ACOP + COMA + TINE in Java

Igor Križnar, Cosylab
26.09.2007

Contents

- Presentation of concept
- Summary of features
- Discussion

Demo 1: Putting application together

- Create new visual class from Frame
- Check generated code
- Add chart, gauger, slider, wheelswitch
- Show customizers
- Show generated code
- Select connection
- Start application
- Drop rest of connections to designed application

Demo 2: Run-time customization

- Add channel to chart
- Add channel to chart with converter
- Add converter to slider
- D&D this setting to native text area
- D&D this settings to another application instance

Demo 3: Inducing Coma

- Start coma starter
- Put together new simple panel,connect to channel
- Save/load with starter
- Add coma to demo
- Save/load xml to demo app

Summary of Features

- ACOP Java Beans features
- COMA features

Goals of ACOP Java Components

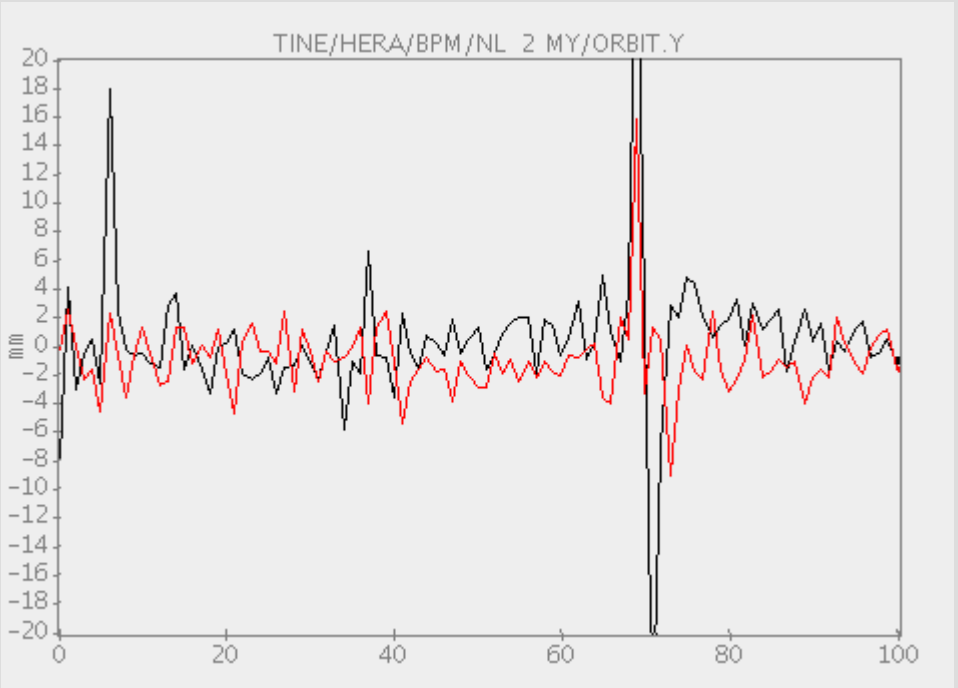
- Follow JavaBeans standards and use Swing components
  - Usable in Eclipse VE and other VCE
  - Suitable for newbies
- Simple panels assembled and customized with mouse clicks
  - Custom customizers
  - With minimal manual coding
- Rapid application development suitable components

Available ACOP Expert Java Components

- Acop
  - Chart with powerful API, familiar with VB interface
  - Pure GUI logic
- AcopTransport
  - Wrapper for TINE and simulation
  - Pure data logic
  - Can be used without Acop chart

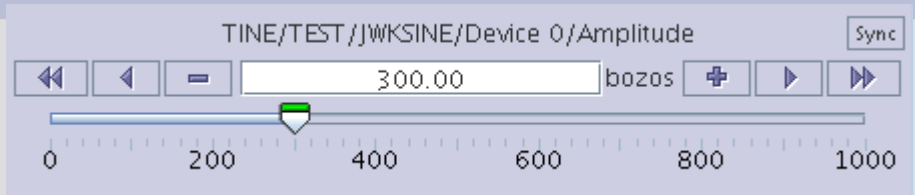
New ACOP Java Components 1

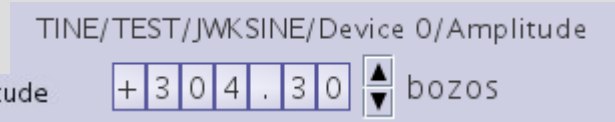
- AcopChart
  - Extends Acop chart in order to fit into RAD toolkit
  - Tries to guess most suitable operation mode
  - Fully configurable in desing-time and run-time mode
  - Still usable as Acop chart

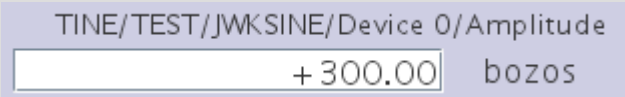


New ACOP Components 2

- AcopSlider
- AcopWheelswitch
- AcopDialknob
  - Changes value with mouse pointer
- AcopLabel
  - Type-in value
- AcopGauger
  - RO value display





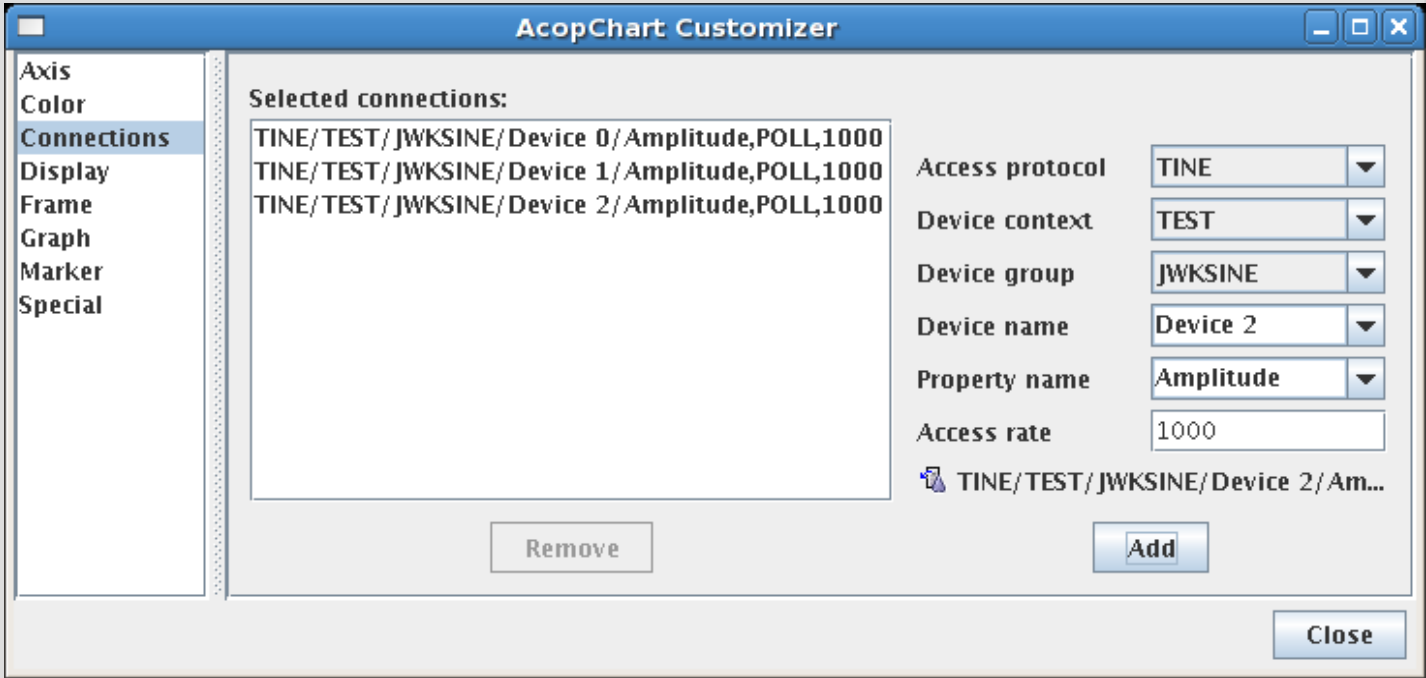


Common ACOP Java Displayer Features

- Connects to TINE trough AcopTransport
- D&D data exchange
  - Simple string (TINE channel name)
  - Connection parameters
  - Component parameters
- At design- and runt-time
  - Connection browser
  - Customizer dialogs
  - Value manipulation plug-ins intercepts value between transport and GUI

Other Useful ACOP Java Component(s)

- ConnectionCustomizer
  - Reusable TINE connection browser component for single and multiple selection
- Extensible set of value manipulation plug-ins
- General purpose visual and non-visual beans

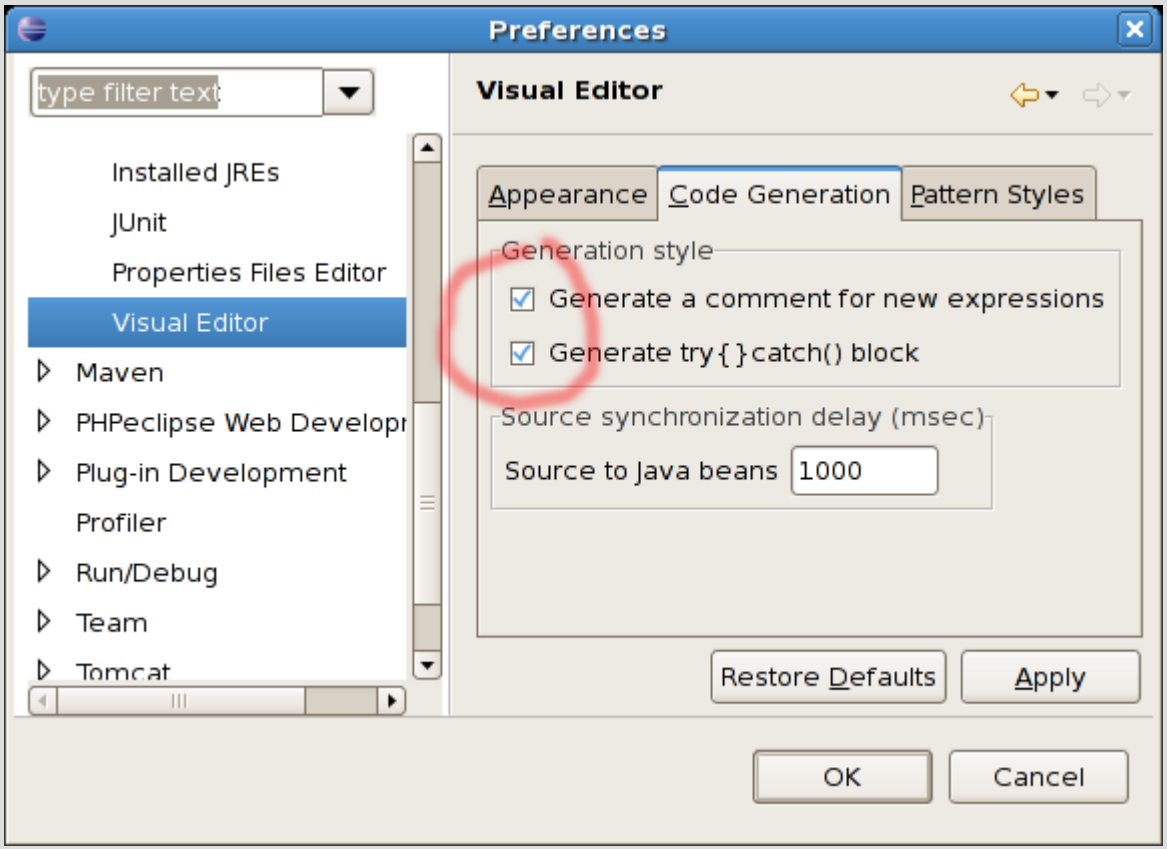


Where To Get Them?

- In `src/main/java`
  - `de.desy.acop.displayers`
    - ACOP displayers
  - `de.desy.acop.displayers.selector`
    - Connection browsers
  - `de.desy.acop.transport`
    - AcopTransport
- In `src/displayers/java`
  - Usefull beans from cosylab
- In `src/test/java`
  - Demos and tests

Before Practice Examples

- Configure your Eclipse VE **!!!**
  - Minimizes exception damage



Demo Applications

- de.desy.acop.demo.AcopChartDemo
  - Simple composition with AcopChart.
  - Connections set in design-time with multiple connection customizer
- de.desy.acop.demo.AcopConvertersDemo
  - On-the-fly value rendering with converters
- de.desy.acop.demo.AcopDisplayersDemo
  - Several Acop displayers
  - Connection set in design-time with connection customizer

COMA

- Runtime Java clients editor
- For “thin clients”
  - Removes need for programming skills
  - Fast and easy application changes
  - Works best with smart non-coupled Java Beans components (like ACOP)
- For “rich clients” as well
  - A lot business logic, coupled GUI components
  - Application can be used in coma in hybrid mode
  - Removes need for programming skills for simple tasks

COMA Features

- Very lightweight:
  - Single line: `new Coma(this);`
  - Or no coding, run trough Coma starter
- Stores client configuration in XML files
  - Stores partial or full client configuration
  - Can be manually edited
- Configuration can be stores, loaded or reloaded (rich client with several “flavors”)
- Operates with Java Beans and reuse customizers (work in progress)

XML configuration file example:

```
<?xml version="1.0" encoding="UTF-8"?>
<config>
    <width> 492</width>
    <height> 427</height>
    <components>
…
      <object class = "de.desy.acop.displayers.AcopDialKnob" name = "object_2" persistent = "false">
            <x> 335</x>
            <y> 264</y>
            <width>99</width>
            <height>115</height>
            <connection>TINE/TEST/SINE/SINEDEV_0/Amplitude,POLL,1000,-1</connection>
            <graphMin type = "class java.lang.Double">0.0</graphMin>
            <enabled type = "class java.lang.Boolean">true</enabled>
            <title type = "class java.lang.String">TINE/TEST/SINE/SINEDEV_0/Amplitude</title>
            <userValue type = "class java.lang.Double">5.139999866485596</userValue>
            <graphMax type = "class java.lang.Double">1000.0</graphMax>
            <minimum type = "class java.lang.Double">0.0</minimum>
            <maximum type = "class java.lang.Double">1000.0</maximum>
            <userMin type = "class java.lang.Double">0.0</userMin>
            <value type = "class java.lang.Double">267.4703674316406</value>
            <dataState type = "class com.cosylab.gui.displayers.DataState">Normal = { 2007-04-03 19:03:52.670 "Success" }
      </dataState>
            <units type = "class java.lang.String">bozos</units>
            <userMax type = "class java.lang.Double">1000.0</userMax>
       </object>
…
</components>
</config>
```