

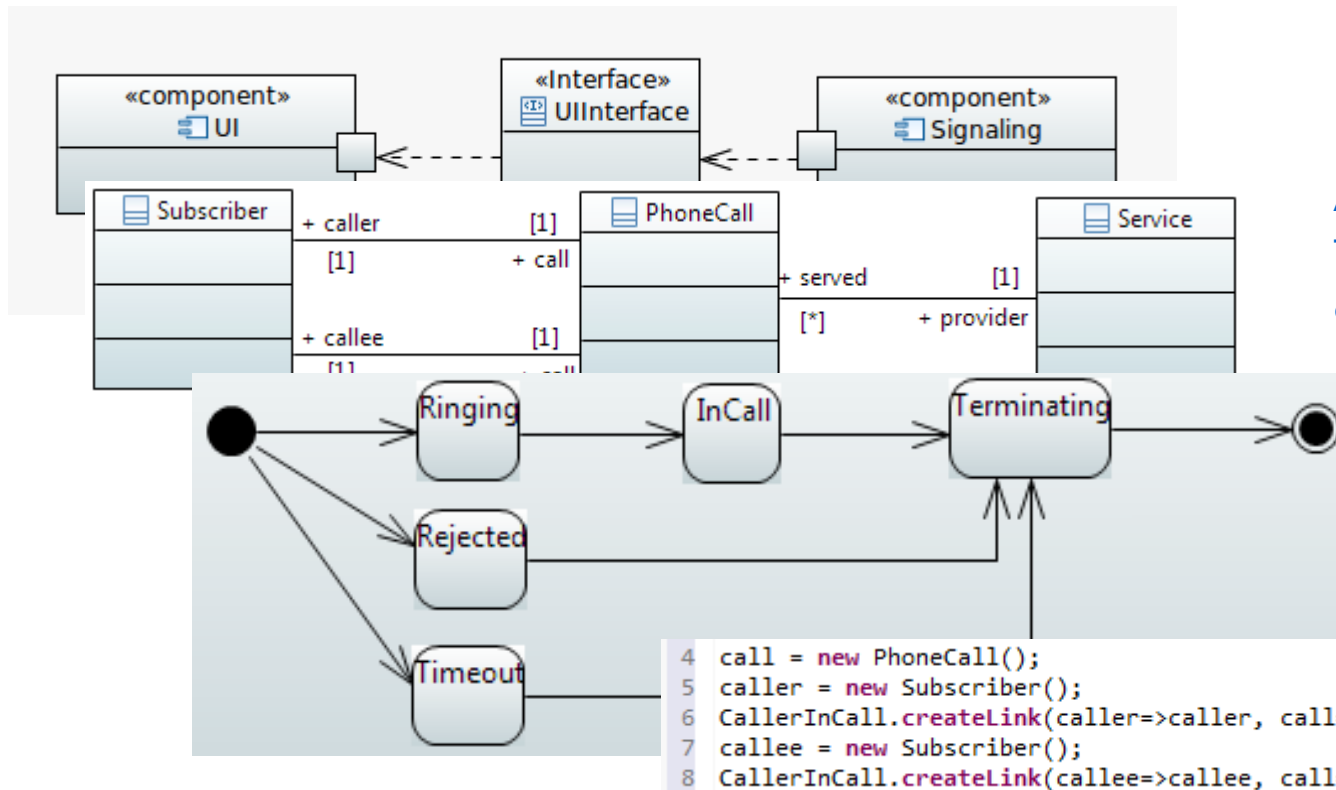
Textual, executable, translatable UML

Gergely Dévai, Gábor Ferenc Kovács, Ádám Ancsin



Eötvös Loránd University, Budapest, Hungary

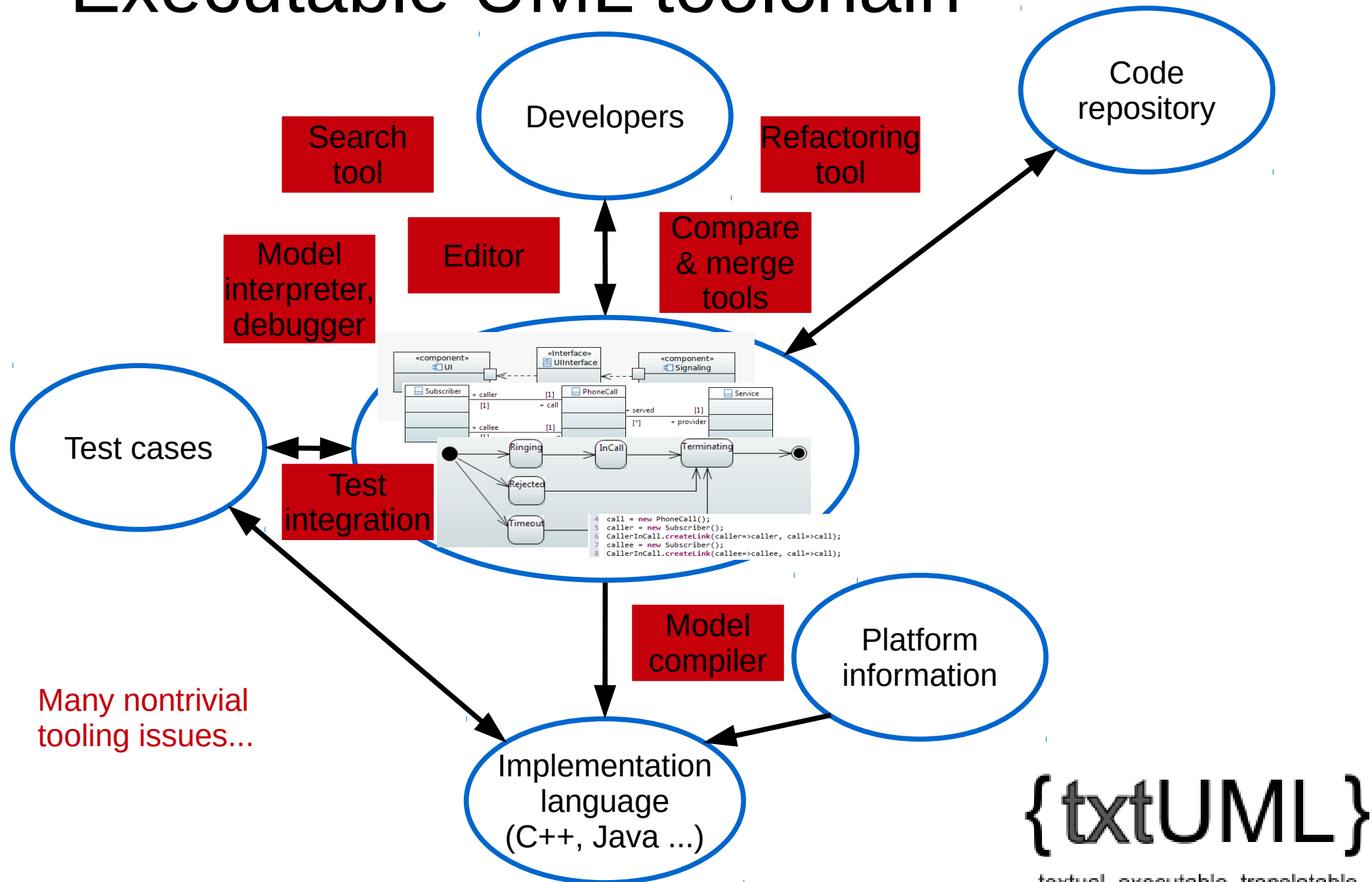
Executable UML



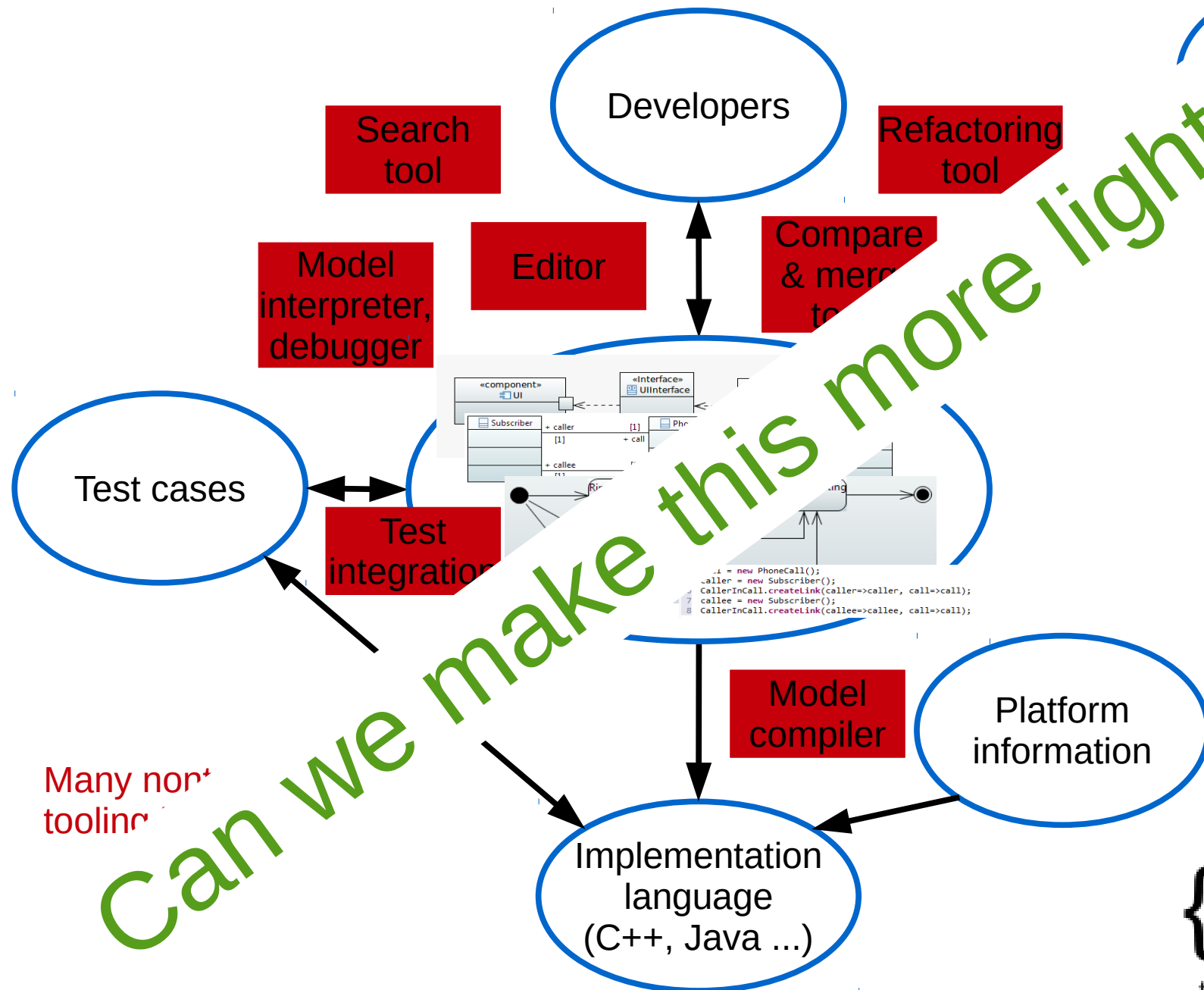
All aspects of the software, from structure to behavior, are modeled.

Models can be executed, debugged without the involvement of the run-time platform.

Executable UML toolchain



Executable UML toolchain



Many non+ toolinc

Can we make this more lightweight?

{txtUML}

textual, executable, translatable

Language embedding

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

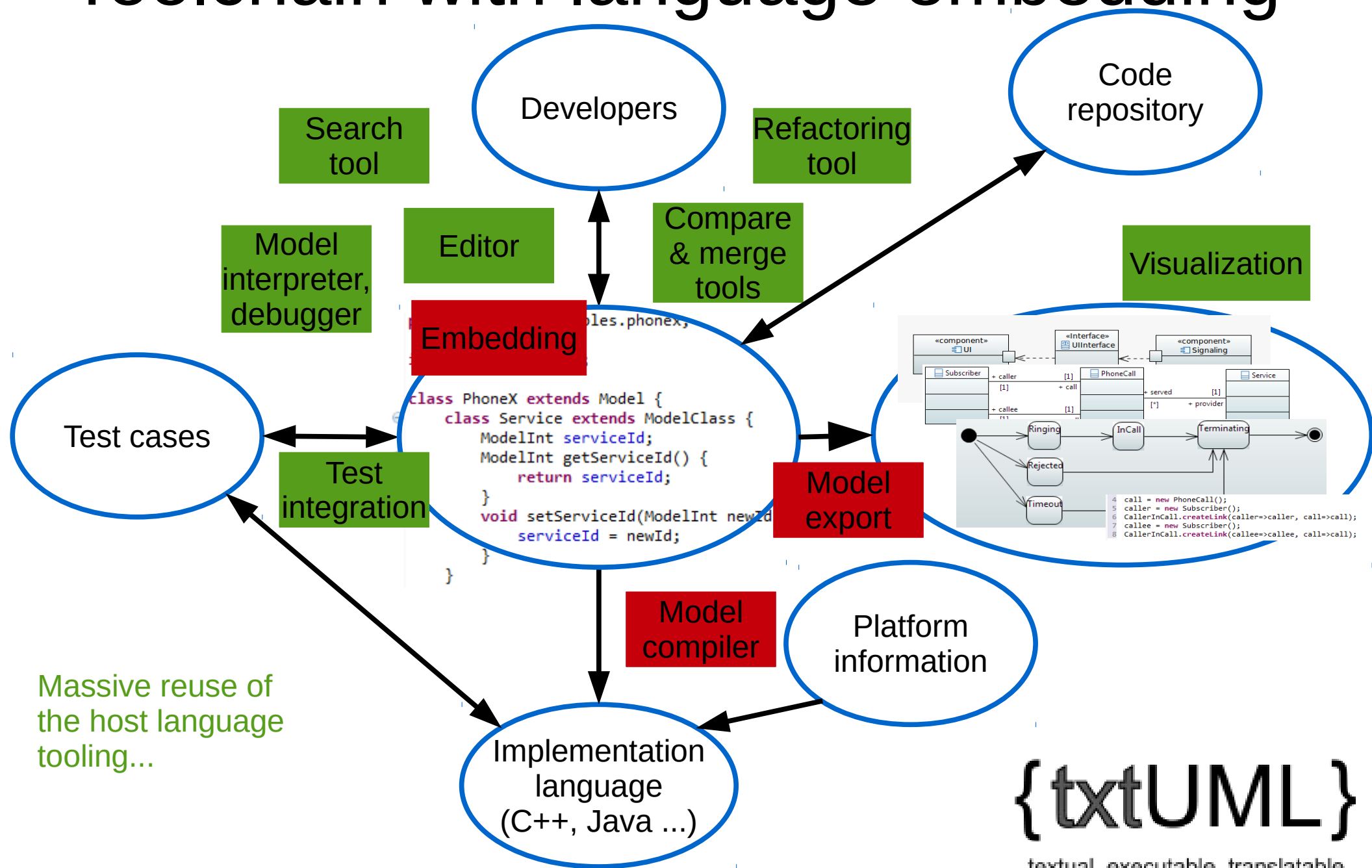
Possible **reuse** of the
compiler, run-time, tooling
of the **host language**.

Idea: Embed UML into Java.

{txtUML}

textual, executable, translatable

Toolchain with language embedding



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>

{txtUML}

textual, executable, translatable

Example

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

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

```
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() { /* ... */ }
}
```

{txtUML}

textual, executable, translatable

Execution, debugging

Debug

Example (3) [AspectJ/Java Application]

- txtuml.examples.example1.Example at localhost:64259
 - Thread [main] (Suspended (breakpoint at line 45 in ExampleModel\$User))
 - ExampleModel\$User.doWork() line: 45
 - ExampleModel.doWork_aroundBody0(ExampleModel, ExampleModel\$User, JoinPoint) line: 62
 - ExampleModel.test() line: 62
 - Example.main(String[]) line: 69
 - Thread [Thread-0] (Running)

C:\Program Files\Java\jre7\bin\javaw.exe (2014.07.10. 18:14:41)

Variables Breakpoints

Name	Value
this	ExampleModel\$User (id=25)
myMachine	ExampleModel\$Machine (id=28)
currentState	ExampleModel\$Machine\$Off (id=30)
identifier	"inst_1447237459" (id=32)

txtuml.examples.example1.ExampleModel\$Machine\$Off@3b5c8049

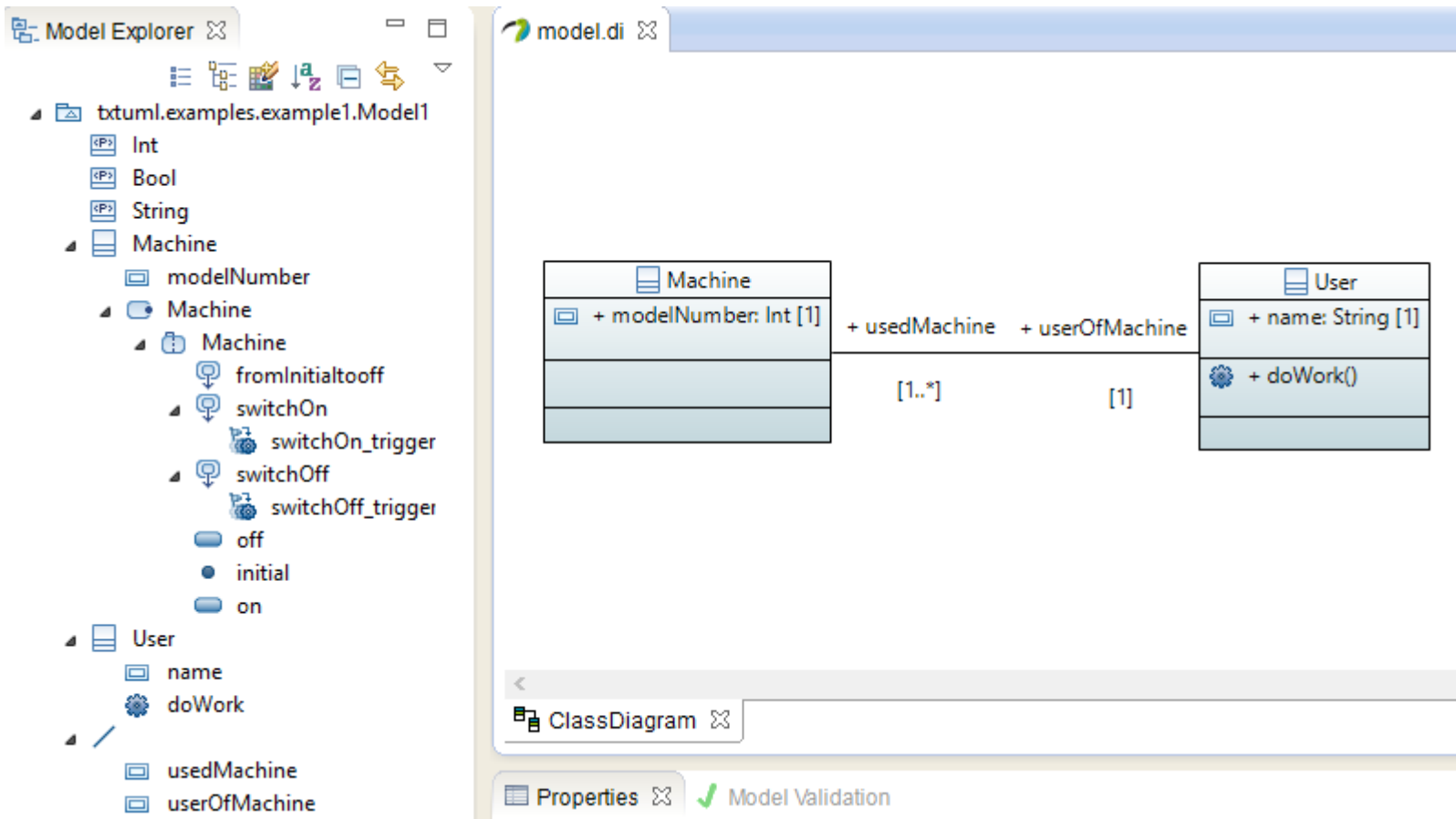
Example.java

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

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



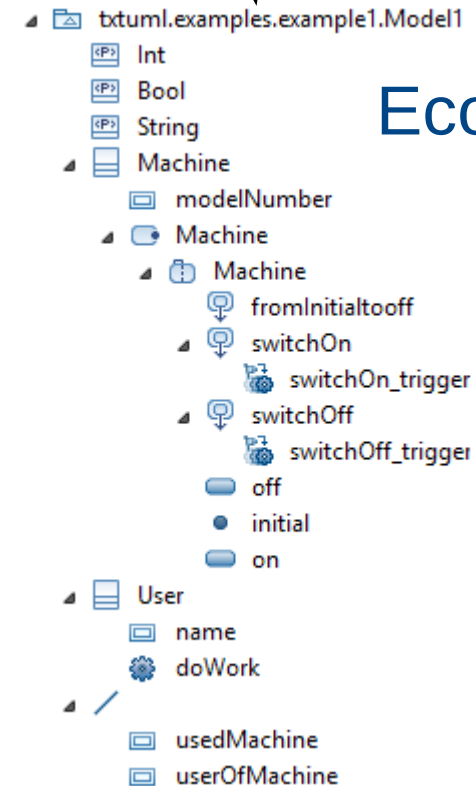
txtUML models can be exported to [Ecore UML2](#), and visualized in the [Papyrus](#) editor.

Implementation uses [Java reflection](#) and [AspectJ](#).

Compilation

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

txtUML
source code



Ecore UML2
model

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

generated
C++ code

{txtUML}

textual, executable, translatable

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!

