spray

A quick way of creating Graphiti

Marko Boger – HTWG Konstanz
Karsten Thoms - Itemis
Jos Warmer - Independant
Fabio Filipelli, Markus Gerhart,
Michael Bauer, Steffen Kollosche – HTWG Konstanz

- Spray is an open source project
  - Started at the CodeGen 2011
  - By Jos Warmer, Karten Thoms and Marko Boger
  - Hosted at http://code.google.com/a/eclipselabs.org/p/spray/
- Our goal is to make the developement of graphical Editors as simple as a textual editor with tools like Xtext
- First Target Platform is Eclipse with Graphiti and EMF
- Version 0.4.0 released in March 2012

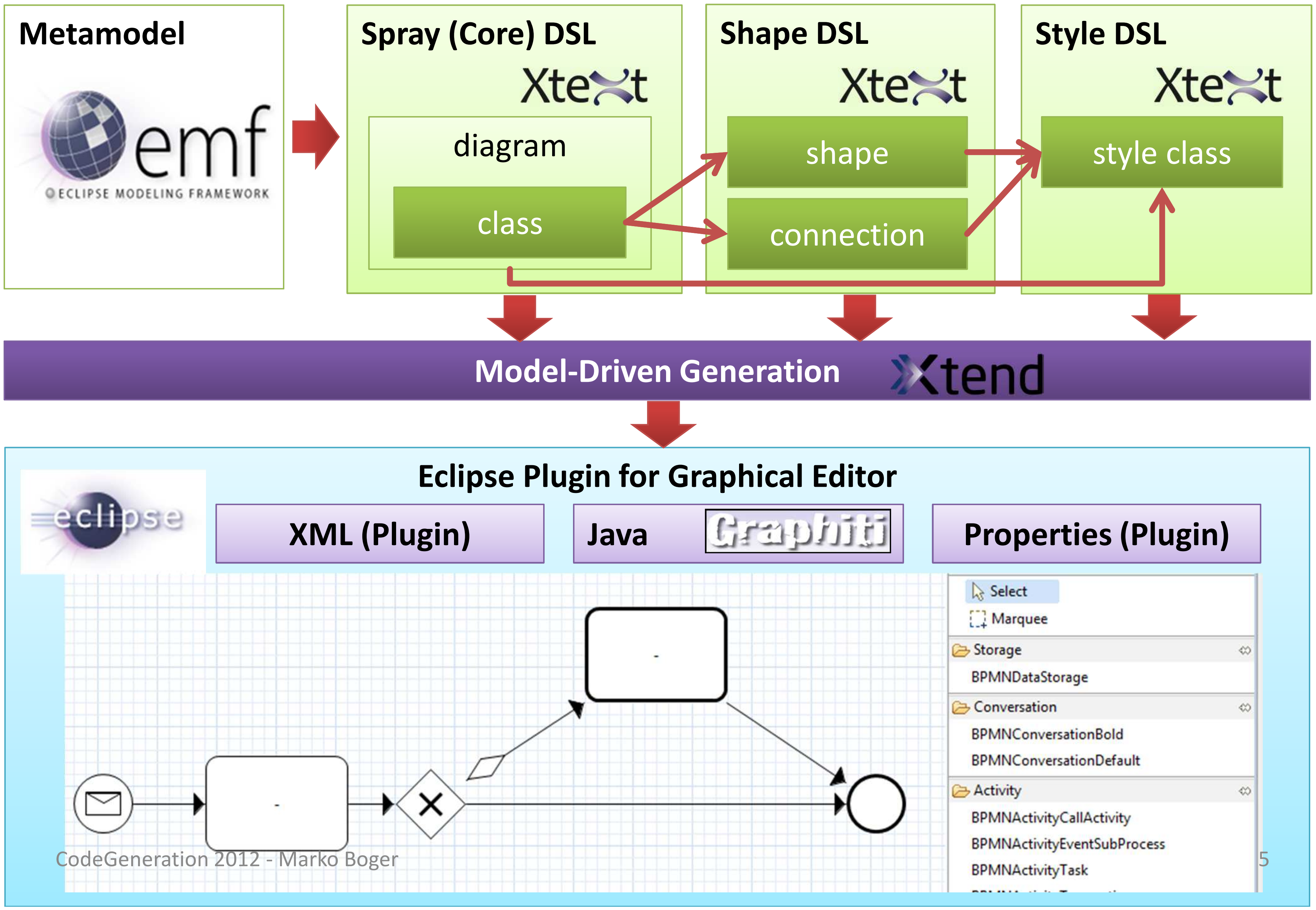
Graphiti

- Framework approach
- Easy to understand (relatively...)
- Hides complexity of GEF, Draw2D
- Only few core concepts
- Everything is a Feature
  - Add, Update, Move, Delete, ...
- Providers
  - Diagram Type Provider, Tool Provider, Image Provider, ...

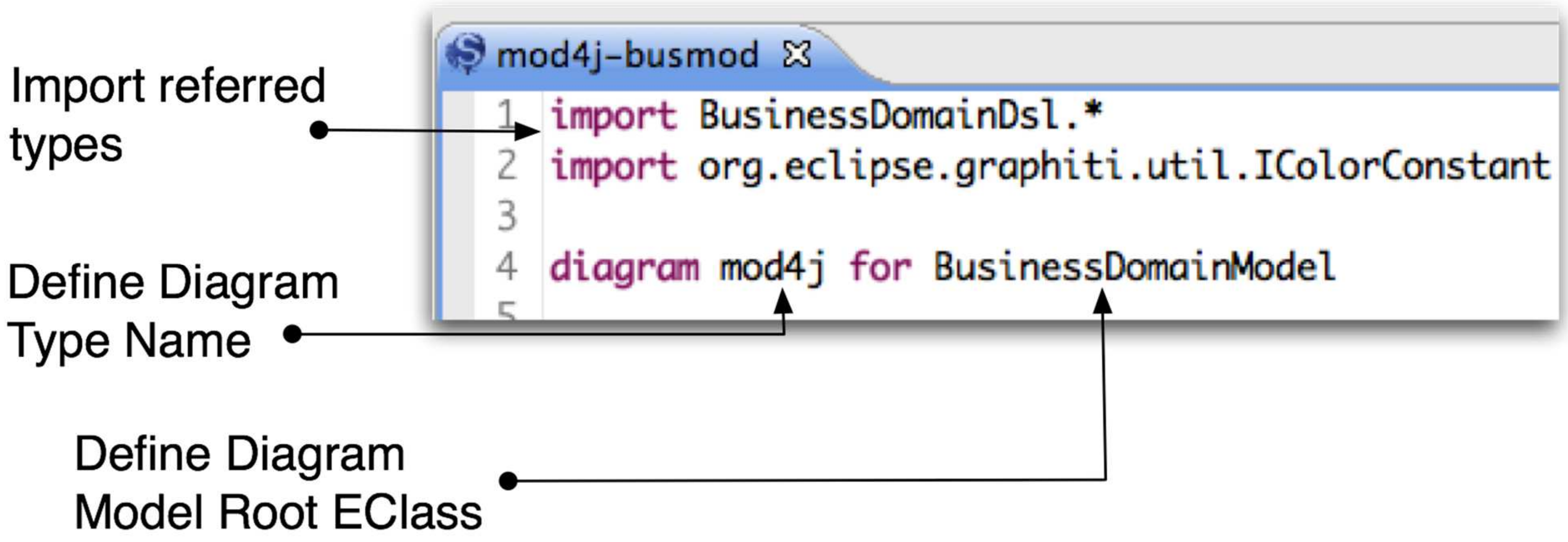
Graphiti

- Much code, often repetitive implementation
- Per mapped meta class
  - at least AddFeature, CreateFeature, UpdateFeature
  - Remove, Delete, Move, Layout, DrillDown, ...
  - Registration in FeatureProvider
  - Configure palette in ToolBehaviourProvider
- Repeat this e.g. for 10 meta classes...

spray - A quick way of creating Graphiti

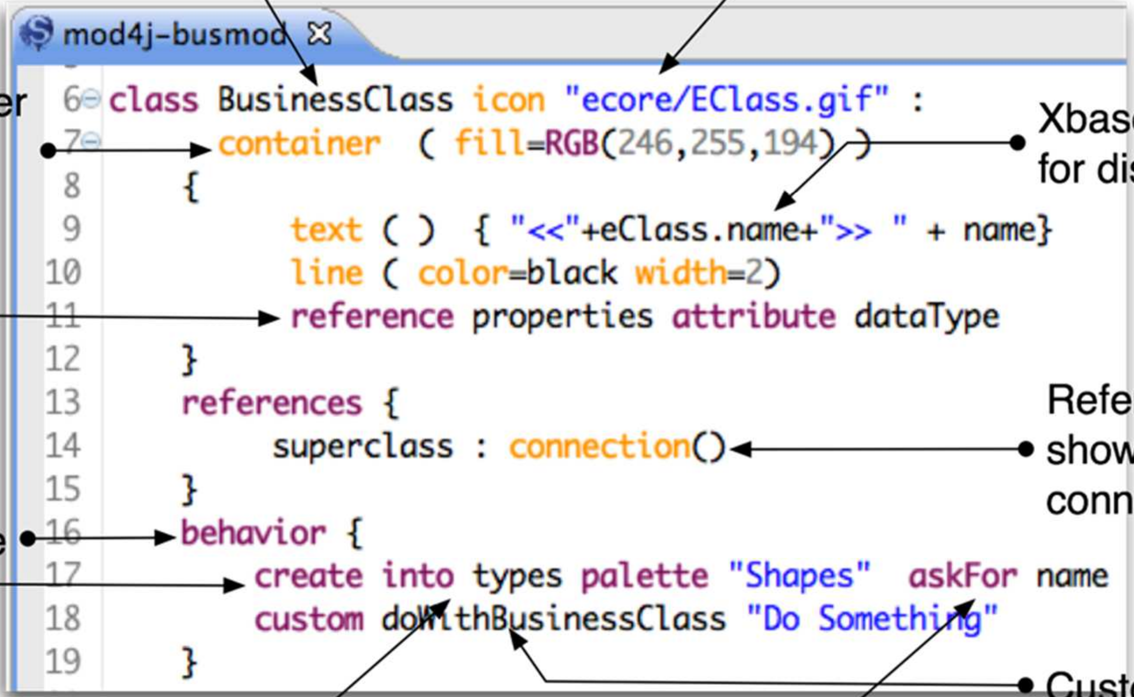


Spray DSL



```
import BusinessDomainDsl.*
import org.eclipse.graphiti.util.IColorConstant

diagram mod4j for BusinessDomainModel
```

Spray DSL

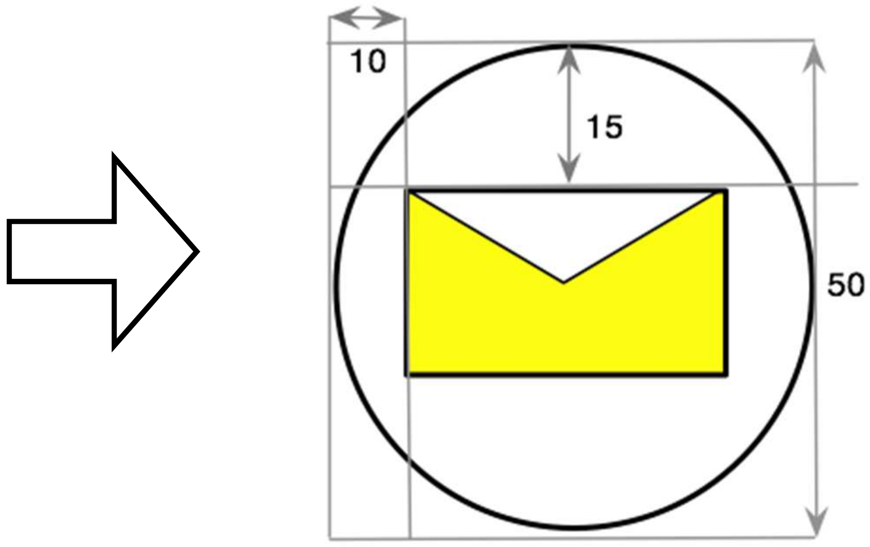


```
class BusinessClass icon "ecore/EClass.gif" :
    container  ( fill=RGB(246,255,194) )
    {
            text ( )  { "<<"+eClass.name+">> " + name}
            line ( color=black width=2)
            reference properties attribute dataType
    }
    references {
         superclass : connection()
    }
    behavior {
        create into types palette "Shapes"  askFor name
        custom doWithBusinessClass "Do Something"
    }
```

mod4j-busmod
Define visualization for EClass BusinessClass
Palette Icon
Rectangle container shape with light yellow fill color
Xbase Expression for displayed text
Reference feature for elements in compartment
Reference feature shown as connections
Behavior for shape
Will be creatable from palette
Custom Behavior
New instances are inserted into containment reference 'types' of model root
On creation open dialog to enter 'name' attribute value

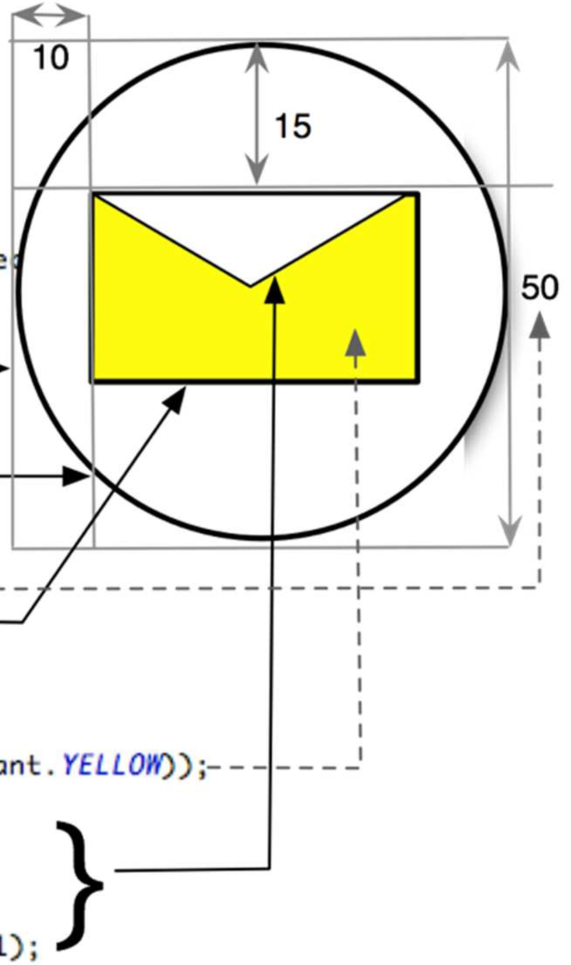
Defining Shapes

The Shape



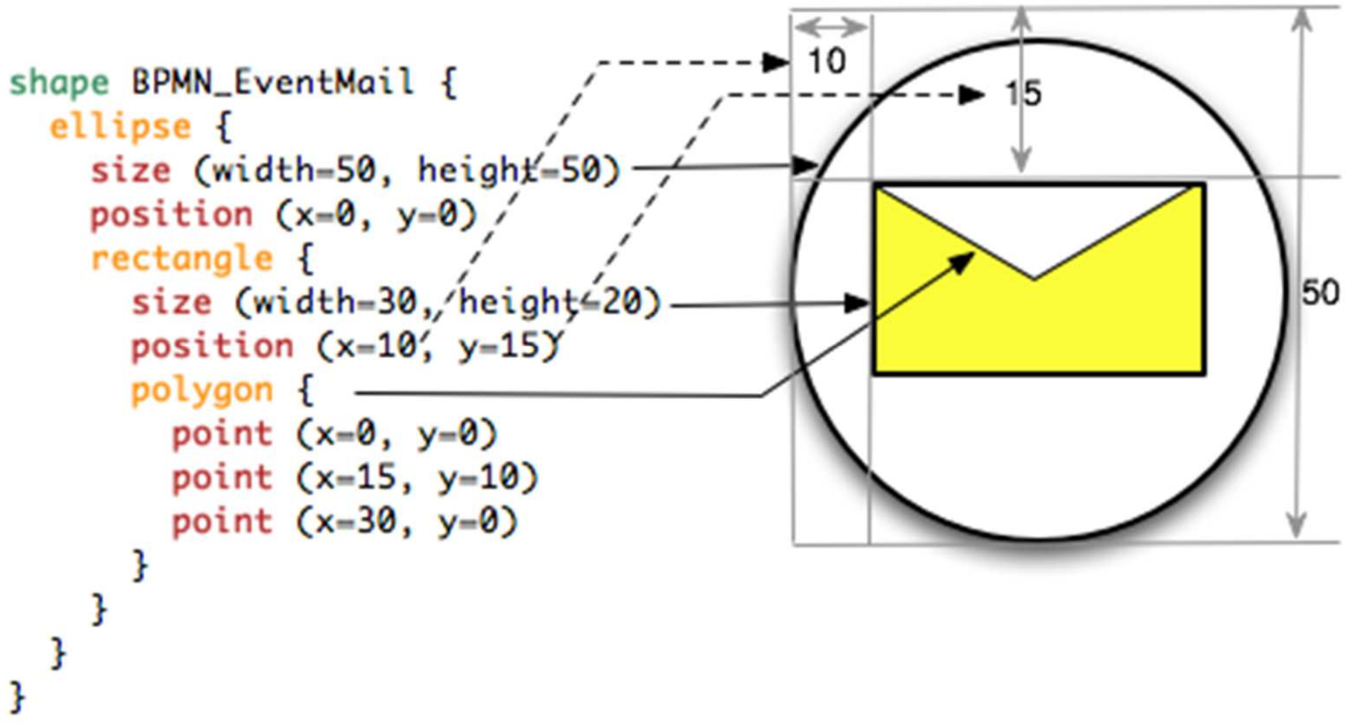
Defining Shapes - with Graphiti

```
// Define general layout
sprayStyle.getStyle(diagram).setProportional(false);
sprayStyle.getStyle(diagram).setStretchH(false);
sprayStyle.getStyle(diagram).setStretchV(false);

// Creating the different figures
// Create a Invisible Rectangle Around the Elements
GraphicsAlgorithm invisibleRectangle = gaService.createInvisibleRe
invisibleRectangle.setStyle(sprayStyle.getStyle(diagram));
invisibleRectangle.setWidth(50);
invisibleRectangle.setHeight(50);

ISprayStyle style_0 = sprayStyle;
Ellipse element_1 = gaService.createEllipse(invisibleRectangle);
ISprayStyle style_1 = style_0;
element_1.setStyle(style_1.getStyle(diagram));
gaService.setLocationAndSize(element_1, 0, 0, 50, 50);
Rectangle element_2 = gaService.createRectangle(element_1);
ISprayStyle style_2 = style_1;
element_2.setStyle(style_2.getStyle(diagram));
gaService.setLocationAndSize(element_2, 10, 15, 30, 20);
element_2.setBackground(gaService.manageColor(diagram,IColorConstant.YELLOW));
List<Point> pointList_1 = new ArrayList<Point>();
pointList_1.add(gaService.createPoint(0, 0, 0, 0));
pointList_1.add(gaService.createPoint(15, 10, 0, 0));
pointList_1.add(gaService.createPoint(30, 0, 0, 0));
Polygon element_3 = gaService.createPolygon(element_2, pointList_1);
ISprayStyle style_3 = style_2;
element_3.setStyle(style_3.getStyle(diagram));
```



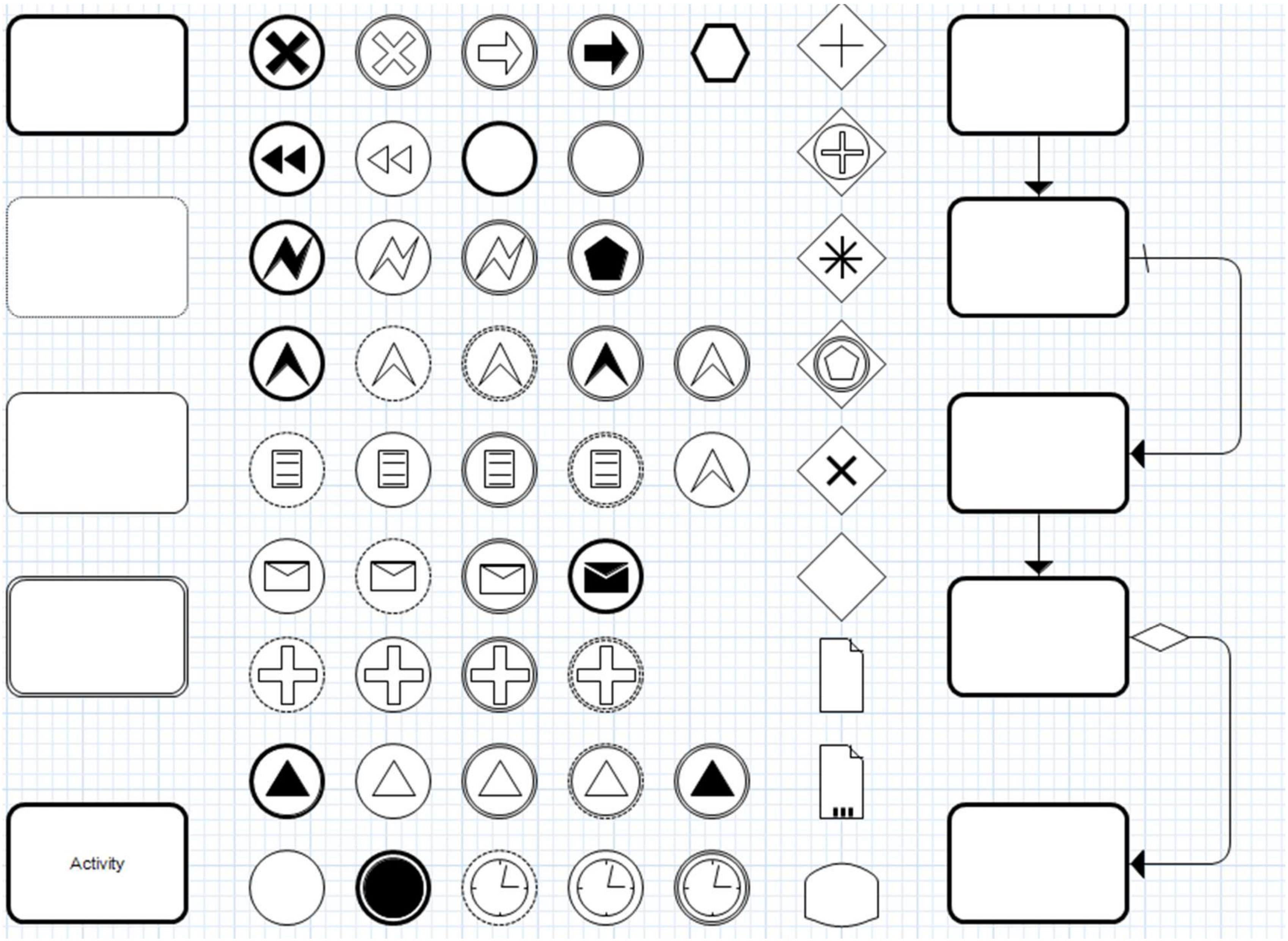
Defining Shapes - with Spray

- Spray provides a simple DSL to define Shapes
  - From primitive nested shapes with properties



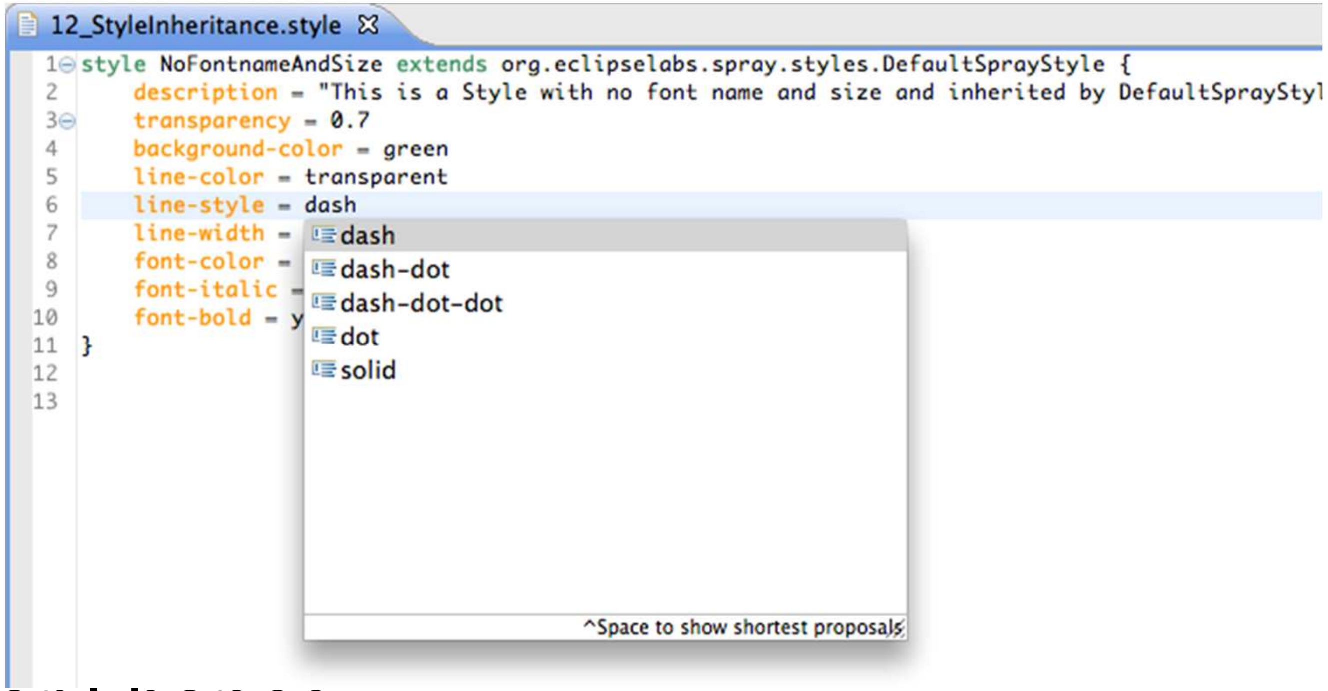
```
shape BPMN_EventMail {
  ellipse {
    size (width=50, height=50)
    position (x=0, y=0)
    rectangle {
      size (width=30, height=20)
      position (x=10, y=15)
      polygon {
        point (x=0, y=0)
        point (x=15, y=10)
        point (x=30, y=0)
      }
    }
  }
}
```

Shapes created with this Shape DSL



Styles

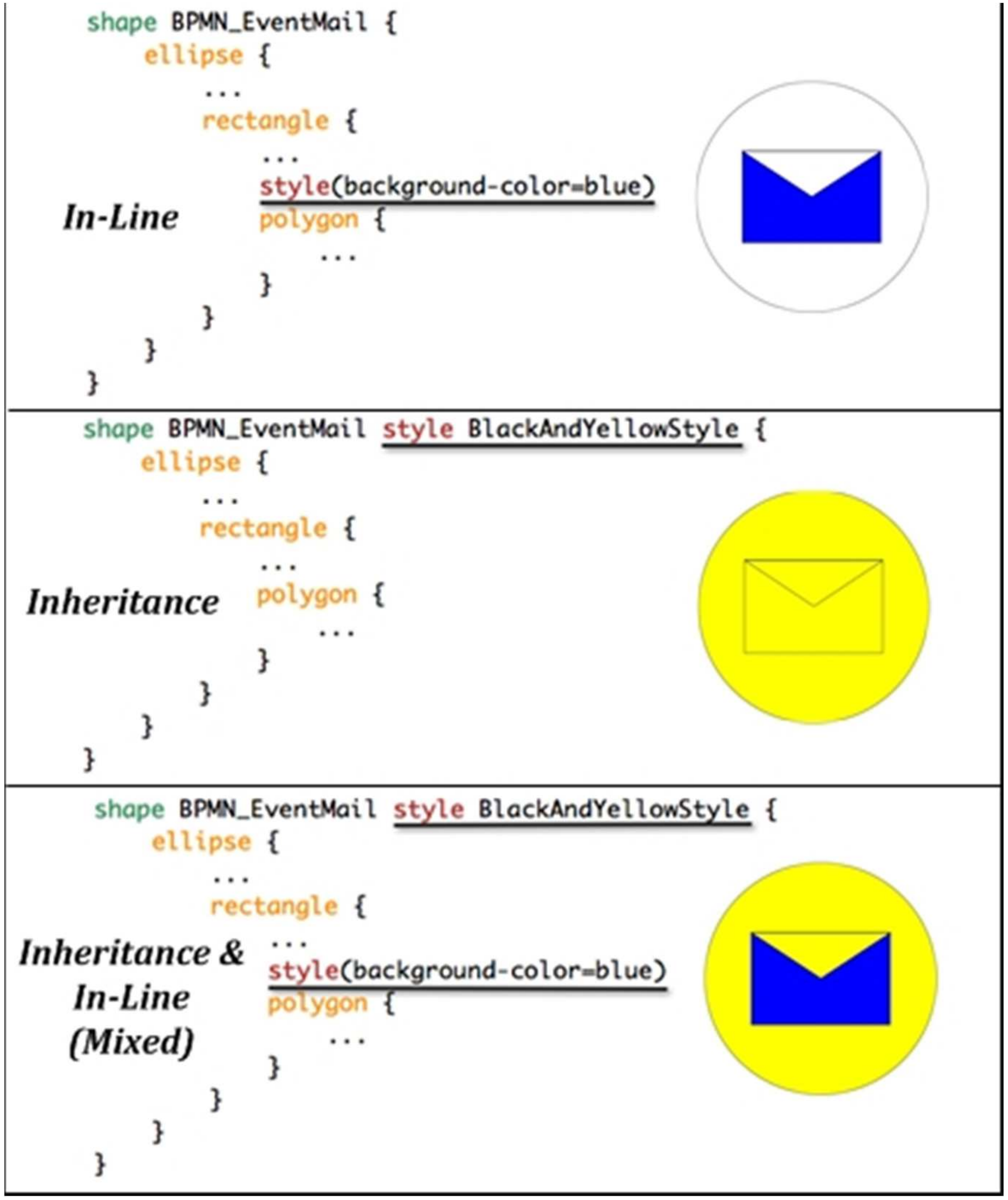
- Style DSL
  - Color
  - Font
  - Line
- Inheritable
- Referrable for Shapes



```
style NoFontnameAndSize extends org.eclipselabs.spray.styles.DefaultSprayStyle {
    description = "This is a Style with no font name and size and inherited by DefaultSprayStyl
    transparency = 0.7
    background-color = green
    line-color = transparent
    line-style = dash
    line-width = 
    font-color = 
    font-italic = 
    font-bold = y
}
```

The Styles DSL

- Inline
- External
- Mixed



```
shape BPMN_EventMail {
    ellipse {
        ...
        rectangle {
            ...
            style(background-color=blue)
            polygon {
                ...
            }
        }
    }
}
```

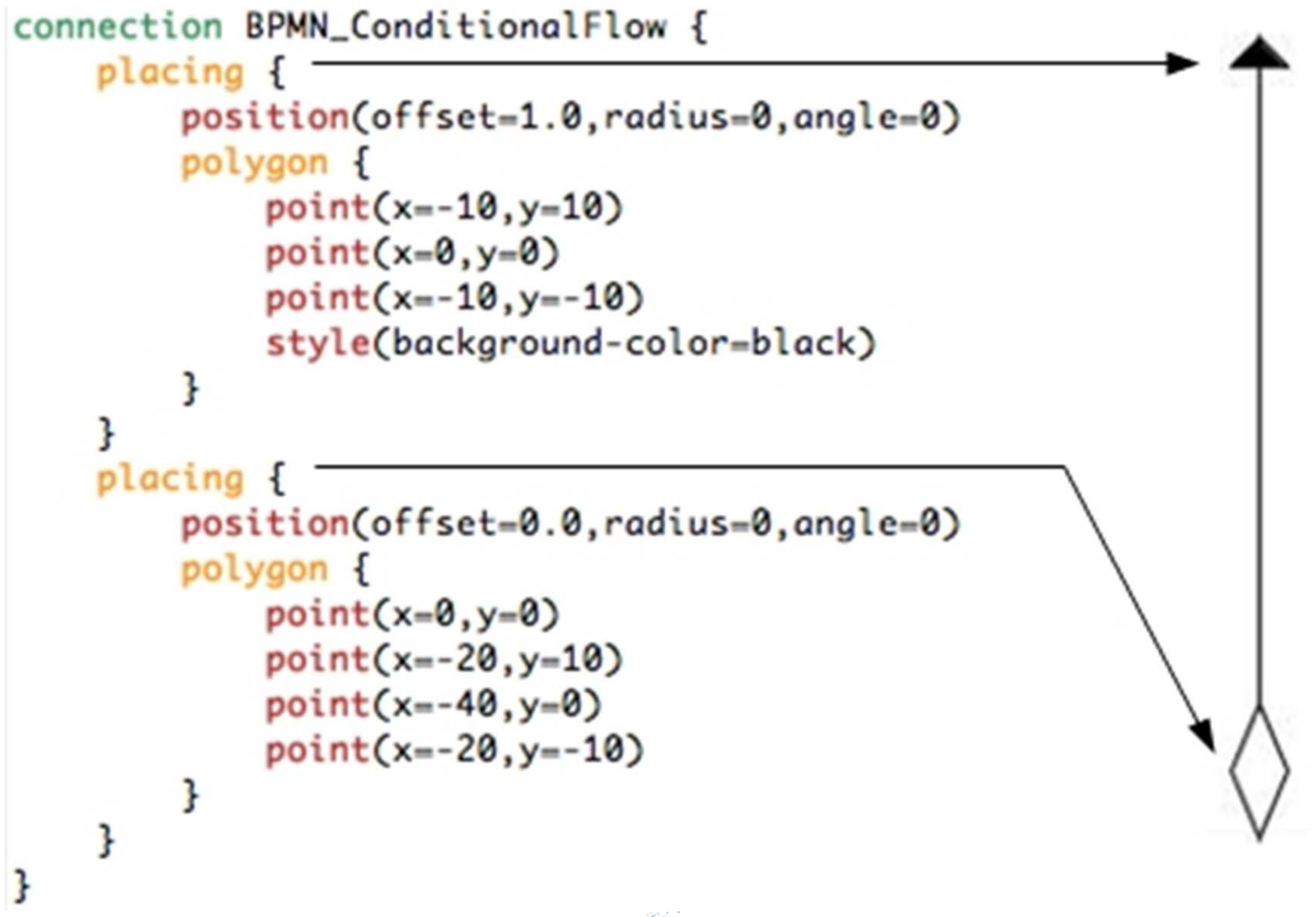
```
shape BPMN_EventMail style BlackAndYellowStyle {
    ellipse {
        ...
        rectangle {
            ...
            polygon {
                ...
            }
        }
    }
}
```

```
shape BPMN_EventMail style BlackAndYellowStyle {
    ellipse {
        ...
        rectangle {
            ...
            style(background-color=blue)
            polygon {
                ...
            }
        }
    }
}
```

Various Styles

- Styles can be reused
- Styles can be inherited
  - Individual attributes can be overridden
- The whole diagram can have a default style
  - Quickly changes the entire look and feel to a CI
- Gradients and shadows in preparation

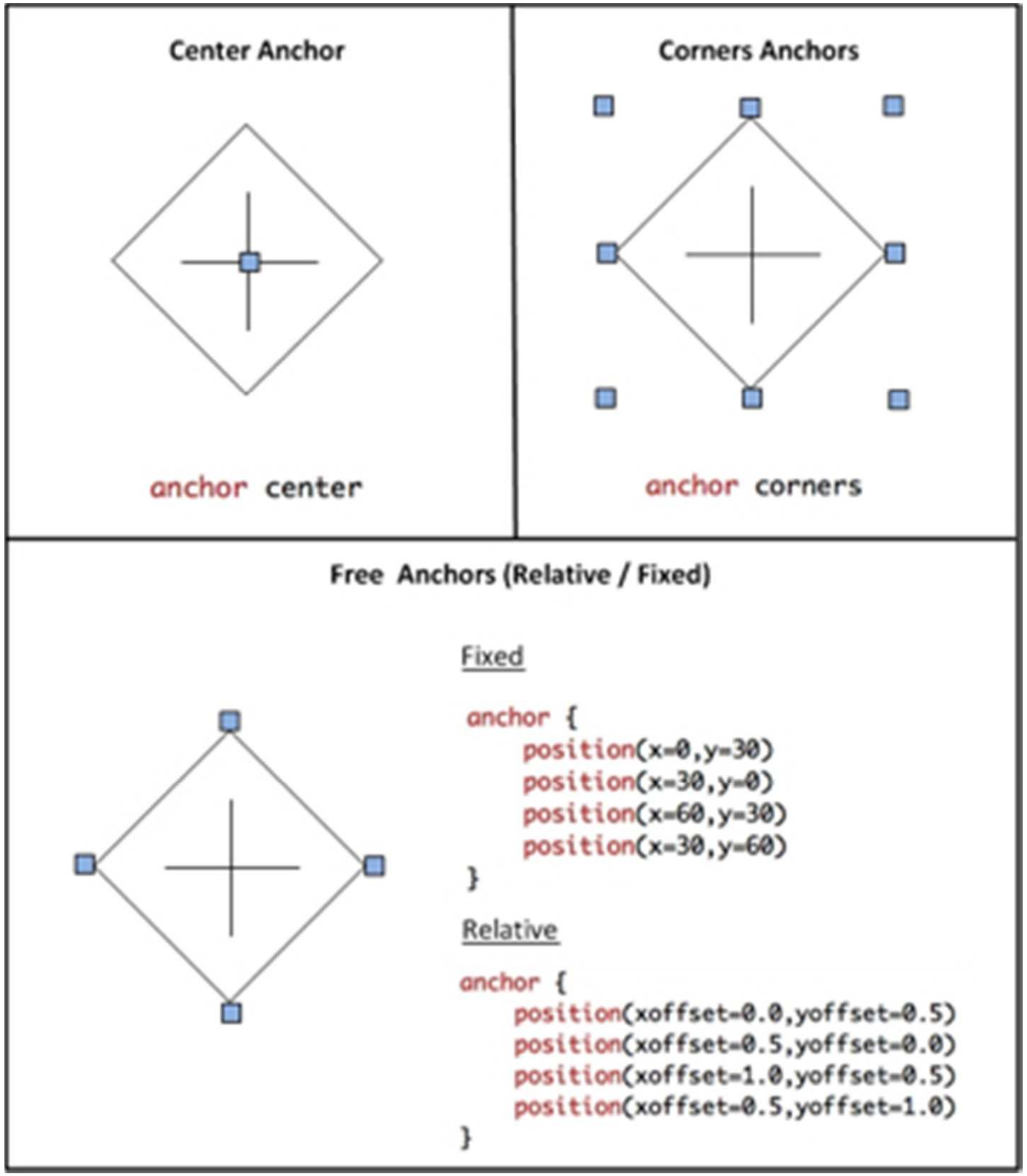
Connections are Shapes



```
connection BPMN_ConditionalFlow {
    placing {
        position(offset=1.0,radius=0,angle=0)
        polygon {
            point(x=-10,y=10)
            point(x=0,y=0)
            point(x=-10,y=-10)
            style(background-color=black)
        }
    }
    placing {
        position(offset=0.0,radius=0,angle=0)
        polygon {
            point(x=0,y=0)
            point(x=-20,y=10)
            point(x=-40,y=0)
            point(x=-20,y=-10)
        }
    }
}
```

Anchor

- An Anchor is the connection point of a connection to a shape
- Its position is a property of the shape
- There are 4 options
  - Center (default)
  - Corners
  - Fixed
  - Relative



```
anchor center
```

```
anchor corners
```

Fixed

```
anchor {
     position(x=0,y=30)
     position(x=30,y=0)
     position(x=60,y=30)
     position(x=30,y=60)
}
```

Relative

```
anchor {
     position(xoffset=0.0,yoffset=0.5)
     position(xoffset=0.5,yoffset=0.0)
     position(xoffset=1.0,yoffset=0.5)
     position(xoffset=0.5,yoffset=1.0)
}
```

Extending the Code

- The code is generated in a way to allow manual extension
  - Generate good code for most situations
  - Program extensions for rare exceptional cases
  - (Extended) Generation Gap Pattern
  - Extensibility of generator using Dependency Injection (Guice)

Our Todo-List

- Clipping
- Resize
- Icons
- Copy-Paste
- Outline
- Underline
- Rapid Button
- Context menu
- Shadows and Glows
- Gradients
- Property Editor
- Text-Support
- Compartments
- Model validation
- Model browser

Future Plans

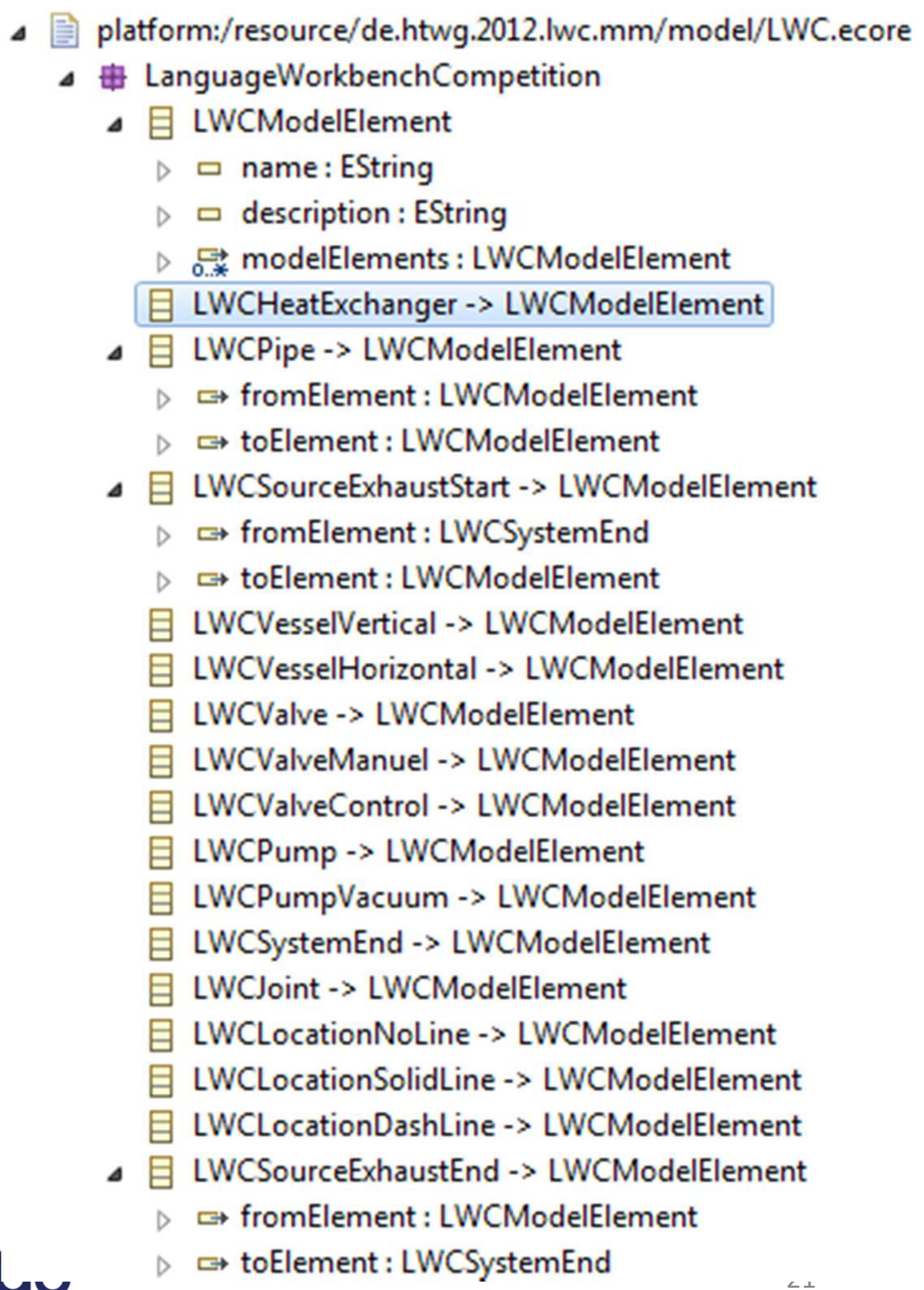
- Complex Diagrams: BPMN, UML, PetriNet
- Evolve DSLs
- Several Build targets
  - Swing
  - Web
- Database support
- Multi-User support

LWC: Piping and Instrumentation

- Graphical Editor
  - Simple Metamodel in EMF
  - Spray (core) DSL
  - Shapes

The Metamodel

- EMF metamodel
  - Can be created with any EMF compliant tool
    - Eclipse tree editor
    - Poseidon for DSLs



```
platform:/resource/de.htwg.2012.lwc.mm/model/LWC.ecore
  LanguageWorkbenchCompetition
    LWCModelElement
      name : EString
      description : EString
      modelElements : LWCModelElement
    LWCHeatExchanger -> LWCModelElement
    LWCPipe -> LWCModelElement
      fromElement : LWCModelElement
      toElement : LWCModelElement
    LWCSourceExhaustStart -> LWCModelElement
      fromElement : LWCSystemEnd
      toElement : LWCModelElement
    LWCVesselVertical -> LWCModelElement
    LWCVesselHorizontal -> LWCModelElement
    LWCValve -> LWCModelElement
    LWCValveManuel -> LWCModelElement
    LWCValveControl -> LWCModelElement
    LWCPump -> LWCModelElement
    LWCPumpVacuum -> LWCModelElement
    LWCSystemEnd -> LWCModelElement
    LWCJoint -> LWCModelElement
    LWCLocationNoLine -> LWCModelElement
    LWCLocationSolidLine -> LWCModelElement
    LWCLocationDashLine -> LWCModelElement
    LWCSourceExhaustEnd -> LWCModelElement
      fromElement : LWCModelElement
      toElement : LWCSystemEnd
```

The Spray (core) DSL

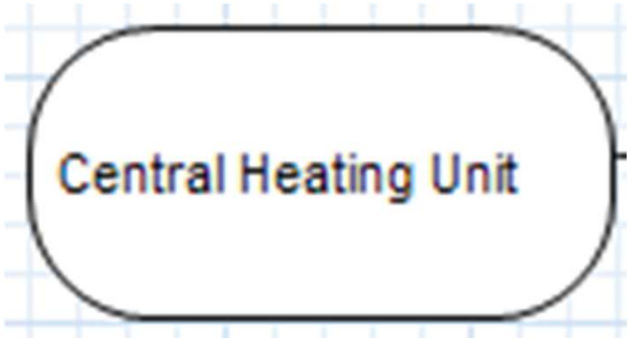
```
diagram LWC for LWCModelElement

class LWCVesselHorizontal alias Horizontal :
      shape LWC_Vessel_Horizontal {
            shapeName = name
      }
behavior {
      create into modelElements palette "Vessels" askFor name;
}


class LWCLocationNoLine alias Noline :
      shape LWC_Location_NoLine {
            shapeName = name.substring(0,name.indexOf(";"))
            valueName = name.substring(name.indexOf(";")+1, name.length)
      }
      behavior {
          create into modelElements palette "Locations" askFor name;
}
```

The Spray (core) DSL (Connections)

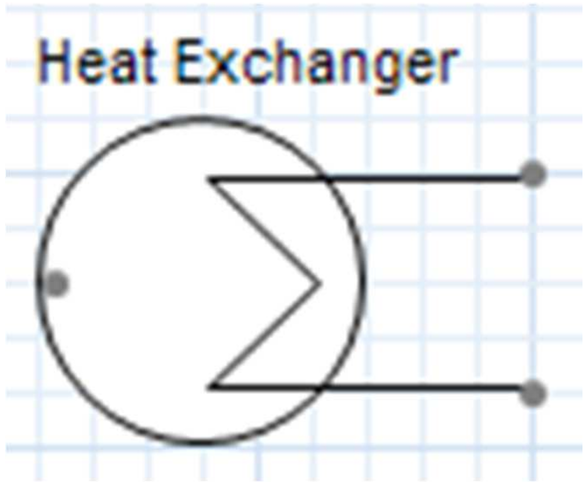
```
class LWCPipe :
      connection LWC_Pipe() {
            from fromElement;
            to toElement;
      }
behavior {
      create into modelElements palette "Connections";
}

class LWCSourceExhaustStart :
      connection LWC_Source_Exhaust () {
            from fromElement;
            to toElement;
      }
      behavior {
            create into modelElements palette "Connections";
      }
}
```

The Shape DSL

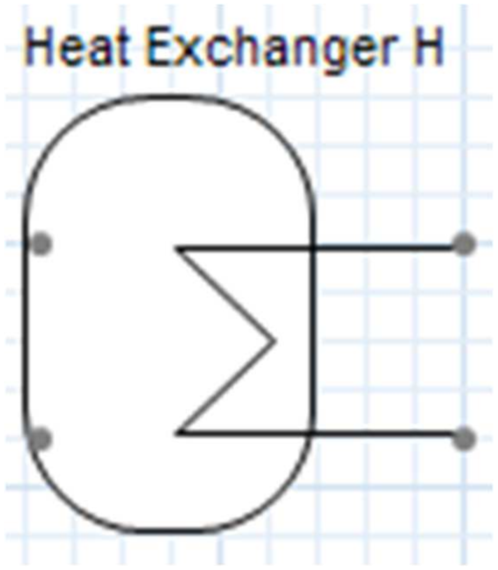
```
shape LWC_Vessel_Horizontal ( java.lang.String shapeName ) {
      rounded-rectangle {
            position(x=0,y=0)
            size(width=60,height=120)
            curve(width=50,height=50)
            text {
                  size(width=60,height=20)
                  position(x=3,y=50)
                  value=shapeName
            }
      }
}
```



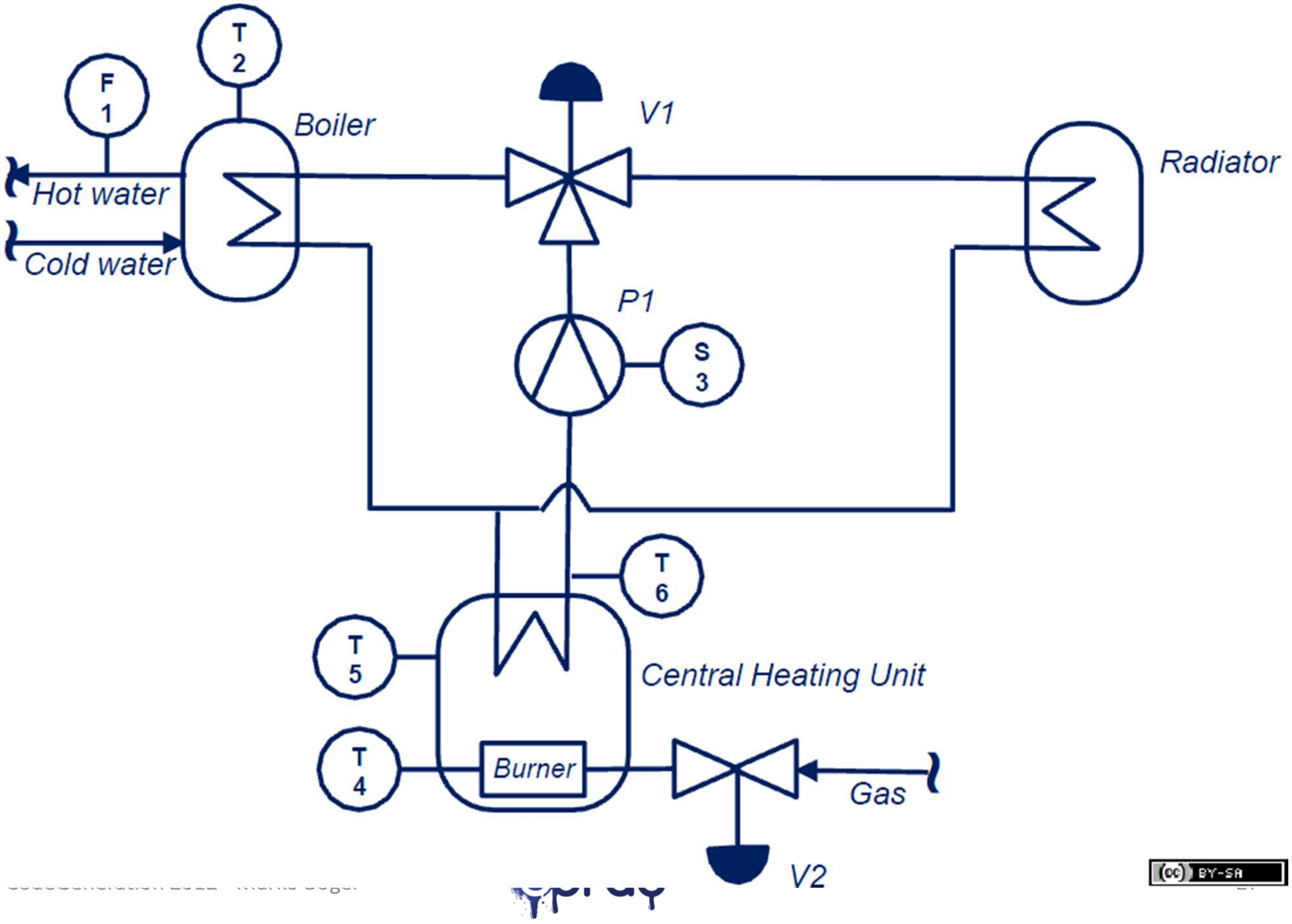
```
shape LWC_HeatExchanger (java.lang.String shapeName) {
      ellipse {
            position(x=0,y=20)
            size(width=60,height=60)
      }
      polyline {
            point(x=90,y=70)
            point(x=30,y=70)
            point(x=50,y=50)
            point(x=30,y=30)
            point(x=90,y=30)
      }
      text {
            position(x=0,y=0)
            size(width=100,height=20)
            value=shapeName
      }
      anchor {
            position(x=0,y=50)
            position(x=90,y=30)
            position(x=90,y=70)
      }
}
```



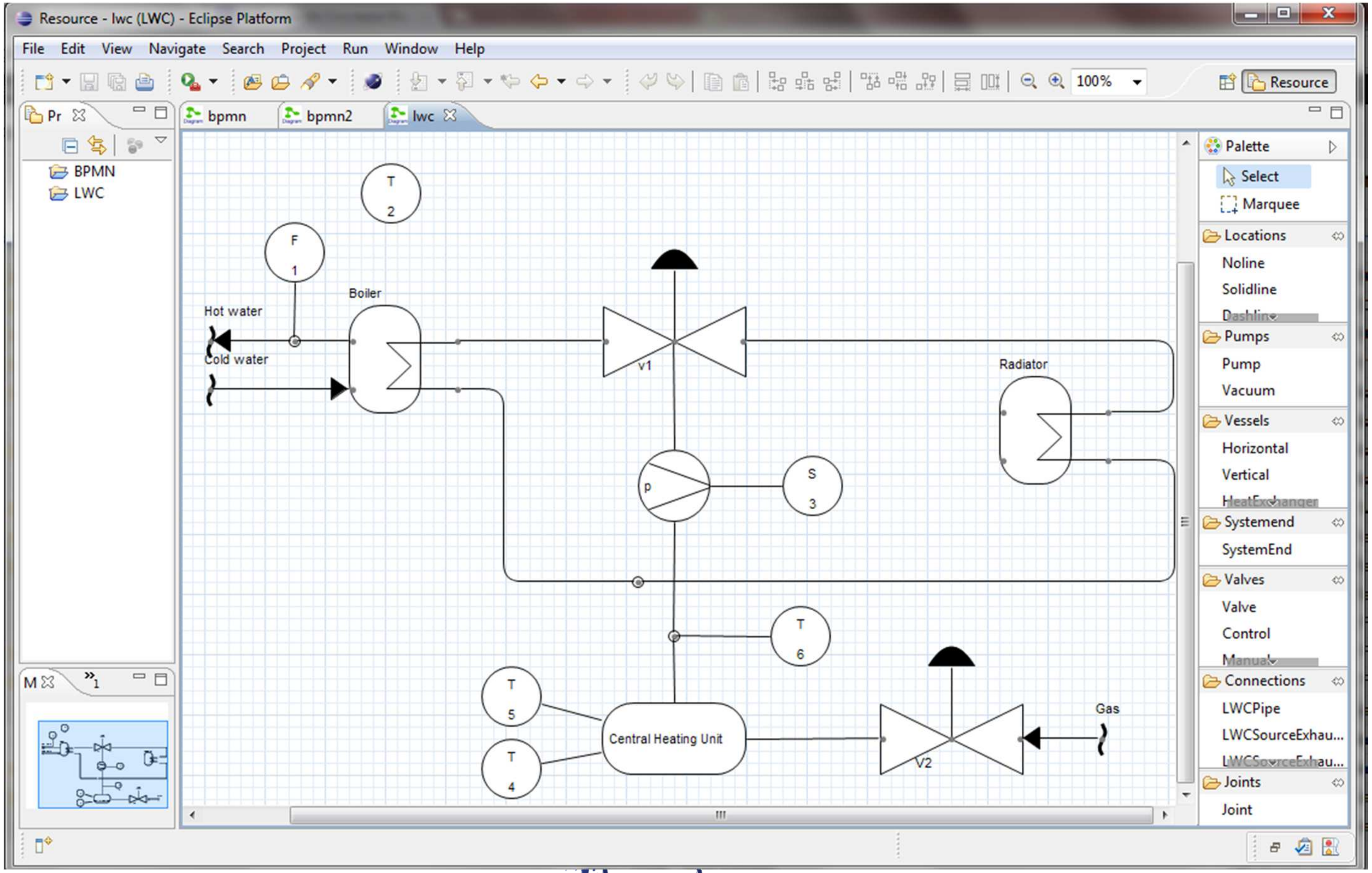
```
shape LWC_HeatExchanger (java.lang.String shapeName) {
	rounded-rectangle {
		position(x=0,y=20)
		size(width=60,height=90)
		curve(width=50, height=50)
	}
	polyline {
		point(x=90,y=90)
		point(x=30,y=90)
		point(x=50,y=70)
		point(x=30,y=50)
		point(x=90,y=50)
	}
	text {
		position(x=0,y=0)
		size(width=100,height=20)
		value=shapeName
	}
	anchor {
		position(x=0,y=50)
		position(x=0,y=90)
		position(x=90,y=50)
		position(x=90,y=90)
	}
}
```



The Challenge



Our Proposal (at this time)



Costs

- Metamodel: 29 lines
- Spray (core) DSL: 135 lines
- Shape DSL: 303 lines
- Style: 0 lines

- Total: 476 lines
  - > hours or days
- Generated Code: 240 Files, ~12.000 lines
  - > weeks or months