# CVL Tool 2.0 – General Plan

## Source Code

* The source code will be stored as Eclipse projects in a private gitHub repository.
* The tool is programmed using Java.
* Much of the functionality will be in the "CVL Library", a Java library.
* Editor code will be in separate packages that depend on the CVL Library.

## Editors

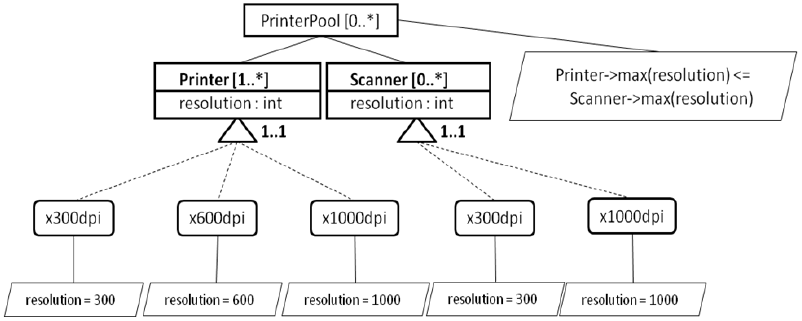
* The editors will be programmed from scratch using Java and Swing.
  + Is Swing the optimal graphics library?
* There will be 4 different editors.

Structure:

* We want explicit storing of diagram information.
  + Extra care to consistency between the model and the diagram.
    - Storing the diagram information weaved into the model as an eAnnotation?
* WYSIWYG
  + Exclude copy-paste of more than one object
  + Drag and drop as a pure diagram placement changing
  + Delete more than one object
  + Allow intermediate inconsistent and incomplete models
    - Restricted by the palette
    - Can the EMF-model in memory be inconsistent and incomplete?
    - Maybe make a more lenient meta-model?
    - Consistency check reports problems with the model.

### 1. CVL VSpec tree-editor

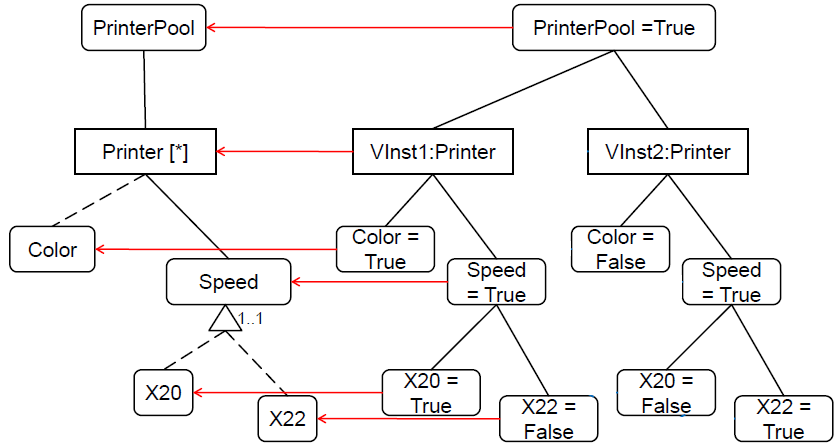
Allows the editing of VSpec trees.



**Figure: An example VSpec tree diagram**

### 2. CVL VSpec resolution-editor

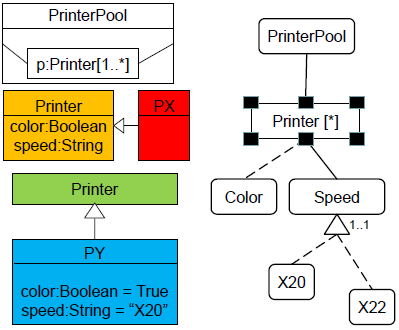
Allows the editing of an instance of a VSpec tree



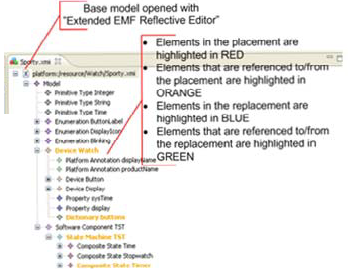
**Figure: An example VSpec instance on the right. Its bindings to the VSpec tree is shown with red lines.**

### 3. CVL VSpec realization-editor

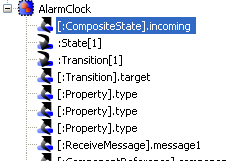
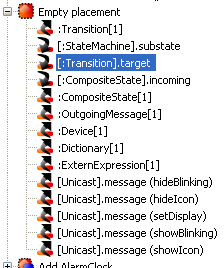
Allows the editing of Vspec realization models. These extend, in a sense, the Vspec trees.



**Figure: An example Vspec realization model on the left and in the middle. Its bindings to the Vspec tree is shown with green lines. The far left side shows an editor enhanced with CVL.**

****

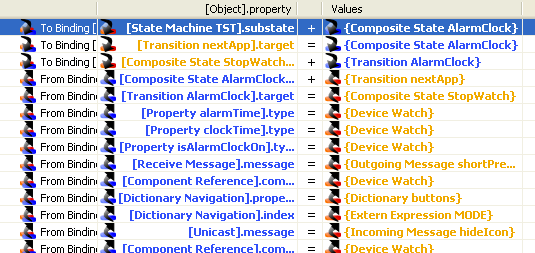
**Figure: EMF-base model opened with Extended EMF Reflective Editor**

****

**Figure: Placement and replacement tree-editors**

### 4. placement-replacement binding editor

Allows the binding between placements and replacements. We currently have no idea how to improve this editor.



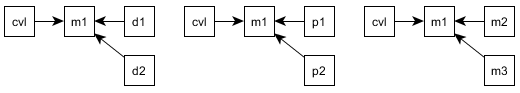
**Figure: Editor for binding placements to replacements**

How to make it better, suggestions:

* Include in the name: Both the name and the name of the reference it records, and where it points to??
* Process improvement: Give informative names when modeling.
* We want this to be more robust ordering of:
  + Creation of the default binding
  + Opening of the binding editor
  + Clicking on the binding-editor tab
* During realization when dealing with fragments, one must manually inspect every binding every time.
  + One should see how this applies to the model.

# The multiple-models and multiple-files problem

## Problem



**Figure: Multiple models pointing to the base-model. 1) Diagrams, 2) profiles and 3) other models.**

As the figure above illustrates, there are three related problems that are instances of the same basic problem.

CVL can only point to one model. What about 1) diagrams that are often partial that point to the base model? 2) What about profiles that point to the base model? 3) What about other models pointing to the base model? For the third situation, we might have further nestings of all three cases.

Solution:

* Will be able to point to many models, even in many files. (need change to the meta-model)
  + Is there a need for a container?
* Placements must be able to span many models and files.

## Idea for direction towards a solution

Feed all relevant models to the CVL tool, and have it resolve absolutely everything.

## CVL Execution model