case of GAUGE), called the " **Primary Data Point**" (PDP). For each step interval a PDP is calculated. However, if the total "UNKNOWN" time within a step interval accounts for more than half the step interval, the PDP for that step interval will also be marked as "UNKNOWN" (which means there weren't enough samples to calculate an appropriate PDP for that step interval).

	The heartbeat interval can be shorter or longer than the step interval. Usually a longer interval is chosen. A heartbeat interval shorter than the step interval means that multiple samples per PDP are required. If a heartbeat interval spans multiple step intervals, multiple PDP's can be calculated from a single sample. The figures on the next page explain some more.
archive *function* *steps*	*function* defines a "consolidation function" (CF). Multiple PDP's can be consolidated into one "**Consolidated Data Point**" (CDP). It is the CDP (not the PDP) that is stored in an *archive* (RRA) within the database. The CF can be AVERAGE, MINIMUM, MAXIMUM or LAST. One CDP is consolidated from *steps* number of PDP's. The parameter *steps* together with the step interval of the PDP determines the **resolution** or "granularity" of the RRA. For example, if step interval = 180 sec and steps is = 5, the RRA will have a resolution of 900 sec (15 minutes), which means one CDP spans 900 sec or in other words 1 CDP is 900 sec wide. ■ AVERAGE The average of *steps* number of PDP's is calculated and stored as one CDP in the archive. ■ MIN or MAX The minimum / maximum value of *steps* number of PDP's is stored as a CDP in the archive. ■ LAST The last value of *steps* number of PDP's is stored as a CDP in the archive.
rows	Defines the size of the archive (i.e. the number of CDP's in the archive). This size is fixed. When the archive is full, the oldest CDP's are overwritten with new ones (= circular database, FIFO).

xff	*xfiles factor* : the ratio of allowed "UNKNOWN" PDP's to the *steps* number of PDP's. If the number of "UNKNOWN" PDP's in a given consolidation interval is greater than *xff* multiplied with *steps* number of PDP's the CDP will get the value "UNKNOWN". If it is smaller, the "UNKNOWN" PDP's will be ignored.







In the configuration file by default two datasources of type GAUGE are already declared. Of course, the names of these datasources needs to be modified to the names of the OR sensors you want to monitor in your own project.

Make sure the names of the OR sensors you are going to use satisfy the naming resistrictions. Also note that the values that are sent to the RRD need to be numerical.

If you only need one datasource, you can delete one in the configuration file; of course, you can also add more datasources if needed.

By default the step interval in the configuration file is set at 60 seconds and the heartbeat interval at 120 seconds. This means that the OR sensor needs to be updated at least every 120 seconds. You might need to modify *step* and *heartbeat* settings according to your specific OR sensors, otherwise you might only get UNKNOWN-values into your database.

Five archives (RRA's) are declared in the configuration file by default. Note that each datasource will have its own 5 archives.

In the first archive *steps* = 1, so only one PDP gets averaged (so CDP = PDP and the resolution of the RRA equals the step interval, here being 60 seconds). 300 CDP's can be stored in this archive. The archive will be full after 300 x 60 seconds = 5 hours. So the time span of the first archive is 5 hours.

In the second archive 10 PDP's are averaged into one CDP. With the default step interval of 60 seconds one CDP is thus written every 600 seconds or every 10 minutes (so the resolution of the RRA is 600 seconds or 10 minutes). 144 CDP's can be stored. This means that the archive will be full after 144 x 10 minutes = 1440 minutes or 24 hours (1 day). So the time span of the second archive is 1 day.

Of course, you may modify the number of RRA's and their consolidation settings to suit your own needs.

I.2 Plotting configuration: options and "plot sources"

```
<rrd_graph_def name="graph1">
    <options>
        <vertical_label>pmeT</vertical_label>
        <title>OpenRemote RRD4J Example</title>
    </options>
    <datasources>
        <def>
            <name>Temp1</name>
            <rrd>test.rrd</rrd>
            <source>Temp1Sensor</source>
            <cf>AVERAGE</cf>
        </def>
        <def>
            <name>Temp2</name>
            <rrd>test.rrd</rrd>
            <source>Weather sensor</source>
            <cf>AVERAGE</cf>
        </def>
    </datasources>
```

name	the name of the graph that will be called from OR.
<*options*>	options for constructing the graph image. For a complete list of available options and their meaning see the documentation on RRD4J and RRDtool: *rrd4j.googlecode.com/git/javadoc/index.html* *oss.oetiker.ch/rrdtool*
<*datasources*>	Before any data can be plotted on a graph, there must be a link to a datasource (<*source*>) in a RRD-file (<*rrd*>) to fetch data from. In one graph you can plot from multiple datasources located in different RRD-files. Under datasources you actually define the "plot sources" that will be plotted on the graph. Each "plot source" needs to be identified by a name (<*name*>).

<*cf*>	Specifies the type of RRA to be used from the given datasource according to the CF assigned to the RRA's. For example, the same datasource in a RRD can have several RRA's of type AVERAGE and several RRA's of type MAX.

I.3 Define the layout of the plots in the graph

```
  <graph>
        <line>
            <datasource>Temp1</datasource>
            <color>#FF0000</color>
            <legend>1-Wire Temp\r</legend>
            <width>2</width>
        </line>
        <line>
            <datasource>Temp2</datasource>
            <color>#0000FF</color>
            <legend>Google Weather Temp\r</legend>
            <width>2</width>
        </line>
        <comment>This chart displays OR sensor values.</comment>
    </graph>
 </rrd_graph_def>
</rrd4j>
```

<*datasource*>	Refers in fact to the "plot source" (see I.2), not to the real data-source in the database.

Refer to the primary documentation on RRD4J and RRDtool [1] for more information about all the layout possibilities.

---

[1] *rrd4j.googlecode.com /git/javadoc/index.html*
oss.oetiker.ch/rrdtool

Once the configuration file is saved and the OR Controller is started up again, the RRD will be automatically created and started.

When datasources are modified or added to the configuration file the old rrd-file (which resides in the same directory of the configuration file) must be deleted or removed before restarting the OR Controller.

II. OPENREMOTE ONLINE DESIGNER

The graph can be displayed using a Webview Widget (you therefore will need OR Online Designer 2.14.0). A sensor needs to be attached to this Webview Widget which contains the URL to the image of the graph. To get this URL into the sensor you will first need to create a command using the "RRD4J Graph URL"-protocol that has the following parameters:

graph name	the name of the graph as given in the configuration file
width	the width of the canvas (the region of the image displaying the actual plots) in pixels.
height	the height of the canvas in pixels.
start time	the start of the time series you would like to display in the graph
end time	the end of the time series you would like to display in the graph
command	getUrl

Start time and *end time* need to be entered in the format *yyyymmdd-hh-mm*.

End time minus *start time* determines the time span of the graph. The *width* of the canvas determines the number of pixel columns in the graph. Dividing the time span (expressed in seconds) by the number of pixel columns, tells us how wide in seconds one pixel column will be: this determines the **resolution of the graph**.

By default RRD calculates the resolution of the graph and tries to get data from an RRA with that same resolution. For example, when the time span is 1 day (86400 sec) and the width of the canvas is 600 pixels, then one pixel column is 144 seconds wide. One pixel column can display only one CDP. If we have in our RRD an RRA with a resolution of 144 seconds per CDP (one CDP is 144 sec wide), then RRD will pick this RRA for displaying the graph, without the need for any further consolidation.

Suppose we have an RRA with a time span of one day (86400 sec) and a resolution of 600 seconds per CDP. We want to display this RRA without loss of information (i.e. without further consolidation). How wide should we choose our graph at least?
One pixel column can display only one CDP. We have:

86400 sec / (600 sec / CDP) = 144 CDP's

in our RRA. So the width of the graph should be at least 144 pixels. Note that this is also the value of *rows* in our RRA. If we choose a width of 432 pixels (3 times 144 pixels), there will be 3 pixel columns for displaying one CDP.

Suppose we also have an RRA with a time span of one week (604800 sec) and the same resolution of 600 seconds per CDP. We would like to display this RRA without loss of information (i.e. without further consolidation). How wide should we choose our graph at least?
One pixel column can display only one CDP. We now have:

604800 sec / (600 sec / CDP) = 1008 CDP's

in our RRA. So the width of the graph should be at least 1008 pixels. If this is not possible, further consolidation will be done by RRD.
Suppose we choose the width to be 504 pixels (1008 / 2) then each two CDP's in our RRA will be further consolidated into one CDP for displaying.

Suppose we have a graph with an available width of 600 pixel columns and a RRA with a resolution of 600 seconds per CDP (one CDP is 600 seconds wide). One pixel column can display only one CDP. So one pixel column must also be 600 seconds wide. What time span should we choose for our graph?
We have 600 pixels columns available, so we can choose a time span for our graph of 600 pixel columns × (600 sec / column) = 360.000 seconds (100 hours) without further consolidation to happen.
Should we choose a time span of 500 hours (1.800.000 sec) then the pixel columns would become 3000 seconds wide (1.800.000 sec / 600 pixel columns). So one pixel column would contain 5 CDP's (3000 seconds / 600 seconds per CDP). Because this is not possible, RRD will further consolidate each 5 CDP's in the RRA into one CDP for displaying.

By default the graph will be automatically scaled so that the y-axis is adjusted to the range of the plotted data. This can however be modified by setting the right options in the XML configuration file.

References:

1. *How To – Creating and Displaying graphs with RRD4J "*
www.openremote.org/pages/viewpage.action?pageId=19988490

2. The documentation about RRD4J
rrd4j.googlecode.com /git/javadoc/index.html

3. The documentation about RRDtool
oss.oetiker.ch/rrdtool