Run-Time Assertion Checking of Data- and Protocol-Oriented Properties of Java Programs

Frank de Boer
Joint work with Stijn de Gouw, Peter Wong and Einar Broch Johnsen [1].

[1] This work has been carried out in HATS (Highly Adaptable and Trustworthy Software using Formal Models) project, an Integrated Project supported by the 7th Framework Programme of the EC within the FET (Future and Emerging Technologies) scheme

Outline

1. The problem.
2. Our solution (Basic Idea).
3. Implementation (Tool).
4. Conclusion (All is Well That Ends Well).

Behavioral Abstraction (Encapsulation)

State of the Art:

- Getters and setters:

  Get_X

- Model variables (JML):

  public model instance JMLObjectBag elementsInQueue

Main Problem:

State of the Art is State-based

It's Histories, Stupid!

What ? Histories? Yes:

Sequences of Messages!

A short history of histories:

- *Proofs of networks of processes* by Misra and Chandy, in IEEE Transactions on Sofware Engineering, 1981.
- *Formal justification of a proof system for CSP* by K.R. Apt in J.ACM, Vol 30, 1983.
- *A theory of communicating sequential processes*, by Brookes, Hoare and Roscoe, in J. ACM, Vol. 31, 1984.
- *Compositionality and concurrent networks: soundness and completeness of a proof system* by Zwiers, de Roever and van Emde Boas, in LNCS, Vol. 194, 1985.
- *Fully abstract trace semantics for a core Java language* by Jeffrey and Rathke, in LNCS, Vol. 344, 2005.
- *An assertion-based proof system for multithreaded Java* by Abraham, de Boer, de Roever and Steffen, in TCS, Vol. 331, 2005.

Example: Compositional Proof Theory for CSP

From non-compositional:

Communication Assumptions $\{p\}c?x\{q\}$ and $\{p\}c!e\{q\}$

Cooperation Test

$$\left.\begin{array}{lll} \{p\} & c!e & \{q\} \\ \{p'\} & c?x & \{q'\} \end{array}\right\} \Rightarrow \{p \wedge p'\}x := e\{q \wedge q'\}$$

to compositional:

Communication Axioms

$$\{\forall x.p[h \cdot (c,x)/h]\}c?x\{p\} \text{ and } \{p[h \cdot (c,e)/h]\}c!e\{p\}$$

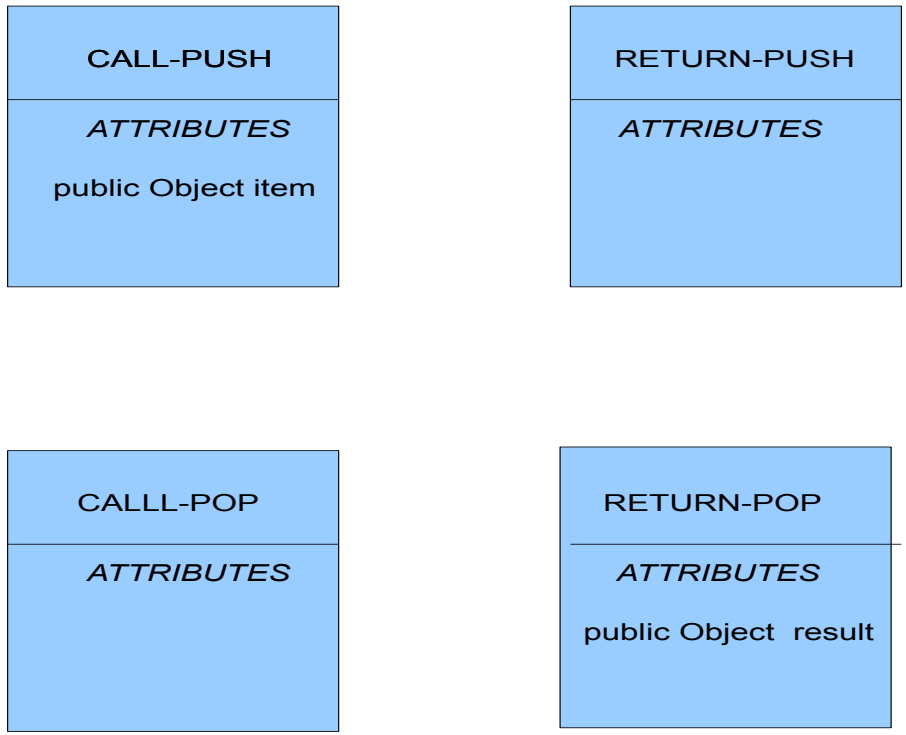
Example:

$$\frac{\{[h]_c = \epsilon\}c?x\{[h]_c = (c,x)\}\ \{[h]_c = \epsilon\}c!0\{[h]_c = (c,0)\}}{\{[h]_c = \epsilon\}c?x \parallel c!0\{[h]_c = (c,x) \wedge [h]_c = (c,0)\}}$$

Specifying Interfaces in Java using Histories: A Running Example

```
interface Stack {
  void push(Object item);
  Object pop();
}
```

The Modelling Framework: Messages



The Modelling Framework: Communication Views

```
view StackHistory {
  call-push push;
  call-pop pop;
}
```

The Modelling Framework: Attribute Grammars [2]

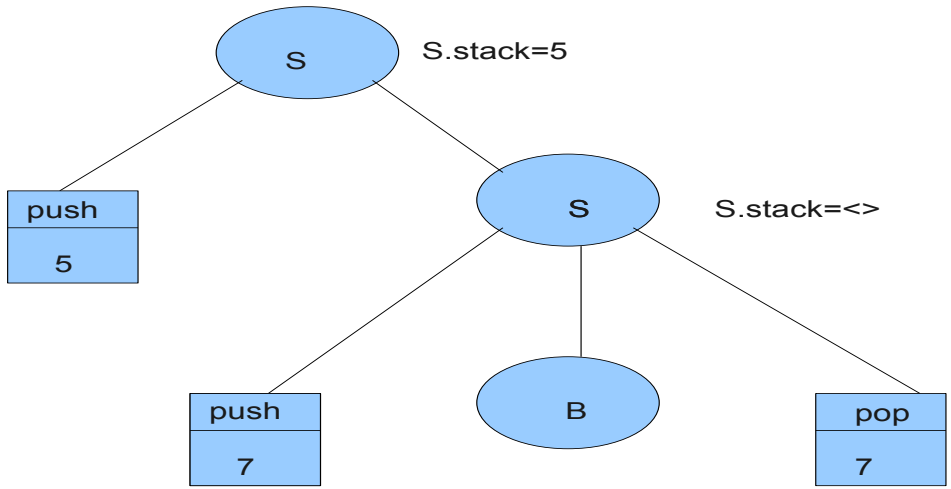
```
class EList extends List {
public EList append(Object element)
public EList append(EList list) }

Elist stack
```

S	::=	push S_1	stack = S_1.stack.append(push.item)
	|	S_1 S_2	stack = S_1.stack.append(S_2.Stack)
	|	B	stack = stack.clear()
B	::=	push B pop	
	|	ϵ	

[2] D. E. Knuth: The genesis of attribute grammars. Proceedings of the international conference on Attribute grammars and their applications (1990), LNCS, vol. 461, 112. Some informal, historical information.

Example

Parse tree of push(5) push(7) pop

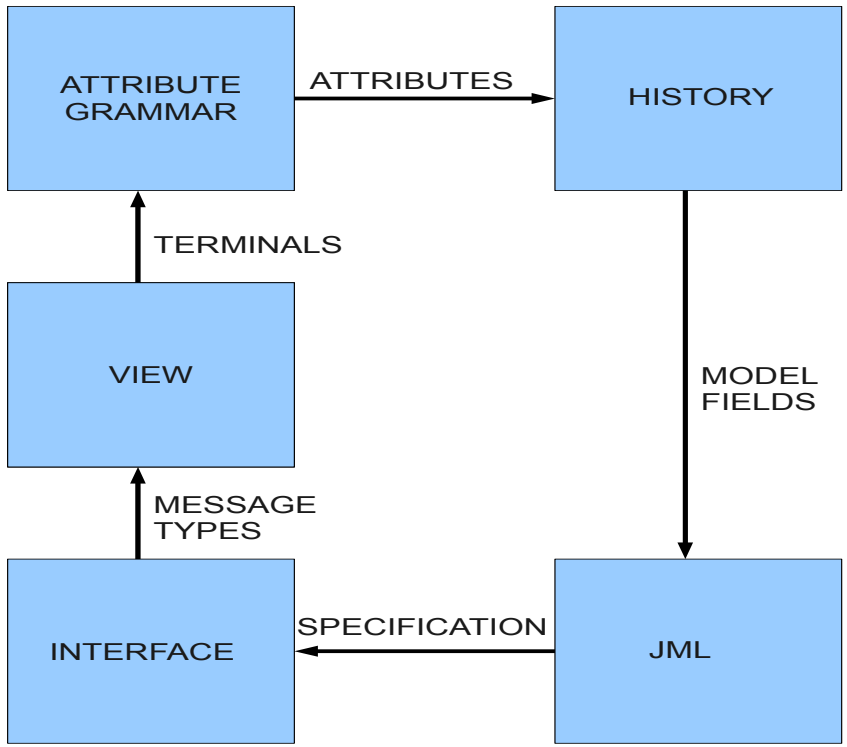


The Modelling Framework: Interface Specifications

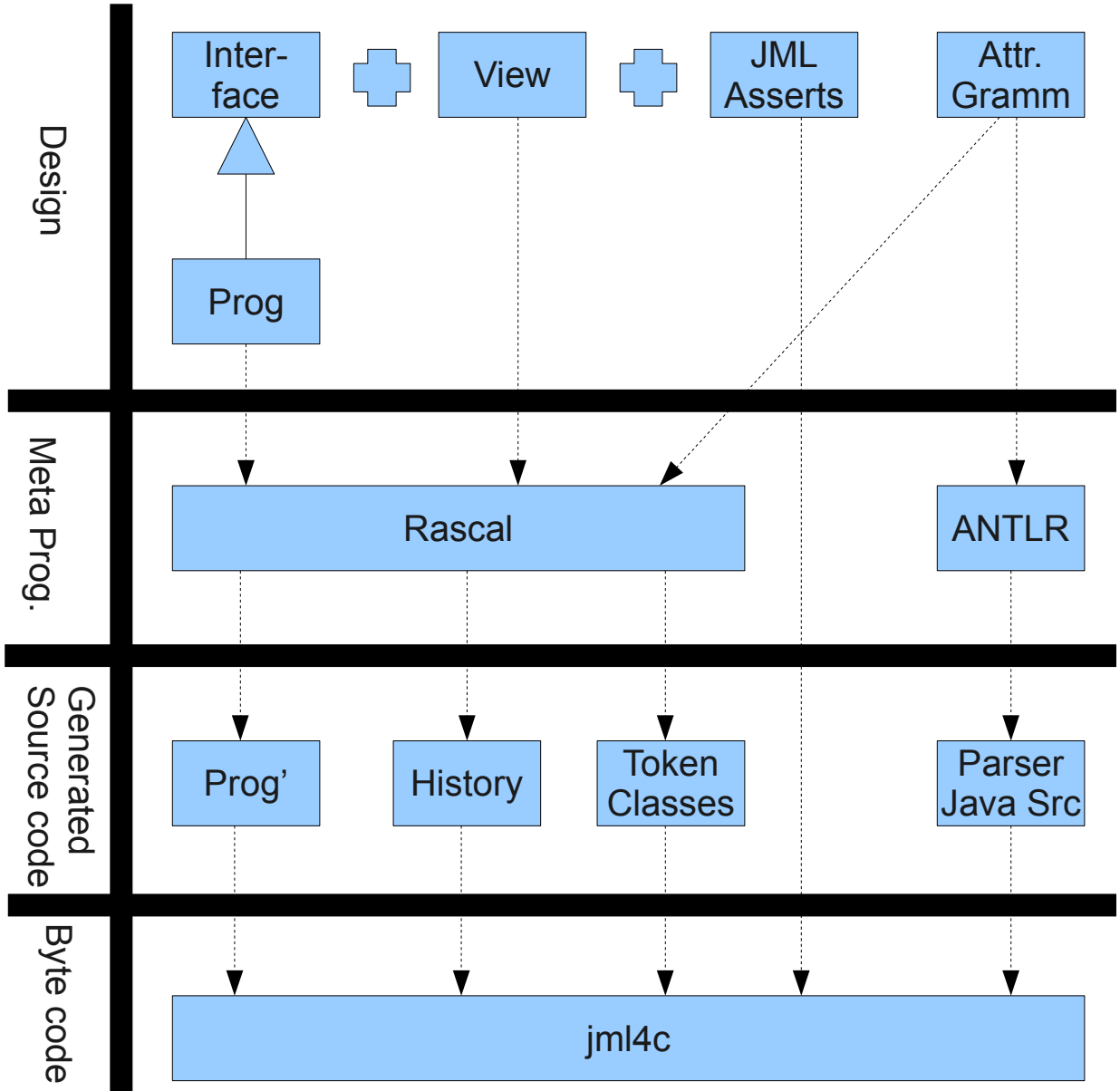
```
class StackHistory {
public EList stack() }
```

```
interface Stack {
//@ public model instance history StackHistory;
//@ ensures history.stack() ==
        \old(history.stack()).append(item);
void push(Object item);
/@ requires history.stack().size() ! = 0;
//@ ensures history.stack() == \old(history.stack()).tail();
//@ ensures \result == \old(history.stack()).head();
Object pop()
```

The Modelling Framework: Summary



Generative Framework



Run-Time Assertion Checking: Method Return

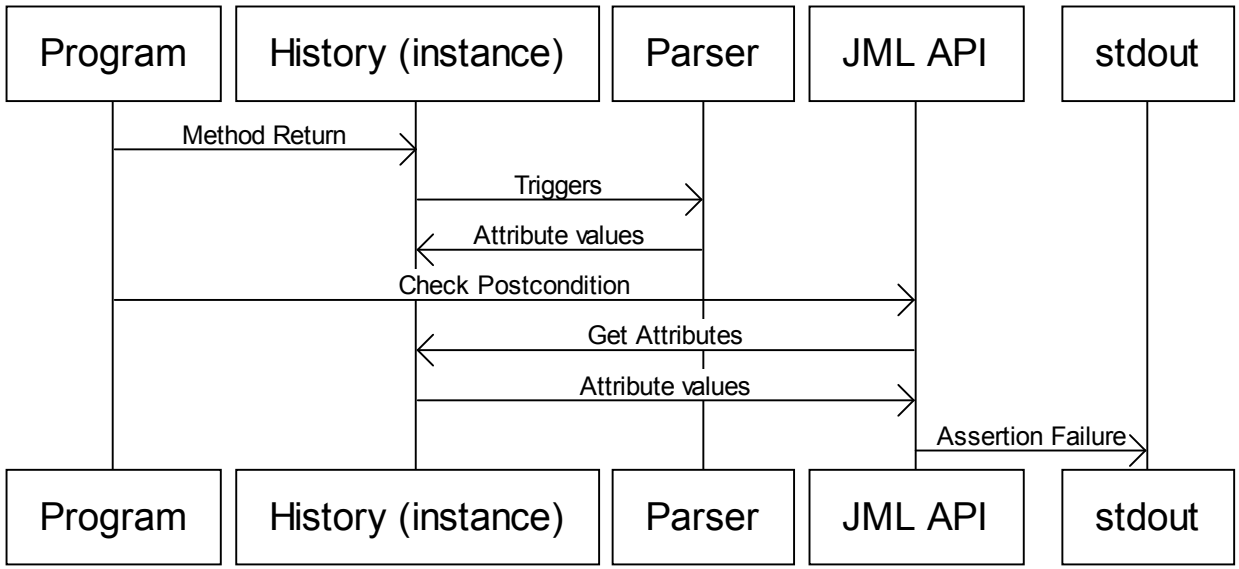


Figure: Sequence Diagram of postcondition assertion failure

Tools

http://sites.google.com/site/cdegouw/

Some Papers

- Frank S. de Boer, Stijn de Gouw.
  Run-Time Verification of Black-Box Components Using Behavioral Specifications: An Experience Report on Tool Development.
  FACS 2012: pp. 128-133.
- Stijn de Gouw, Frank S. de Boer, Peter Y.H. Wong, and Einar Broch Johnsen.
  Run-Time Checking of Data- and Protocol-Oriented Properties of Java Programs: An Industrial Case Study.
  SAC 2013.
- Frank S. de Boer, Stijn de Gouw, Einar Broch Johnsen, Andreas Kohn, and Peter Y. H. Wong.
  Run-Time Assertion Checking of Data- and Protocol-Oriented Properties of Java Programs: An Industrial Case Study.
  Special Issue of the Journal on Transactions on Aspect-Oriented Software Development. To appear.
- Stijn de Gouw, Frank S. De Boer and Peter Y.H. Wong.
  Run-Time Verification of Coboxes. Submitted to Forte13.

Summarizing

Attribute grammars provide a *systematic* and *generic* approach for specifying histories which

- allows a *declarative* expression of complex properties of histories and
- provides a *separation of concerns* between
  - *protocol oriented properties* (grammar)
  - *data-oriented properties* (attributes)
- allows a seamless integration into JML

Main Thing To Do: Error Reporting

But Also Don't Underestimate

General Tool Development/Maintenance

Thank You For Your Attention!