

WebCA

User manual

Revision: 1.0

Status: Draft

Project: WebCa

Document ID: CSL-DOC-07-34340

File: webca.doc

Owner: msekoranja

Last modification: July 14, 2008

Created: December 6, 2007



Document History

Revision	Date	Changed/ reviewed	Section(s)	Modification
0.1	2007-12-06	gjansa	All	Created.
0.2	2008-09-01	gjansa	All	Created/Modified.
0.3	2008-10-01	gjansa	1	Web browsers version added.
0.4	2008-25-01	gjansa	5.17	Changed attributes and elements names.
0.5	2008-13-02	gjansa	All	Updated.
1.0	2008-14-07	msekoranja	All	Released.

Confidentiality

This document is classified as a **public document**. As such, it or parts thereof are openly accessible to anyone listed in the Audience section, either in electronic or in any other form.

Scope

This document covers installation and usage of WebCA. It also covers how new components can be added to WebCA.

Audience

Users creating EPICS Channel Access clients using WebCA.

Typography

This document uses the following styles:



A box like this contains important information.



Warning!
A box like this provides information, which should not be disregarded!



Glossary of Terms

References

- [1] NPCA plugin api documentation, Cosylab, 2007 (<http://webca.cosylab.com/>)
- [2] WebCA design document, Cosylab, 2007
- [3] Extensible Markup Language (XML) <http://www.w3.org/XML/>
- [4] XML Schema <http://www.w3.org/XML/Schema>
- [5] XSL Transformations, <http://www.w3.org/TR/xslt>
- [6] The Yahoo! User Interface Library (YUI) <http://developer.yahoo.com/yui/>

Table of Contents

1. Introduction	7
2. Simple example	8
3. Context menu	9
4. Number format	10
5. Macro substitution	11
6. Components	12
6.1. CAML attributes	12
6.2. Standard component attributes	13
6.3. Layout components	13
6.3.1. HorizontalPanel.....	13
6.3.2. VerticalPanel.....	14
6.3.3. Column.....	15
6.3.4. Row	16
6.3.5. Panel.....	17
6.4. TabView.....	18
6.5. Image.....	19
6.6. StaticText.....	20
6.7. Mux.....	21
6.8. TextUpdate	23
6.9. TextEntry	23
6.10. Message Button.....	24
6.11. MenuButton	25
6.12. RadioButton	27
6.13. WheelSwitch	28
6.14. Slider	29
6.15. Gauge	29
6.16. RelatedDisplay	30
6.17. X-Y Chart.....	32
6.18. bar chart	33
6.19. Intensity plot	35
7. Complex example	37
8. Adding new component to WebCA	38

Figures

Figure 1: WebCA.....	7
Figure 2: Context menu example	9
Figure 3: PV inspector example	9
Figure 4: Number format examples.....	10
Figure 5: Complex layout example.....	18



Figure 6: TabView example	19
Figure 7: StaticText example	21
Figure 8: Mux component; left in Safari, right in Firefox.....	22
Figure 9: TextUpdate example	23
Figure 10: TextEntry example	24
Figure 11: MessageButton example	25
Figure 12: MenuButton example	26
Figure 13: RadioButton example.....	28
Figure 14: WheelSwitch example.....	28
Figure 15: Slider example	29
Figure 16: Gauge example.....	30
Figure 17: X-Y chart example	33
Figure 18: Bar Chart.....	35
Figure 19: Intensity plot.....	36
Figure 20: Example device control panel	37

Tables

Table 1: Supported browsers	7
Table 2: CAML object attributes.....	12
Table 3: Standard component attributes	13
Table 4: HorizontalPanel attributes	14
Table 5: HorizontalPanel standard attributes	14
Table 6: HorizontalPanel elements	14
Table 7: VerticalPanel attributes	15
Table 8: VerticalPanel standard attributes	15
Table 9: VerticalPanel elements	15
Table 10: Column attributes	16
Table 11: Column standard attributes	16
Table 12: Row attributes	16
Table 13: Row standard attributes	16
Table 14: Panel attributes	17
Table 15: Panel standard attributes	17
Table 16: Panel elements	17
Table 17: TabView attributes	18
Table 18: TabView standard attributes	19
Table 19: TabView elements.....	19
Table 20: Tab attributes	19
Table 21: Image attributes	20
Table 22: Image standard attributes	20
Table 23: StaticText attributes	20
Table 24: StaticText standard attributes	20
Table 25: Mux standard attributes.....	21
Table 26: Mux elements	22
Table 27: Sequence attributes	22
Table 28: Sequence elements	22
Table 29: MacroValuePair attributes	22
Table 30: TextUpdate attributes.....	23
Table 31: TextUpdate standard attributes	23
Table 32: TextEntry attributes	24
Table 33: TextEntry standard attributes	24
Table 34: MessageButton standard attributes	25
Table 35: MessageButton elements	25
Table 36: On and off attributes.....	25
Table 37: MenuButton standard attributes	26
Table 38: MenuButton elements	26



Table 39: MenuItem attributes	26
Table 40: RadioButton attributes	27
Table 41: RadioButton standard attributes	27
Table 42: RadioButton elements	27
Table 43: RadioButtonItem attributes	28
Table 44: WheelSwitch standard attributes	28
Table 45: Slider attributes	29
Table 46: Slider standard attributes	29
Table 47: Gauge attributes	30
Table 48: Gauge standard attributes	30
Table 49: RelatedDisplay attributes	31
Table 50: RelatedDisplay standard attributes	31
Table 51: RelatedDisplay elements	31
Table 52: Display attributes	31
Table 53: Display elements	32
Table 54: MacroValuePair attributes	32
Table 55: X-Y chart attributes	32
Table 56: X-Y chart elements	32
Table 57: xySeries attributes	33
Table 58: Bar chart attributes	34
Table 59: Bar chart elements	34
Table 60: Series attributes	34
Table 61: Bar chart attributes	35
Table 62: Intensity plot standard attributes	35



1. INTRODUCTION

WebCA is a framework which allows users easy and fast creation of Channel Access (CA) Clients using web browsers.

Users create CA clients by composing WebCA components in XML ([3]) document called CAML document. During creation CAML document is validated against XML Schema ([4]). This concept expunges errors in CAML document.

XML document is then transformed by web browser using XSL Transformation (XSLT [5]) into HTML powered by JavaScript. Web browser must have NPCA plugin [1] installed.

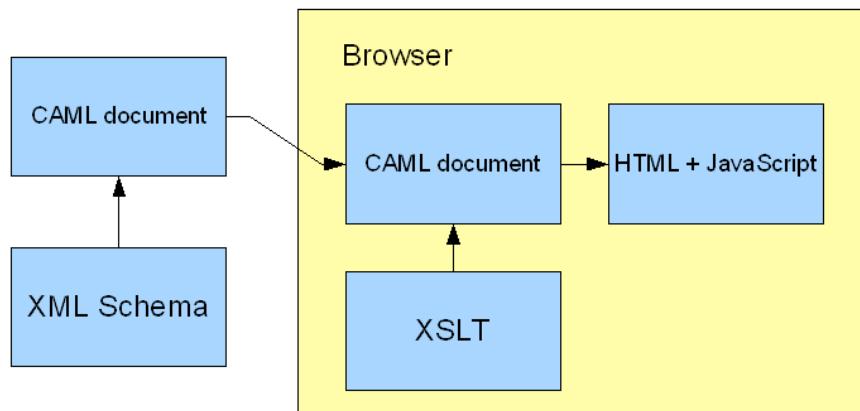


Figure 1: WebCA

WebCA was tested on the following web browsers:

OS	Browser	Version	Comments
MAC OS X Leopard	Safari	3.0.4	WebKit nightly build was used due to SVG render problems.
	Firefox	2.0.0.11	SVG render problems.
	Firefox	3 beta 2	
	Camino	1.5.3	Same SVG problems as Firefox 2.0.0.11 (due to Gecko engine)
Windows XP SP2	Firefox	2.0.0.10	
	Safari	3.0.4	WebKit r28899 was used due to SVG render problems.
Linux (SL 4)	Firefox	2.0.0.9	

Table 1: Supported browsers

2. SIMPLE EXAMPLE

Simple xml file is shown below.

```
<?xml version="1.0"?>
<?xmlstylesheet href="xsl/webca.xsl" type="text/xsl" ?>
<caml
  title="WebCa Simple Example"
  webcaPath="file:///D:/workspace/nPCA2/webca/">>

<staticText>Simple example of PV monitor.</staticText>

<textUpdate readbackPV="record1"/>

</caml>
```

Copy this xml into your favorite xml editor and change the webcaPath attribute of caml object. This attribute must point to WebCA installation. Open file in one of the supported browsers (see 1 for supported browsers).



3. CONTEXT MENU

If user presses right mouse button over component (e.g. Slider) context menu will popup with “PV inspector” as the only option to select. If this option is selected new window will be created with PV details displayed.

Example of context menu and PV inspector window is shown below.

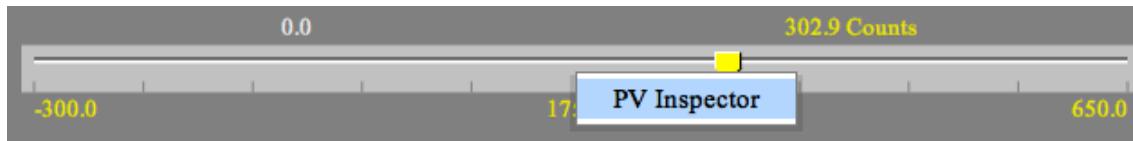


Figure 2: Context menu example

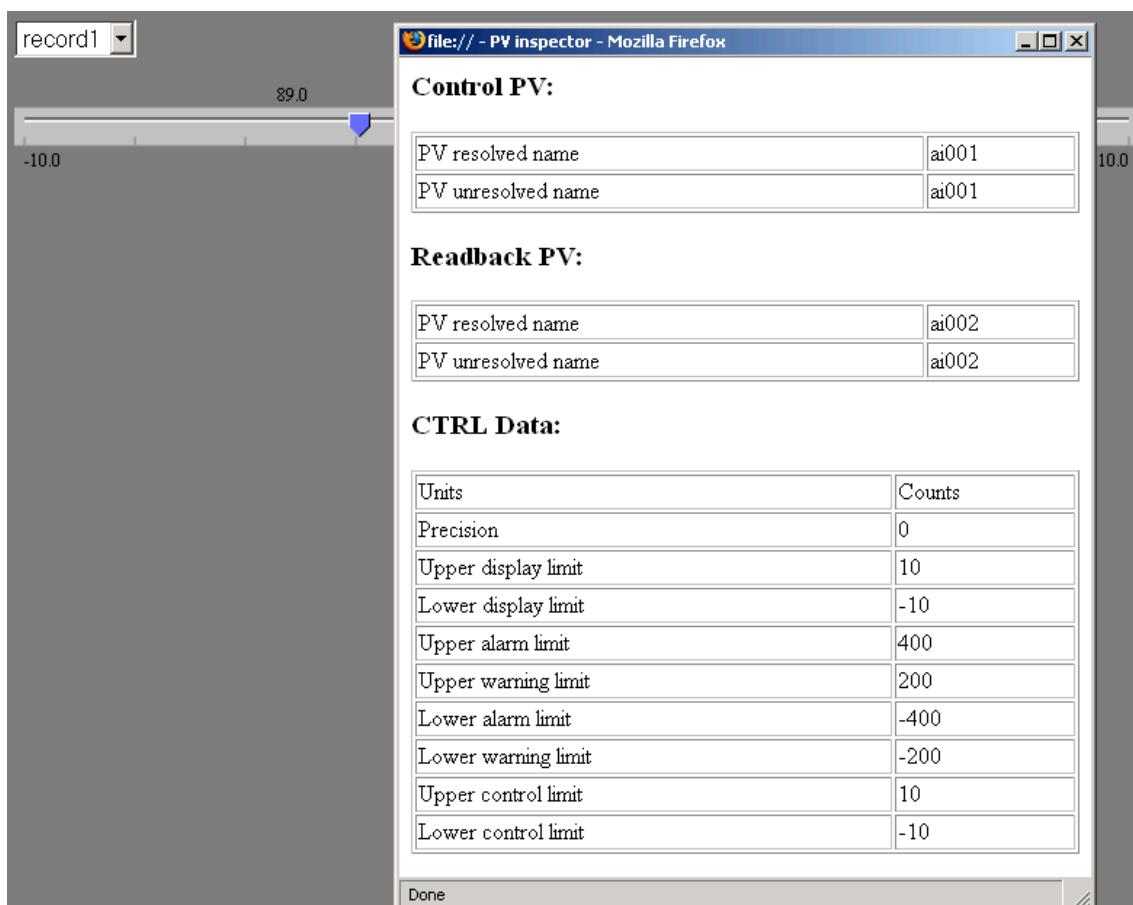


Figure 3: PV inspector example

4. NUMBER FORMAT

Components which are used to display or control decimal values has numberFormat attribute which can be used to define format in which decimal value should be displayed.

Formats are specified using the following characters >#, 0, ., E, +< where meaning is the following:

- # - digit that is displayed only if not zero
- 0 - digit that is displayed also when zero
- . - used for decimal point
- E - used for exponential format
- + - used for sign

If exponential format is used >#< must not appear in format specification.

Examples of formats definitions and wheelSwitch components using these formats are shown below:

- +##000.00
- +00.000
- +00.000###
- +00.00E+00

-	0	0	0	.	9	9	units	 
-	0	0	.	9	9	2	units	 
-	0	.	9	9	2	9	units	 
-	9	9	.	2	9	E	-	 

Figure 4: Number format examples



5. MACRO SUBSTITUTION

Macro substitution is supported by WebCA. There are two ways of supplying macros to xml document:

- Using query string. Example is shown below:

<file:///D:/workspace/npc2/webca/slider.xml?NAME=ai003&NUMBER1=1&NUMBER2=2>

Slider.xml will be opened with macros NAME, NUMBER1 and NUMBER2 defined with values ai003, 1 and 2 respectively.

- Using “substitutor” component. Example of this component is mux. See 6.7 for details on mux.



Macros defined in query string have precedence from macros defined in substitutor component.

6. COMPONENTS

6.1. CAML ATTRIBUTES

The following are attributes of the CAML object.

Attribute	value	required/optional	description
webcaPath	string	required	Defines the relative path of the WebCa installation from xml document.
style	string	optional	Defines additional class of all components which can be used in CSS to define additional styles.
title	string	optional	Defines the title of the xml document.
pendEvents	decimal	optional	Defines the number of the events processed by npca plugin when npca plugin's pendEvents function is called. See ref [1] for details on pendEvents function. If not defined default value is 42.
pendEventsPeriodMs	decimal	optional	Defines the time interval in ms in which plugin's pendEvents function is called. See ref [1] for details on pendEvents function. If not defined default value 100 ms is used.

Table 2: CAML object attributes

6.2. STANDARD COMPONENT ATTRIBUTES

attribute	value	required/optional	description
size	enumerated type (small, medium large)	optional	Defines the size of the text.
style	string	optional	Defines additional class of the component which can be used in CSS to define additional styles.
readbackPV	string	required on monitor components, optional or prohibited otherwise	Readback PV name.
controlPV	string	Required on control components, prohibited otherwise	Control PV name.
alarmSensitive	boolean	optional	Defines whether component is alarm sensitive. Default value is true.
displayFormat	string	optional	Defines the display format on applicable components. For details on displayFormat see 4.

Table 3: Standard component attributes

6.3. LAYOUT COMPONENTS

The following layout components are available:

- horizontalPanel
- verticalPanel
- column
- row
- panel

6.3.1. HorizontalPanel

HorizontalPanel is used to create panel with one row and unlimited columns.

Example of xml definition is shown below

```
<horizontalPanel caption="Simple horizontal Panel"
                 style="panelWithSomeDecorations">
    <column>
        ...
    </column>
    <column>
```

```

    ...
</column>
.
.
.
</horizontalPanel>
```

HorizontalPanel attributes:

attribute	value	required/optional	description
caption	string	optional	Defines the caption for the horizontalPanel.
width	pixels %	optional	Sets width of the horizontal panel.

Table 4: HorizontalPanel attributes

Standard attributes:

Style

Table 5: HorizontalPanel standard attributes

For a full description of standard attributes, go to 6.2

HorizontalPanel elements:

Element	min/max occurs	required/optional	description
column	0/unbounded	optional	Defines column in horizontalPanel.

Table 6: HorizontalPanel elements

6.3.2. VerticalPanel

VerticalPanel is used to create panel with one column and unlimited rows.

Example of xml definition is shown below.

```

<verticalPanel caption="Simple vertical Panel"
                style="panelWithSomeDecorations">
    <row>
        ...
    </row>
    <row>
        ...
    </row>
    .
    .
    .
```

```
</verticalPanel>
```

VerticalPanel attributes:

attribute	value	required/optional	description
caption	string	optional	Defines the caption for the verticalPanel
width	pixels %	optional	Sets width of the vertical panel.

Table 7: VerticalPanel attributes

Standard attributes:

Style

Table 8: VerticalPanel standard attributes

For a full description of standard attributes, go to 6.2

VerticalPanel elements:

Element	min/max occurs	required/optional	description
row	0/unbounded	optional	Defines row in verticalPanel

Table 9: VerticalPanel elements

6.3.3. Column

Column is building block for horizontalPanel and panel.

Example of xml definition is shown below.

```
<column caption="Column caption"
        height="300px" width="50%"
        style="SomeStyle"
        colspan="4">
    ...
</column>
```

Column attributes:

attribute	value	required/optional	description
caption	string	optional	Defines the caption for the column
height	pixels %	optional	Sets height of the column
width	pixels %	optional	Sets width of the column
colspan	decimal	option	Defines the number of columns this column should span if used inside panel

Table 10: Column attributes

Standard attributes:

style

Table 11: Column standard attributes

For a full description of standard attributes, go to 6.2.

6.3.4. Row

Row is building block for verticalPanel or panel.

Example of xml definition is shown below.

```
<row caption="Column caption"
      height="300px" width="50%"
      style="SomeStyle">
    ...
</row>
```

Row attributes:

attribute	value	required/optional	description
caption	string	optional	Defines the caption for the row
height	pixels %	optional	Sets height of the row
width	pixels %	optional	Sets width of the row

Table 12: Row attributes

Standard attributes:

style

Table 13: Row standard attributes

For a full description of standard attributes, go to 6.2.

6.3.5. Panel

Panel is used to build complex layouts.

Example of xml definition is shown below.

```
<panel style="SomeStyle">
    <row>
        <column>
            ...
        </column>
    </row>
    <row>
        ...
    </row>
</panel>
```

Panel attributes:

attribute	value	required/optional	description
caption	string	optional	Defines the caption for the panel.
width	pixels %	optional	Sets width of the panel.

Table 14: Panel attributes

Standard attributes:

style

Table 15: Panel standard attributes

For a full description of standard attributes, go to 6.2.

Panel elements:

Element	min/max occurs	required/optional	description
row	0/unbounded	optional	Defines row in panel

Table 16: Panel elements

Example of building layout using above components:

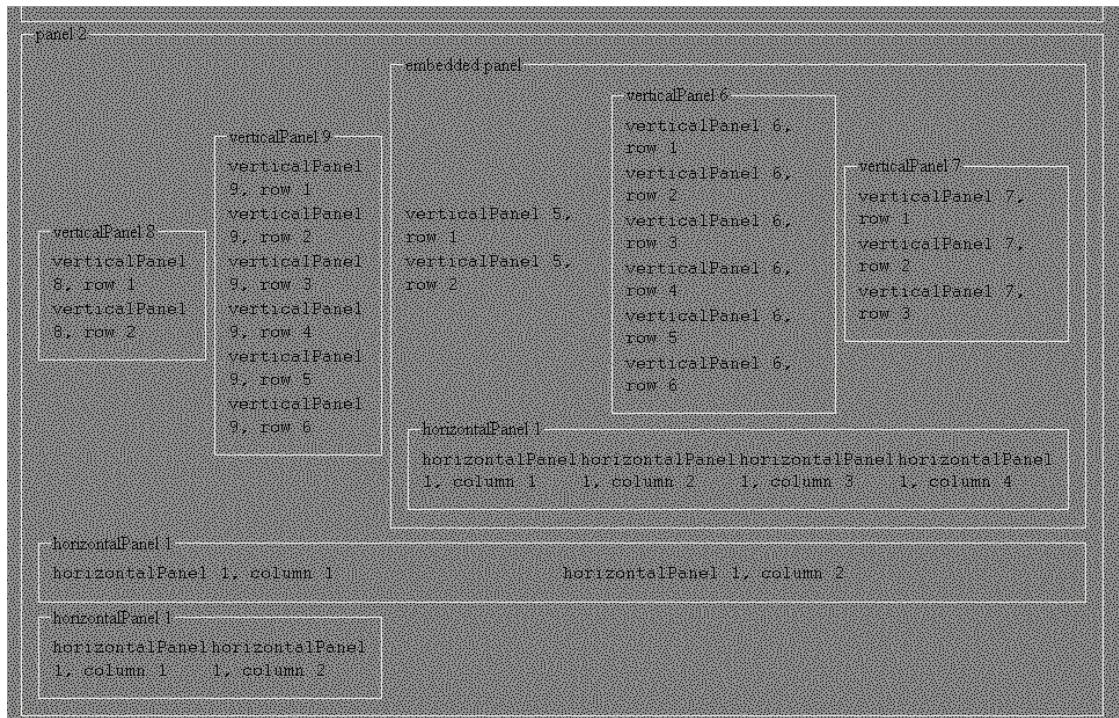


Figure 5: Complex layout example

6.4. TABVIEW

TabView is used to create tabbed view.

Example of xml definition is shown below.

```

<tabView style="SomeStyle" orientation="top">
    <tab name="Tab 1" selected="true">
        ...
    </tab>
    <tab name="Tab 2">
        ...
    </tab>
</tabView>
  
```

TabView attributes:

attribute	value	required/optional	description
orientation	enumerated value (top, left, right, bottom)	optional	Defines orientation of tabbed view. Default value is top.

Table 17: TabView attributes

Standard attributes:

style

Table 18: TabView standard attributes

For a full description of standard attributes, go to 6.2.

TabView elements:

Element	min/max occurs	required/optional	description
tap	1/unbounded	required	Defines a single tab in tabbed view.

Table 19: TabView elements

Tab attributes:

attribute	value	required/optional	description
name	string	required	Defines the name of the tab.
selected	boolean	optional	Defines which tab is selected when page is opened.

Table 20: Tab attributes

Example of tabView is shown below:

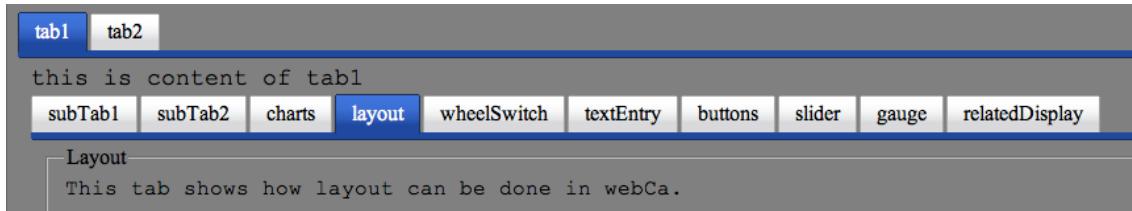


Figure 6: TabView example

6.5. IMAGE

Example of xml definition is shown below:

```

</img>
```

Image attributes:

attribute	value	required/optional	description
src	string	required	The URL of the image to display.
alt	string	optional	Defines a short description of the image.
width	pixels %	optional	Sets the width of an image.
height	pixels %	optional	Sets the height of an image.

Table 21: Image attributes

Standard attributes:

size, style

Table 22: Image standard attributes

For a full description of standard attributes, go to 6.2

6.6. STATICTEXT

StaticText is used to display static text.

Example of xml definition is shown below.

```
<staticText size="small" style="someStyle" href="example.xml">
    Some text to display. And macro to be displayed $(NAME) .
</staticText>
```

StaticText attributes:

attribute	value	required/optional	description
href	string	optional	If defined staticText will act as a link.
target	enumerated type (_blank, _parent, _self, _top)	optional	Defines how link should be opened. Default value is _blank.

Table 23: StaticText attributes

Standard attributes:

size,style

Table 24: StaticText standard attributes

For a full description of standard attributes, go to 6.2

StaticText supports macro substitutions. See 5 for more information on macro substitutions.

Example of static text is shown below.

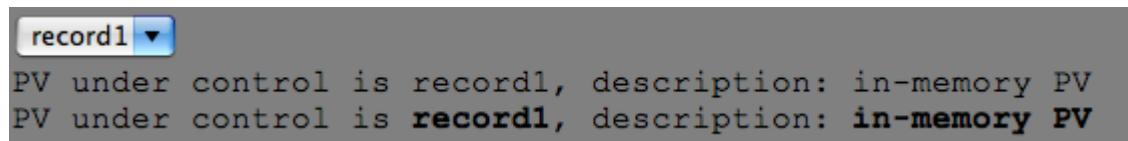


Figure 7: StaticText example

6.7. Mux

Mux is used for macro substitutions.

Example of xml definition is shown below:

```
<mux size="small" style="someStyle">
    <sequence name="record1" selected="true">
        <macroValuePair macroName="$ (NAME)"
                        macroValue="record1"/>
        <macroValuePair macroName="$ (DESC)"
                        macroValue="in-memory PV"/>
    </sequence>
    <sequence name="record2">
        <macroValuePair macroName="$ (NAME)"
                        macroValue="record2"/>
        <macroValuePair macroName="$ (DESC)"
                        macroValue="counter PV"/>
    </sequence>
    <sequence name="enum">
        <macroValuePair macroName="$ (NAME)"
                        macroValue="enum"/>
        <macroValuePair macroName="$ (DESC)"
                        macroValue="enumerated PV"/>
    </sequence>
</mux>
```

Standard attributes:

size,style

Table 25: Mux standard attributes

For a full description of standard attributes, go to 6.2

Mux elements:

Element	min/max occurs	required/optional	description
sequence	1/unbounded	required	Defines a sequence of macro value pairs

Table 26: Mux elements

Sequence attributes:

attribute	value	required/optional	description
name	string	required	Defines the name of the sequence.
selected	boolean	optional	Defines a default selected sequence

Table 27: Sequence attributes

Sequence elements:

Element	min/max occurs	required/optional	description
macroValuePair	1/unbounded	required	Defines a sequence of macro value pairs

Table 28: Sequence elements

MacroValuePair attributes:

Attribute	value	required/optional	description
macroName	string	required	The name of the macro.
macroValue	string	required	The value of the macro.

Table 29: MacroValuePair attributes

Example of mux component is shown below.



Figure 8: Mux component; left in Safari, right in Firefox

For details on macro substitutions see 5.

6.8. TEXTUPDATE

Example of xml definition is shown below:

```
<textUpdate
    readbackPV="PVname"
    alarmSensitive="true"
    dataType="numeric"
    displayFormat="#0.00"
    size="small"
    style="someStyle"/>
```

TextUpdate attributes:

Attribute	value	required/optional	description
dataType	Enumerated type (numeric, text)	optional	Defines the type of data. Default value is numeric.

Table 30: TextUpdate attributes

Standard attributes:

size, style, readbackPV, alarmSensitive, displayFormat
--

Table 31: TextUpdate standard attributes

For a full description of standard attributes, go to 6.2

Example of textUpdate is shown below.

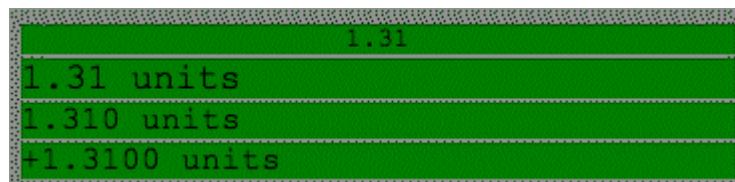


Figure 9: TextUpdate example

6.9. TEXTENTRY

Example of xml definition is shown below:

```
<textEntry
    controlPV="PVName"
    alarmSensitive="true"
    dataType="numeric"
    displayFormat="#0.00"
    size="small"
    style="smoeStyle"/>
```

TextEntry attributes:

Attribute	value	required/optional	description
dataType	Enumerated type (numeric, text)	optional	Defines the type of data. Default value is numeric.

Table 32: TextEntry attributes

Standard attributes:

size, style, controlPV, alarmSensitive, displayFormat

Table 33: TextEntry standard attributes

For a full description of standard attributes, go to 6.2

Example of textEntry is shown below.

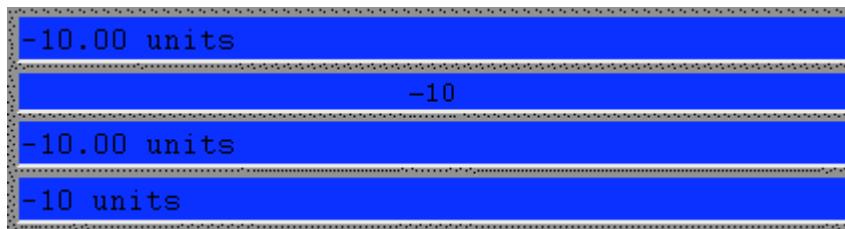


Figure 10: TextEntry example

6.10. MESSAGE BUTTON

Example of xml definition is shown below:

```
<messageButton controlPV="PVName"
    alarmSensitive="false"
    size="small"
    style="someStyle">
    <on caption="On caption" value="on value"></on>
    <off caption="off caption" value="off value"></off>
</messageButton>
```

Standard attributes:

size, style, controlPV, alarmSensitive
--

Table 34: MessageButton standard attributes

For a full description of standard attributes, go to 6.2

MessageButton elements:

Element	min/max occurs	required/optional	description
on	1/1	required	Defines the 'on' state of the messageButton
off	1/1	required	Defines the 'off' state of the messageButton

Table 35: MessageButton elements

On and off attributes:

Attribute	value	required/optional	description
caption	string	required	State caption
value	string	required	State value

Table 36: On and off attributes

Example of messageButton is shown below.

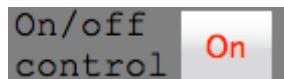


Figure 11: MessageButton example

6.11. MENUBUTTON

Example of xml definition is shown below:

```
<menuButton controlPV="controlPVName" readbackPV="readbackPVName"
           alarmSensitive="true">
    <menuButtonItem value="0" name="min"/>
    <menuButtonItem value="50" name="medium"/>
    <menuButtonItem value="100" name="full"/>
</menuButton>
```

Standard attributes:



size, style, controlPV, alarmSensitive, readbackPV, controlPV

Table 37: MenuButton standard attributes

For a full description of standard attributes, go to 6.2

MenuButton elements:

Element	min/max occurs	required/optional	description
menuButtonItem	0/unbounded	optional	Defines an item of the menu. If not defined button will try to get menu from PV.

Table 38: MenuButton elements

MenuButtonItem attributes:

Element	min/max occurs	required/optional	description
value	string	required	Menu item value
name	string	required	Menu item name

Table 39: MenuButtonItem attributes

Example of menuButton is shown below.

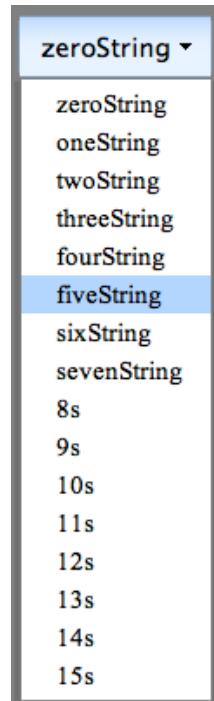


Figure 12: MenuButton example

6.12. RADIOBUTTON

Example of xml definition is shown below:

```
<radioButton controlPV="controlPVName"
             readbackPV="readbackPVName"
             alarmSensitive="true">
    <radioButtonItem value="0" name="min"/>
    <radioButtonItem value="50" name="medium"/>
    <radioButtonItem value="100" name="full"/>
</radioButton>
```

RadioButton attributes:

Element	value	required/optional	description
type	enumerated type (radio, toggle)	optional	When radio is selected radio button is traditional HTML radio button. If toggle is selected radio button is YUI [6] radio button. Default value is radio.

Table 40: RadioButton attributes

Standard attributes:

size, style, controlPV, alarmSensitive, readbackPV, controlPV

Table 41: RadioButton standard attributes

For a full description of standard attributes, go to 6.2

RadioButton elements:

Element	min/max occurs	required/optional	description
radioButtonItem	0/unbounded	optional	Defines an item of the menu. If not defined button will try to get menu from PV.

Table 42: RadioButton elements

RadioButtonItem attributes:

Element	value	required/optional	description
value	string	required	Menu item value
name	string	required	Menu item name

Table 43: RadioButtonItem attributes

Example of radioButton is shown below.



Figure 13: RadioButton example

6.13. WHEELSWITCH

Example of xml definition is shown below:

```
<wheelSwitch
    controlPV="PVName"
    alarmSensitive="true"
    displayFormat="#0.00"
    size="small"
    style="someStyle"/>
```

Standard attributes:

size, style, controlPV, alarmSensitive, displayFormat

Table 44: WheelSwitch standard attributes

For a full description of standard attributes, go to 6.2

Example of wheelSwitch is shown below:



Figure 14: WheelSwitch example

6.14. SLIDER

Example of xml definition is shown below:

```
<slider
    controlPV="controlPVName"
    readbackPV="readbackPVName"
    maxValue="650"
    minValue="-300"
    increment="0.01"
    displayFormat="#0.000"/>
```

Slider attributes:

attribute	value	required/optional	description
maxValue	string	optional	Max value to be displayed.
minValue	string	optional	Min value to be displayed.
increment	decimal	optional	Increment to be used when clicking slider area. Default value is 0.1.

Table 45: Slider attributes

Standard attributes:

size, style, displayFormat, alarmSensitive, controlPV, readbackPV

Table 46: Slider standard attributes

For a full description of standard attributes, go to 6.2

Example of slider is shown below:

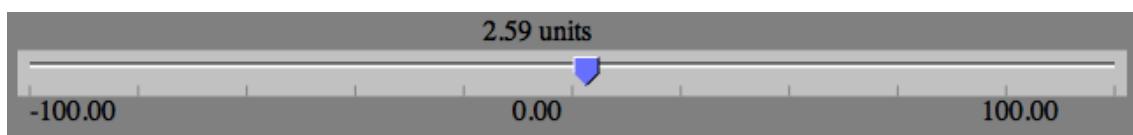


Figure 15: Slider example

6.15. GAUGE

Example of xml definition is shown below:

```
<gauge
    readbackPV="readbackPVName"
    maxValue="650"
```

```
minValue="-300"
displayFormat="#0.000"/>
```

Gauge attributes:

attribute	value	required/optional	description
maxValue	string	optional	Max value to be displayed.
minValue	string	optional	Min value to be displayed.

Table 47: Gauge attributes

Standard attributes:

```
size, style, displayFormat, alarmSensitive, readbackPV
```

Table 48: Gauge standard attributes

For a full description of standard attributes, go to 6.2

Example of gauge is shown below:

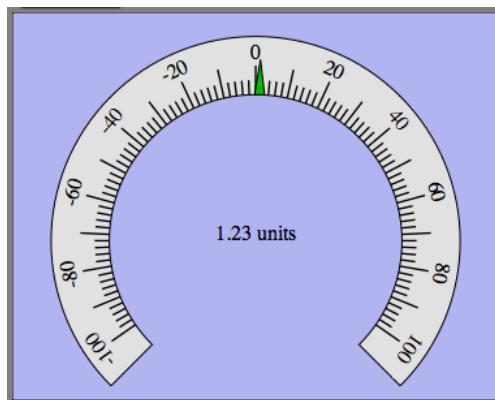


Figure 16: Gauge example

6.16. RELATEDDISPLAY

Example of xml definition is shown below:

```
<relatedDisplay type="image" src="logo_cosylab.gif"
                alt="alt" width="500px" height="200px" >
    <display src="slider.xml" name="beamline1">
        <macroValuePair macroName="${NAME}" macroValue="val1"/>
        <macroValuePair macroName="${NAME1}" macroValue="val2"/>
        <macroValuePair macroName="${DESC}" macroValue="desc"/>
    </display>
</relatedDisplay>
```

RelatedDisplay attributes:

Attribute	value	required/optional	description
type	enumerated type	required	Defines type of related display (image or button)
src	string	Required if image type.	The URL of the image to display. Only applicable for image type.
alt	string	optional	Defines a short description of the image. Only applicable for image type.
width	pixels %	optional	Sets the width of an image. Only applicable for image type.
height	pixels %	optional	Sets the height of an image. Only applicable for image type.

Table 49: RelatedDisplay attributes

Standard attributes:

size, style

Table 50: RelatedDisplay standard attributes

For a full description of standard attributes, go to 6.2

RelatedDisplay elements:

Element	min/max occurs	required/optional	description
display	1/1	required	Defines the display to launch

Table 51: RelatedDisplay elements

Display attributes:

Attribute	value	required/optional	description
src	string	required	The URL of the display to open.
name	string	required	Name of the display. Caption on the button if button is the type.
macroPropagation	boolean	optional	Defines if active macros should be propagated to opened display. Default value is false.
target	enumerated type (_blank, _parent, _self, _top)	optional	Defines how new display should be opened. Default value is _blank.

Table 52: Display attributes

Display elements:

Element	min/max occurs	required/optional	description
macroValuePair	0/unbounded	optional	Defines the macro/value pair

Table 53: Display elements

MacroValuePair attributes:

Attribute	value	required/optional	description
macroName	string	required	The name of the macro.
macroValue	string	required	The value of the macro.

Table 54: MacroValuePair attributes

6.17. X-Y CHART

To use X-Y Chart to plot array y against array x use the following xml definition:

```
<xyChart  xAxisLabel="Axis X label"
          yAxisLabel="Axis Y label">
  <xySeries Y-PVname="waveformY" name="energy"
             X-PVname="waveformX"/>
</xyChart>
```

X-Y chart component attributes:

Attribute	value	required/optional	description
xAxisLabel	string	optional	X axis label
yAxisLabel	string	optional	Y axis label

Table 55: X-Y chart attributes

X-Y chart elements:

Element	min/max occurs	required/optional	description
xySeries	1/unbounded	required	Defines an array of values

Table 56: X-Y chart elements

xySeries component attributes:

Attribute	value	required/optional	description
Y-PVname	string	required	PV name for y values.
X-PVname	string	optional	PV name for x values. If not defined y values are plotted against their index in the array.
Name	string	optional	Name of series.
Type	enumerated type (scatter, line)	optional	Default value is line.

Table 57: xySeries attributes

Example of X-Y chart is shown below.

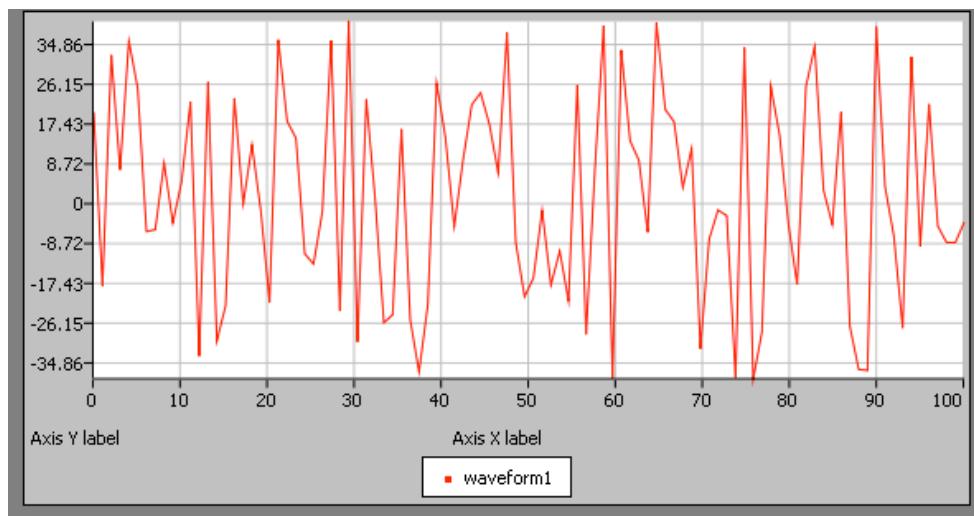


Figure 17: X-Y chart example

6.18. BAR CHART

Bar chart component is used to display array of values in bar chart.

Example of xml definition is shown below.

```
<barChart xAxisLabel="Axis X label"
          yAxisLabel="Axis Y label"
          xLabelsPV="xLabelsWaveform"
          xLabels="['mon','tue','wen']">
    <series PVname="barWaveform1" name="series1"/>
    <series PVname="barWaveform2"/>
</barChart>
```

Bar chart attributes:

Attribute	value	required/optional	description
xAxisLabel	string	optional	X axis label
yAxisLabel	string	optional	Y axis label
xLabelsPV	string	optional	PV name for x axis series labels, if not defined xLabels will be used
xLabels	string	optional	Array of x series labels

Table 58: Bar chart attributes

Bar chart elements:

Element	min/max occurs	required/optional	description
series	1/unbounded	required	Defines one array of values in bar chart.

Table 59: Bar chart elements

Series attributes:

Attribute	value	required/optional	description
PVname	string	required	PV name for array of values.
name	string	optional	Name of series.

Table 60: Series attributes

Example of bar chart component connected to two series of data is shown below

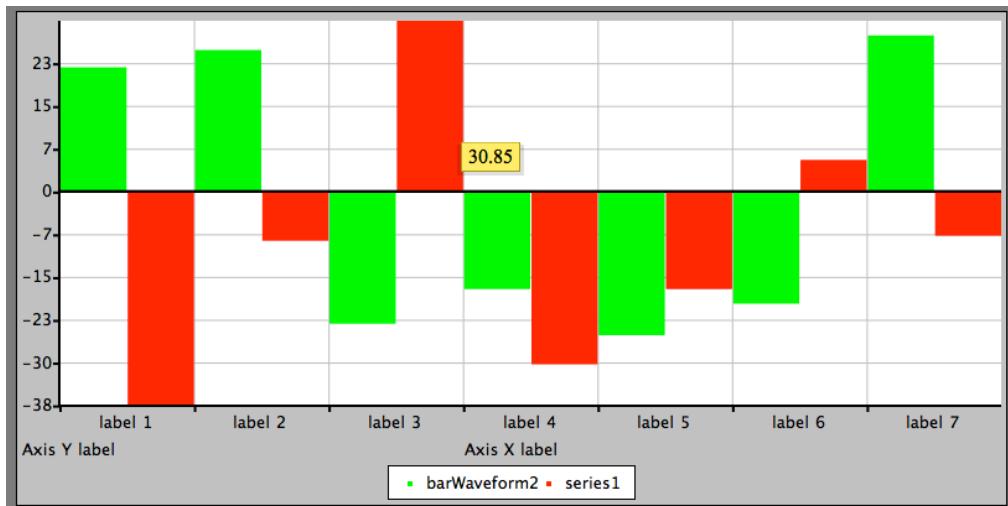


Figure 18: Bar Chart

6.19. INTENSITY PLOT

Example of xml definition is shown below.

```
<intensityPlot readbackPV="PVname" width="320" height="240"
maxValue="256" waterfall="false"/>
```

Intensity plot attributes:

Attribute	value	required/optional	description
width	pixel %	required	width of intensity plot
height	pixel %	required	height of intensity plot
maxValue	decimal	required	Intensity plot consists of grayscale image, when minimum value (0) is represented as black pixel and maximum value element as white. This attribute defines maximum value.
waterfall	boolean	required	if true, intensity plot is shown as waterfall (rows are sequentially added one after another over the time)

Table 61: Bar chart attributes

Standard attributes:

size, style, readbackPV, alarmSensitive

Table 62: Intensity plot standard attributes

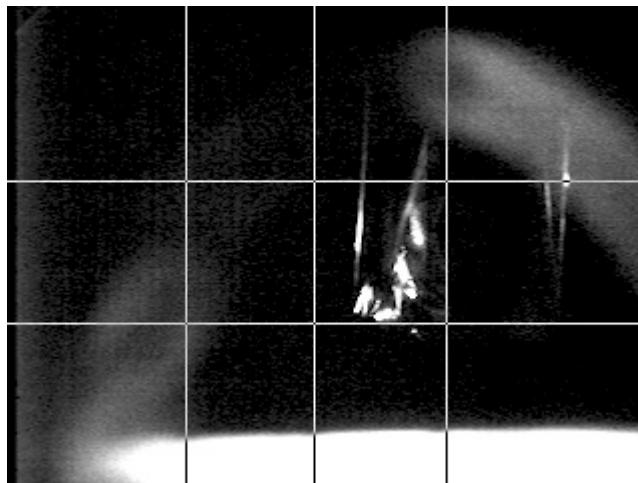


Figure 19: Intensity plot

7. COMPLEX EXAMPLE

Example device control panel created with WebCA is located in WebCA distribution under example directory. There is also EPICS application containing records to which this panel will connect.

Screenshot of this panel is shown below.

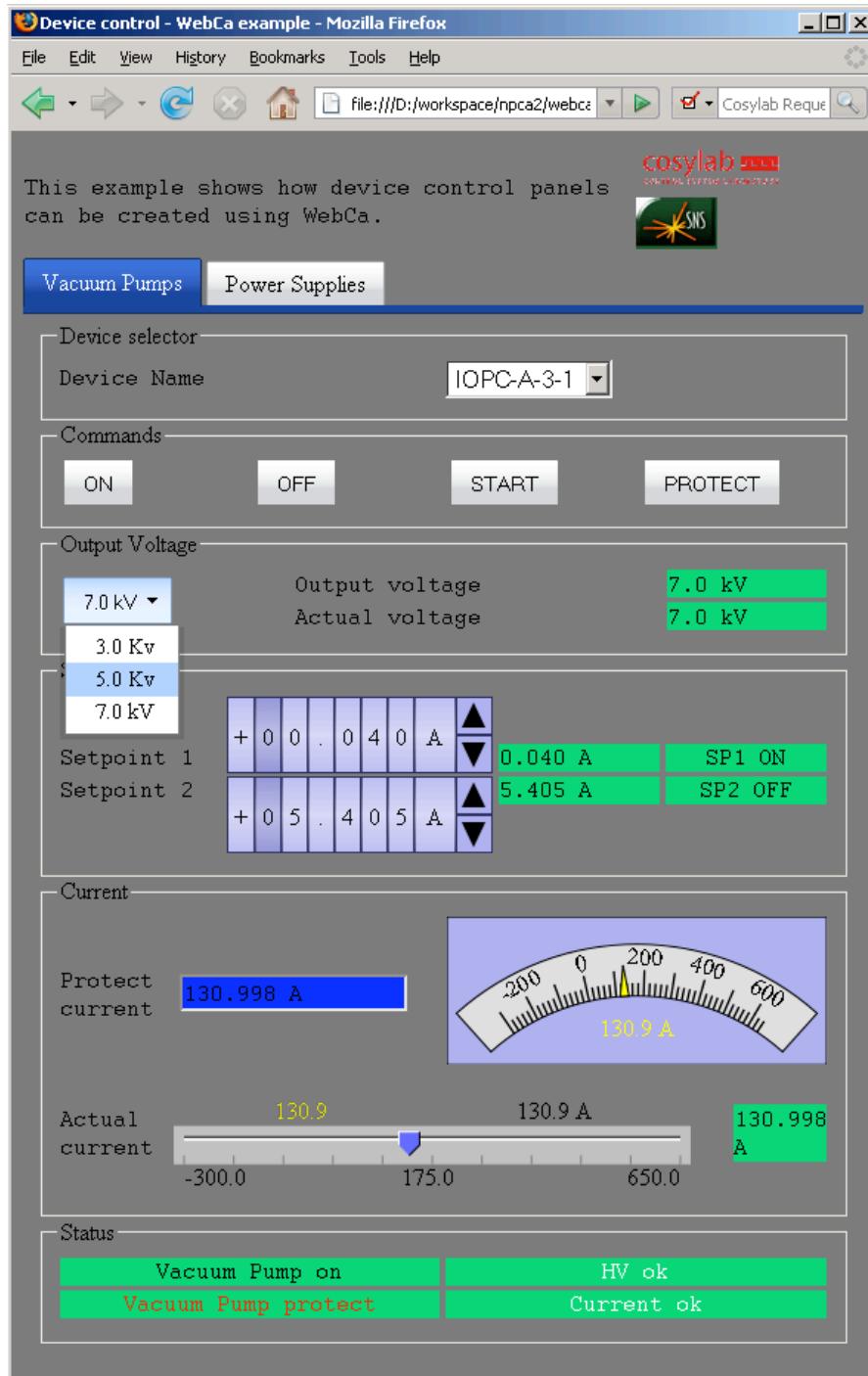


Figure 20: Example device control panel

8. ADDING NEW COMPONENT TO WEBCA

To add new component number of steps must be performed. We will use “barMeter” as an example component that will be added to WebCA.

- Add barMeter component to xml schema
 - Create new file named barMeter.xsd under xmlSchema directory. This file contains barMeter xmlSchema definition.
 - Add the following two lines in file webca.xsd:
 - <xsl:include schemaLocation=" barMeter.xsd"/>
 - <xsl:element ref=" barMeter"/> - this line should be included in group named “components”.
- Create JavaScript files under js/barMeter directory which defines the behavior of the component.
- Modify webca.xsl file under xsl directory
 - Add inclusion of JavaScript files like this:

```
<script type="text/javascript">
  <xsl:attribute name="src">
    <xsl:value-of
      select="@webcaPath"/>js/chart/barMeter.js</xsl:attribute>
</script>
```

- Create xsl template which defines barMeter and include it at the end of webca.xsl file
- In JavaScript “init” function which is at the beginning of the body section in webca.xsl file add these lines:

```
<xsl:for-each select="descendant::tabView">
  <xsl:value-of select="local-name()"/><xsl:value-of
    select="generate-id()"/>Init()
</xsl:for-each>
<xsl:for-each select="descendant::xi:include">
  <xsl:for-each
    select="document(@href)/caml/descendant::tabView">
    <xsl:value-of select="local-name()"/><xsl:value-of
      select="generate-id()"/>Init()
  </xsl:for-each>
</xsl:for-each>
```

- Modify file common.xsl under xsl direcotory
 - Add barMeter component to “includedComponentsArray” template. Use the code below:

```
<xsl:for-each select="document(@href)/caml/descendant::barMeter">
    ids.push("<xsl:value-of
        select='local-name()' /><xsl:value-of select='generate-id()' />");

    components.push(<xsl:value-of select='local-name()' /><xsl:value-of
        select='generate-id()' />);

</xsl:for-each>
```

- Add barMeter component to “componentsArray” template. Use the code below:

```
<xsl:for-each select="descendant::barMeter">
    ids.push("<xsl:value-of
        select='local-name()' /><xsl:value-of select='generate-id()' />");

    components.push(<xsl:value-of
        select='local-name()' /><xsl:value-of select='generate-id()' />);

</xsl:for-each>
```