Textual, executable, translatable UML

Gergely Dévai, Gábor Ferenc Kovács, Ádám Ancsin

Eötvös Loránd University, Budapest, Hungary
Executable UML

All aspects of the software, from structure to behavior, are modeled.

Models can be executed, debugged without the involvement of the runtime platform.
Executable UML toolchain

Developers

Refactoring tool

Compare & merge tools

Editor

Model interpreter, debugger

Test cases

Search tool

Test integration

Model compiler

Implementation language (C++, Java ...)

Platform information

Code repository

Many nontrivial tooling issues...
Executable UML toolchain

Can we make this more lightweight?

Many non-trivial tooling issues...

Implementation language (C++, Java ...)
Language embedding

In an existing (host) language create an API providing the constructs of a new (embedded) language.

Possible reuse of the compiler, run-time, tooling of the host language.

Idea: Embed UML into Java.
Toolchain with language embedding

Developers

Editor

Search tool

Compare & merge tools

Refactoring tool

Code repository

Visualization

Test cases

Test integration

Model compiler

Model export

Platform information

Implementation language (C++, Java ...)

Massive reuse of the host language tooling...
txtUML

- Textual, executable, translatable UML
- Prototype implementation
  - Classes, state machines, action language
  - Export to Ecore UML format – visualization in Papyrus UML editor
  - Proof-of-concept C++ code generator
- http://txtuml.inf.elte.hu
Example

class Machine extends ModelClass {
    /* ... */
}
class User extends ModelClass {
    /* ... */
}
class Usage extends Association {
    @One Machine usedMachine;
    @Many User userOfMachine;
}
class ButtonPress extends Signal {}

class Off extends State {
    /* ... */
}
class On extends State {
    public void entry() {
        /* ... */
    }
    public void exit() {
        /* ... */
    }
}

@From(Off.class) @To(On.class) @Trigger(ButtonPress.class)
class SwitchOn extends Transition {
    public void effect() {
        /* ... */
    }
}

void doWork() {
    Action.log("User: starting to work...");
    Machine myMachine =
        Action.selectOne(this, Usage.class, "usedMachine");
    Action.send(myMachine, new ButtonPress());
}
Execution, debugging

*txtUML* code is Java: You can run and debug it with your favorite tools.

The *txtUML* API implements the run-time semantics of modeling entities.
Visualization

```
Machine
+ modelNumber: Int [1]
  + switchOn
  + switchOn_trigger
  - off
  - initial
  + switchOff
  + switchOff_trigger
User
+ name: String [1]
  + doWork()
```

txtUML models can be exported to Ecore UML2, and visualized in the Papyrus editor.

Implementation uses Java reflection and AspectJ.
Compilation

```cpp
struct Machine {
    std::vector<User*> userOfMachine;
    enum state {
        state_Init,
        state_Off,
        state_On
    } state current_state;
    /* ... */
};
```

```cpp
class Machine extends ModelClass { /* ... */ } 
class User extends ModelClass { /* ... */ } 
class Usage extends Association {
    @One Machine usedMachine;
    @Many User userOfMachine;
}
```

txtUML source code

generated C++ code

Ecore UML2 model
Summary

- **textual:**
  UML embedded in Java

- **executable:**
  Can be run and debugged with standard Java tools

- **translatable:**
  Can be converted to diagrams and compiled to implementation languages

http://txtuml.inf.elte.hu

Thank you for the attention!