Combinatorial Interaction Testing with CitLab

Angelo Gargantini

Università di Bergamo - Italy

http://cs.unibg.it/gargantini

Joint work with Paolo Vavassori

CitLab in brief

http://code.google.com/a/eclipselabs.org/p/citlab/

Language for CIT problems

1. with a precise formal semantics and a grammar by Xtext
2. A textual editor integrated in the eclipse IDE

Set of tools

3. for importing/exporting CIT problems
4. for generating test suites (by using external tools)

Framework

5. based on the Eclipse Modeling Framework (EMF), library to manipulate combinatorial problems in Java
6. A rich collection of Java utility classes and methods
7. A rich collection of benchmarks

INSTALLING CitLab

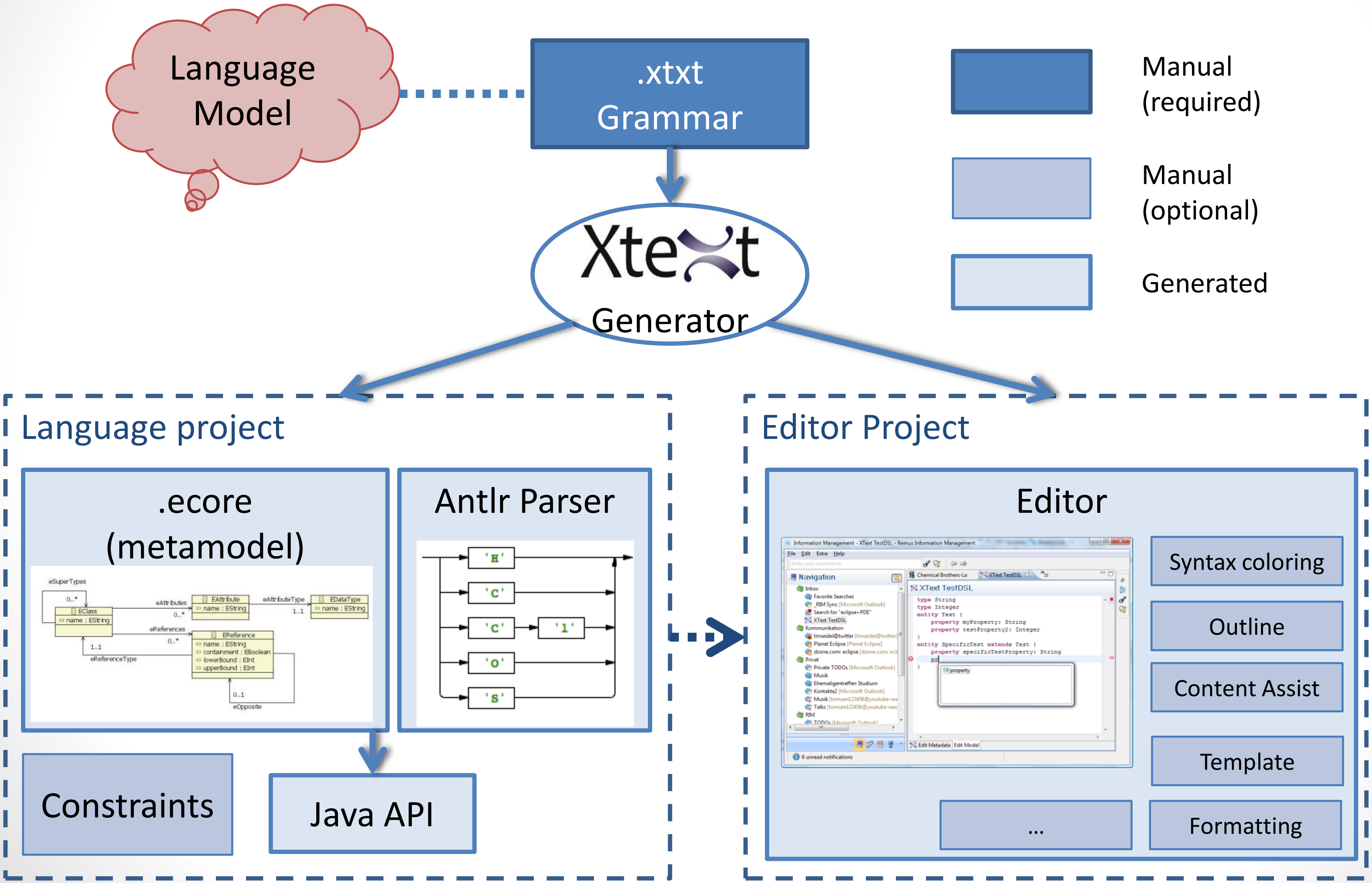
As eclipse plugin

http://svn.codespot.com/a/eclipselabs.org/citlab/CitLabPlugins/

CitLab EDITOR

DEMO

CitLab internals – using XTEXT x DSL



Language Model

.xtxt Grammar

Xtext Generator

■ Manual (required)

▨ Manual (optional)

□ Generated

Language project

.ecore (metamodel)

Antlr Parser

Constraints

Java API

Editor Project

Editor

```
type String
type Integer
entity Test {
    property myProperty: String
    property testProperty2: Integer
}
entity SpecificTest extends Test {
    property specificTestProperty: String
    pr
}
```

Syntax coloring

Outline

Content Assist

Template

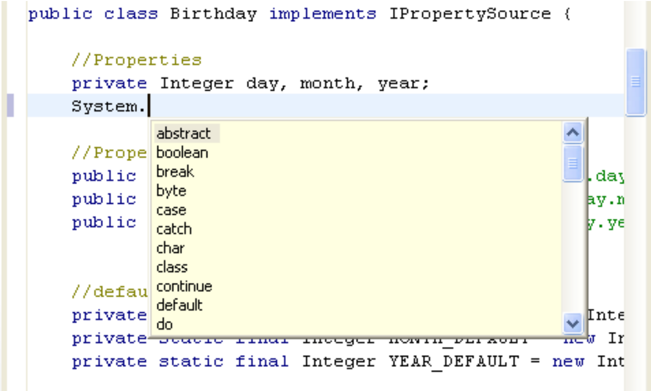
…

Formatting

Editor features

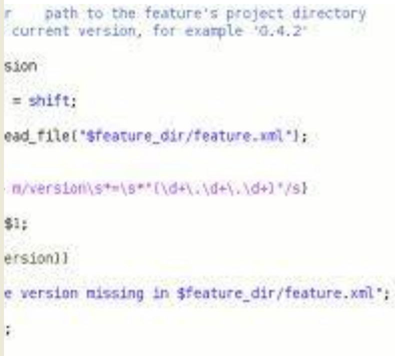
- Syntax Coloring
- Content Assist
- Template Proposals
- Rich Hover
- Rename Refactoring
- Quick Fixes
- Outline

- Folding
- Hyperlinks for all Cross References
- Find References
- Toggle Comment
- Mark Occurrences
- Formatting

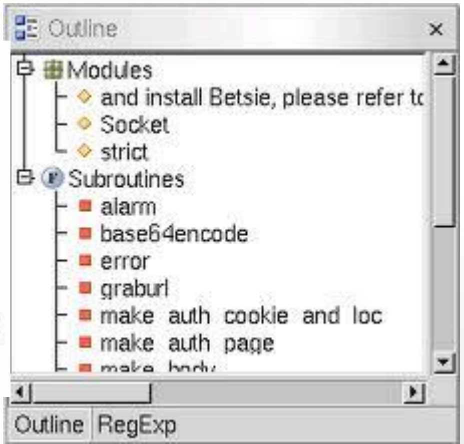


```
public class Birthday implements IPropertySource {

    //Properties
    private Integer day, month, year;
    System.|

    //Prope
    public                                           .day
    public                                           ay.m
    public                                           y.ye


    //defau
    private                                          Inte
    private                                          new Ir
    private static final Integer YEAR_DEFAULT = new Int
```

```
abstract
boolean
break
byte
case
catch
char
class
continue
default
do
```



```
r    path to the feature's project directory
 current version, for example '0.4.2'

sion

 = shift;

ead_file("$feature_dir/feature.xml");

 m/version\s*=\s*"(\d+\.\d+\.\d+)"/s)

$1;

ersion)]

e version missing in $feature_dir/feature.xml";

;
```

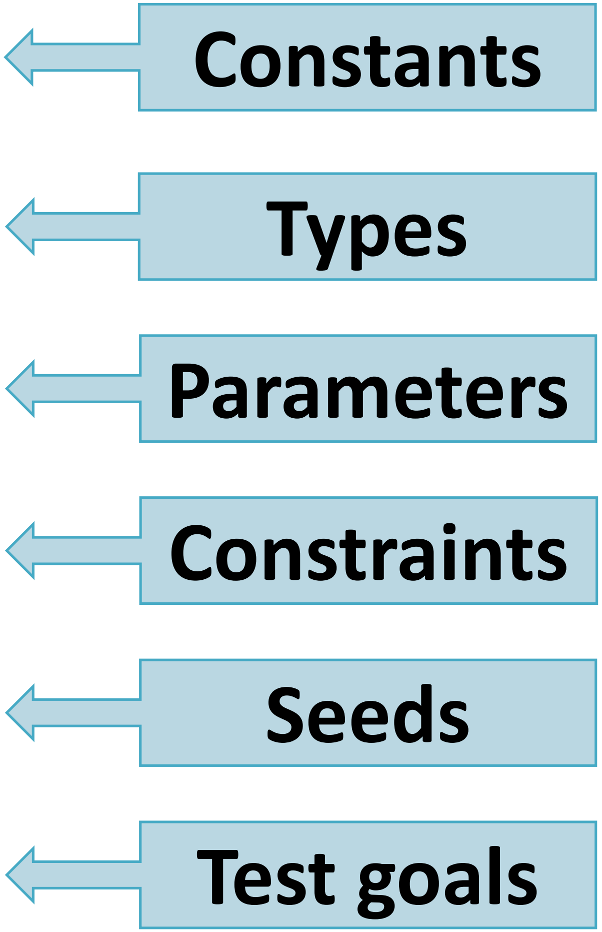


Angelo Gargantini -Combinatorial Interaction Testing with CitLab

MODELLING COMBINATORIAL PROBLEMS

CitLab Language in a glance

```
Model Model
Definitions
...
end
Types:
...
end
Parameters:
...
end
Constraints:
# ... #
end
Seeds:
# ... #
end
TestGoals:
# ... #
end
```



Constants

Types

Parameters

Constraints

Seeds

Test goals

Example:
A family of phones, that can have several types of cameras, display,…

Parameters and their types

- To describe a combinatorial problem would be sufficient to specify the number of variables and their cardinality.
- CITLAB language forces the designer to name parameters and to specify their types by listing all the values in their domain.
- **Choice:** explicit parameter names to facilitate the modeling of real systems and to ease the specification of constraints and seeds

Enumerative for parameters that can take a value in a set of symbolic constants.

the display of the cell phone can be colored (with 16 or 8 millions colors) or black and white,

```
Enumerative display { 16MC 8MC BW };
```

Angelo Gargantini -Combinatorial Interaction Testing with CitLab

Parameters (2)

Boolean for parameters that can be either true or false.

the phone can have an email viewer

```
Boolean emailViewer;
```

Numerical values in a range for parameters that take any value in an integer range.

Phones have a number of lines between 10 and 30, but only every 5 is valid

```
Range textLines [ 10 .. 30 ] step 5;
```

A list of Numbers for parameters that take any value ina set of integers.

The phone has been produced in 2012 and 2013

```
Numbers year {2012 2013};
```

Angelo Gargantini -Combinatorial Interaction Testing with CitLab

Named Types

- Types can be defined with their name in the Types section to be used in parameters declaration
  - Instead of an “anonymous” type.

The phone can have two cameras (front and rear) of different type.

```
Types:
  EnumerativeType cameraType { 2MP 1MP NOC };
end
Parameters:
  Enumerative rearCamera : cameraType;
  Enumerative frontCamera : cameraType;
  ...
end
```

Advantages: the use of named parameter types to make more compact and more maintainable the models in case many parameters share the same domain.

Definitions

- Sometimes it is useful to have constants

If the phone has an email viewer bigger then a threshold 27

- For (numerical) constants, to be used in constraints, …

```
Number threshold = 27;
```

Constraints

- In CITLAB, we adopt the language of propositional logic with equality and arithmetic to express constraints
- General Form (GF) constraints
  - propositional calculus and Boolean operators

    ```
    a or b => c and d
    ```

  - equality and inequality

    If the phone has an email viewer then

    ```
    # emailViewer==true  => textLines>=threshold #
    ```

  - arithmetic over the integers
  - relational and arithmetic operators for numeric terms

    ```
    # textLines >= threshold + 10 #
    ```

- A valid test must satisfy all the constraints

Angelo Gargantini -Combinatorial Interaction Testing with CitLab

Seeds

- The testers can also force the inclusion of their favorite test cases by specifying them as seed tests.
  - They can be tests generated by other criteria
  - Critical complete combinations
- The seeds must be included in the generated test set without modification
- CITLAB considers only complete seeds, i.e., seeds that assign a valid value to each parameter

```
Seeds:
# emailViewer=false , display=display.16MC ,
frontCamera=cameraType.NOC ,year=2012 ,
rearCamera=cameraType.2MP , textLines=30 #
end
```

- Partial seeds? Critical partial combination the tester would like to be included in the test suite

Test Goals

- Critical situation that must be tested
- Predicates that must be covered by some tests
- Test goals can be again in GF (as the constraints):

```
// the display has at least threshold lines
# textLines>=threshold #
```

Model validation

- XTEXT provides several levels of validation for the defined language
  - the user can specify additional constraints for the model by providing validation fragments

- Validation rules:
- Every seed is complete
- Assignment to range is correct
- No seed can violate any constraint
- ….

TEST GENERATION

Test generation

- CITLAB does not include in itself generators. Currently supports the following test generators, each defined as generator plugin:
  - AETG is a plugin developed by students following the pseudo code for the greedy algorithm of AETG.
  - IPO is a plugin developed by us following the pseudo code for IPO.
  - Random is a simple random algorithm that adds new randomly built tests until all the n-wise combinations are covered.
  - ACTS is an external test generator tool developed by the NIST.
  - CASA is an external tool for test generation based on simulated annealing by Myra Cohen and colleagues.
  - ATGT_SMT is an external tool combining heuristics and SMT solving.

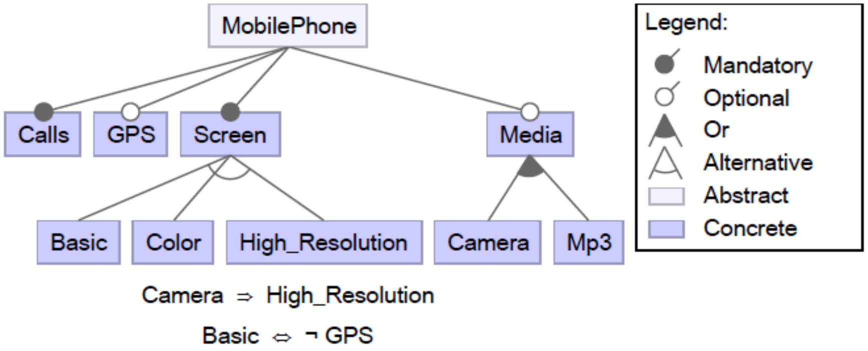
- Some support constraints, seeds, …

IMPORTING/EXPORTING

Importers and Exporters

- For test suites
  - To excel
- For models
  - Feature models
  - See our IWCT 13 paper, new Friday

Calvagna, Gargantini, Vavassori,
Combinatorial Testing for Feature Models Using CitLab


MobilePhone
Calls
GPS
Screen
Media
Basic
Color
High_Resolution
Camera
Mp3
Camera ⇒ High_Resolution
Basic ⇔ ¬ GPS
Legend:
Mandatory
Optional
Or
Alternative
Abstract
Concrete