A .NET-based Test-Data Generator for Combinatorial Grammar- and Schema-based Testing

Vadim Zaytsev
with: Ralf Lämmel (VU), Wolfram Schulte (MSR)

14 April 2004

*Grammar*ware

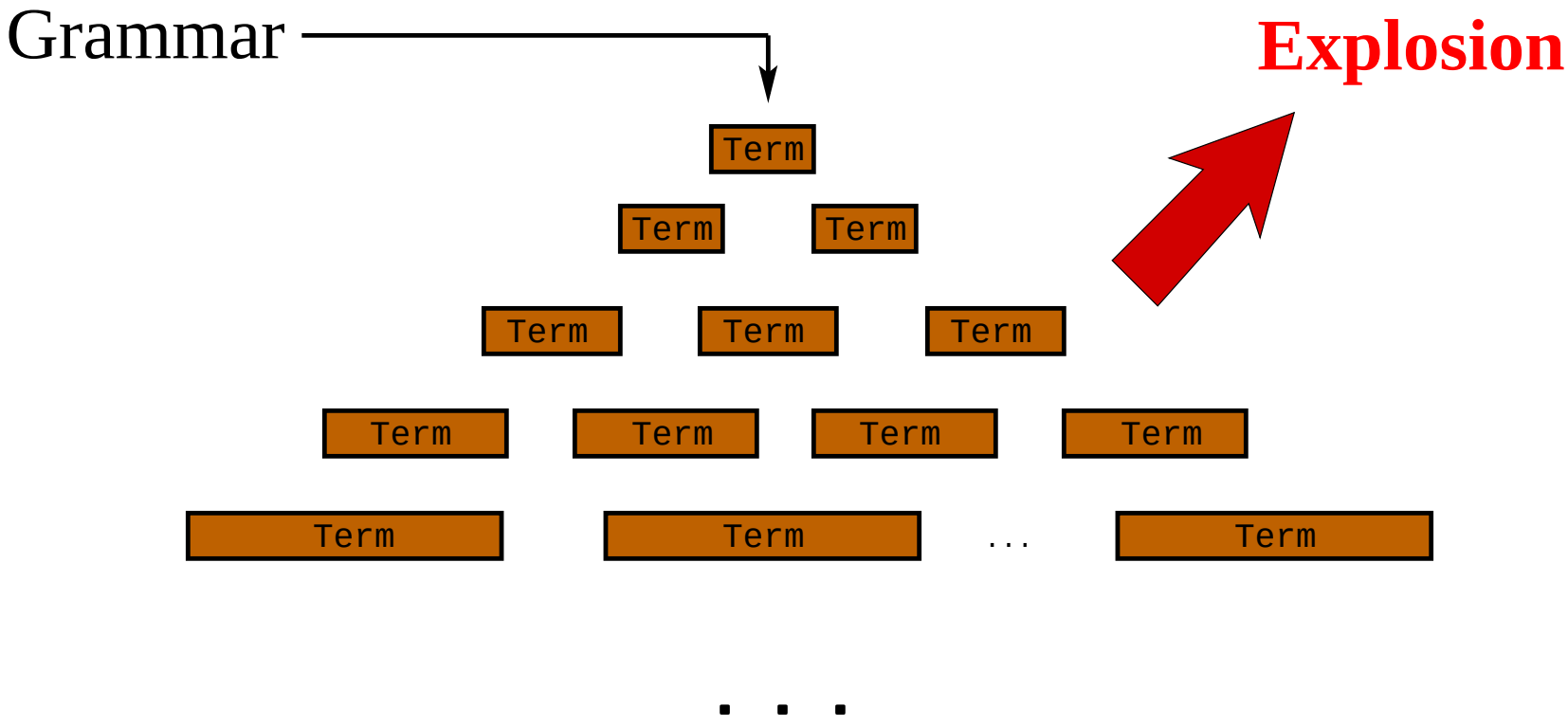
- grammars
- grammar-dependent software
- In this project:
  - XML Validators
  - W3C XML Schemata as grammars

`http://www.cs.vu.nl/grammarware/`

Scenarios for grammar-based testing

- testing virtual processors
  - virtual machines
  - just-in-time compilers
- testing front-ends
  - automated software modification & analysis
- testing implementations
  - optimisation of XPath

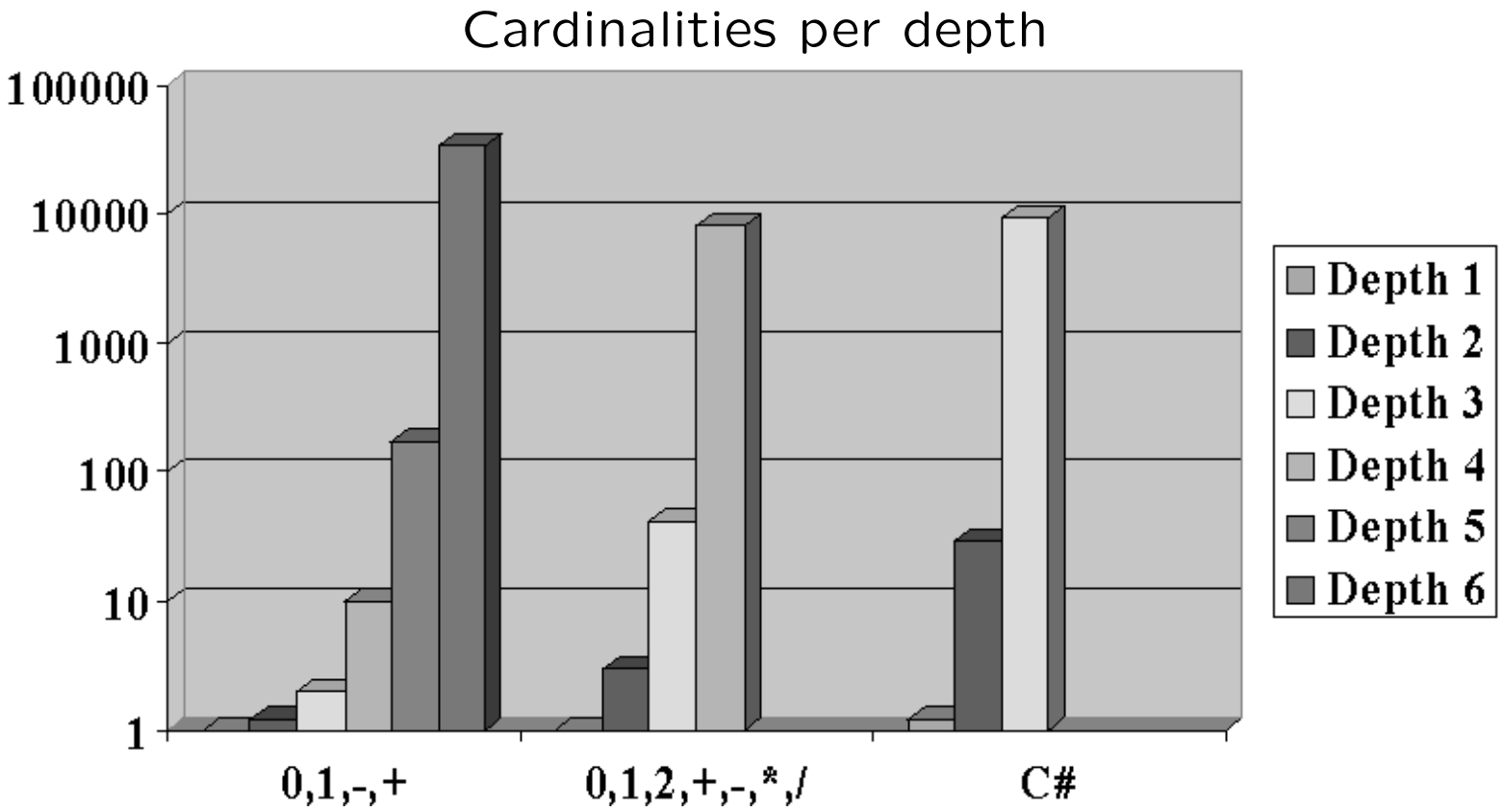
Combinatorial exploration



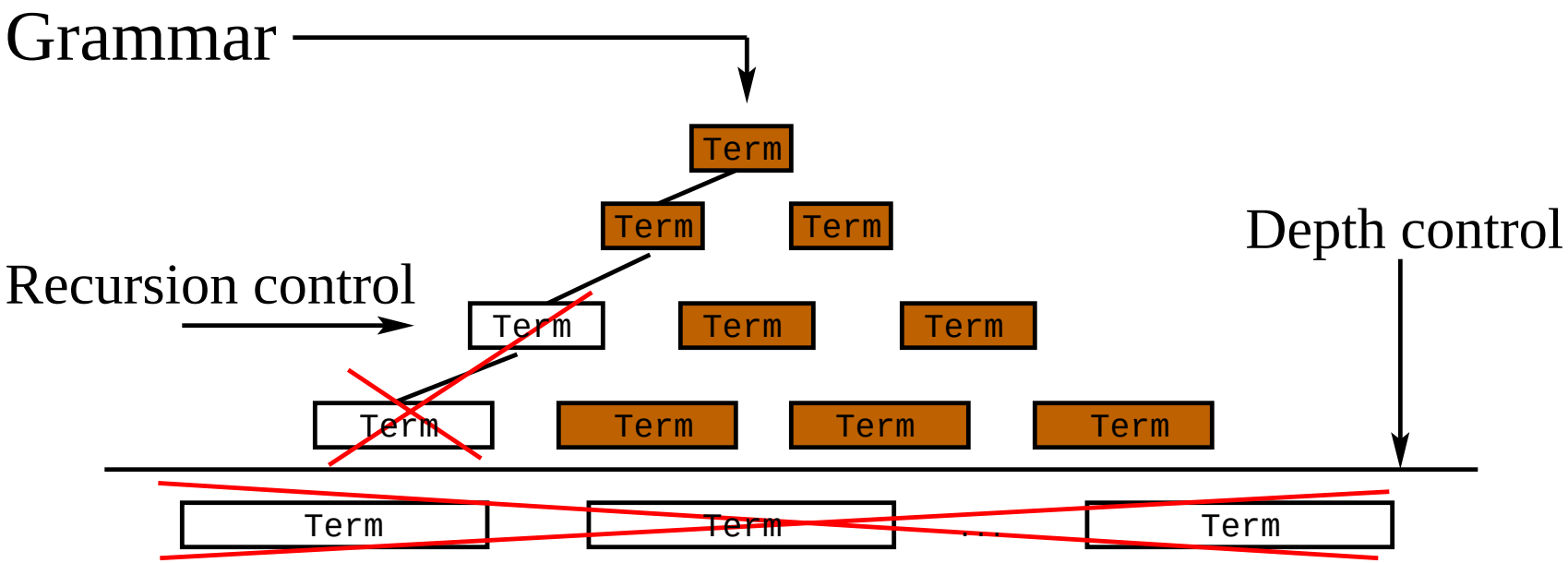
. . .

Adversary of stochastic testing

Explosion examples



Controlled explosion



. . .

\+ other mechanisms

Control mechanisms

- depth control
  - maximum "length" of terms
- recursion control
  - nested constructor application
- equivalence control
  - build equivalence classes

Control mechanisms (contd.)

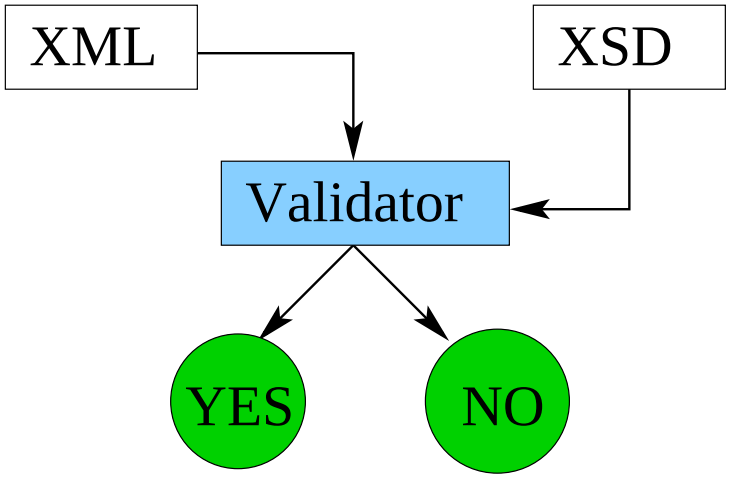
- balance control
  - limit the preceding levels
- combination control
  - limit Cartesian product
  - pair-wise testing
- context control
  - enforce context conditions

Emphases in this project

- The case study of XSD usage in testing XML Validators
- Implementing and using control mechanisms for test data generation
- Developing a tool to support combinatorial testing

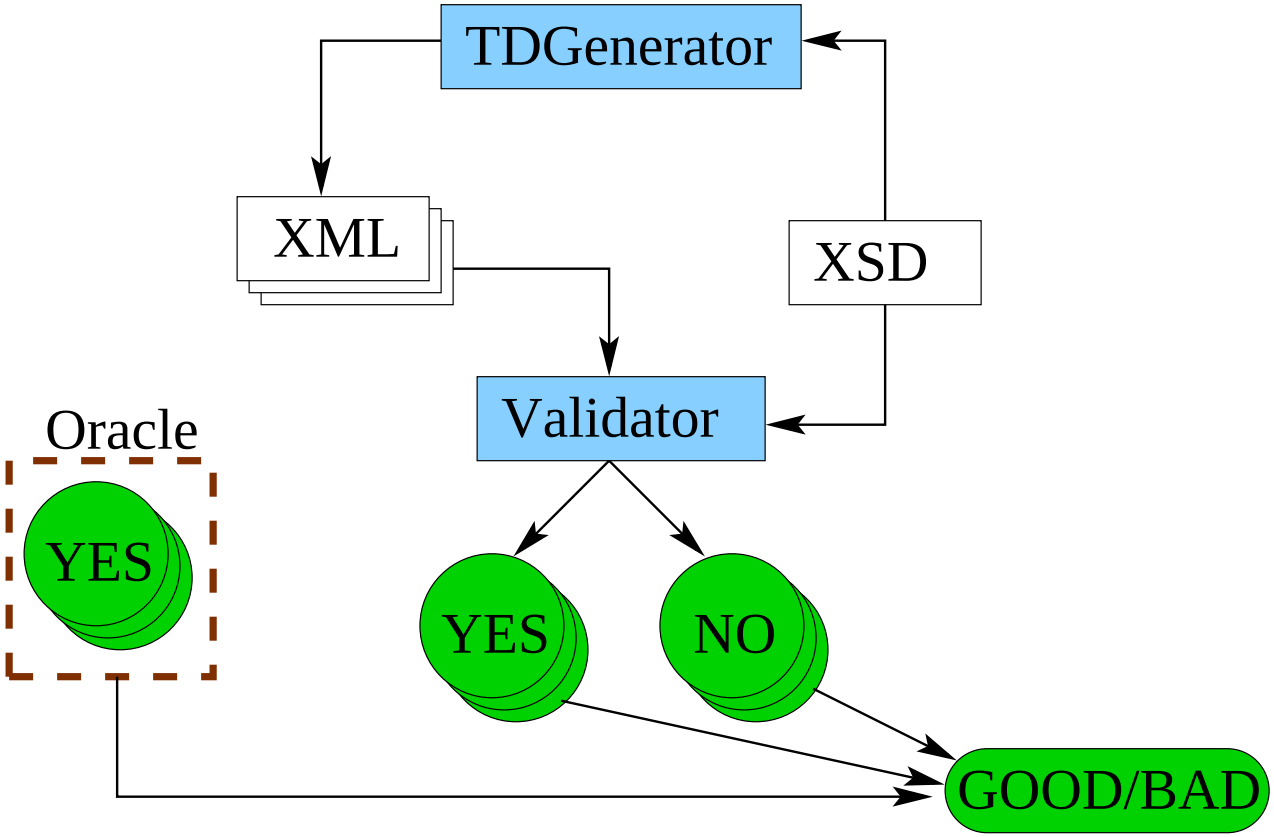
Problem

System Under Test



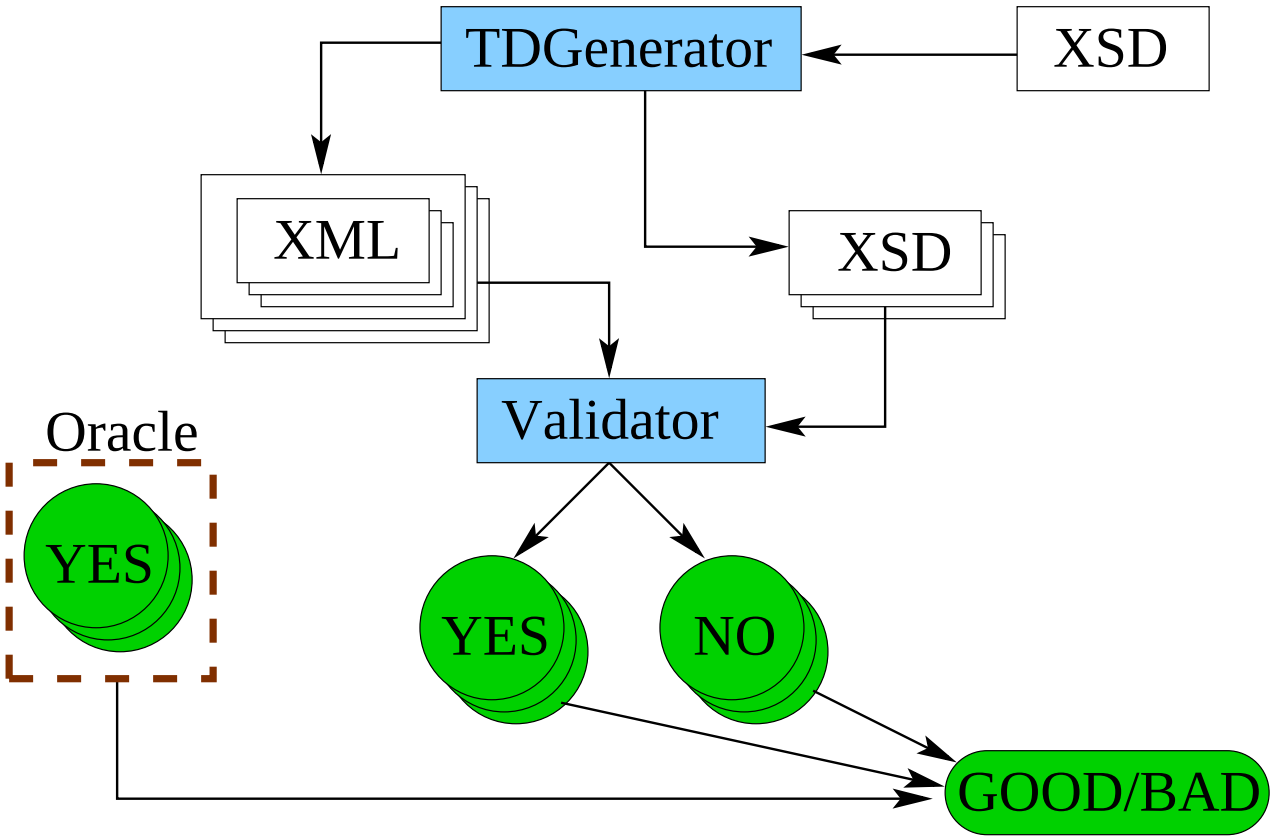
Solution

Stage A:



Solution

Stage B:



What kind of *Oracle*?

- Differential testing
  - run two or more against one another
  - if the outputs are different, something has to be wrong
  - in our case: different XML Validators
    - using Microsoft .NET API
    - Sun Multi-Schema XML Validator (JAXB)
    - Ant Validation Task (JBind)

The Tool we have

grammar terms

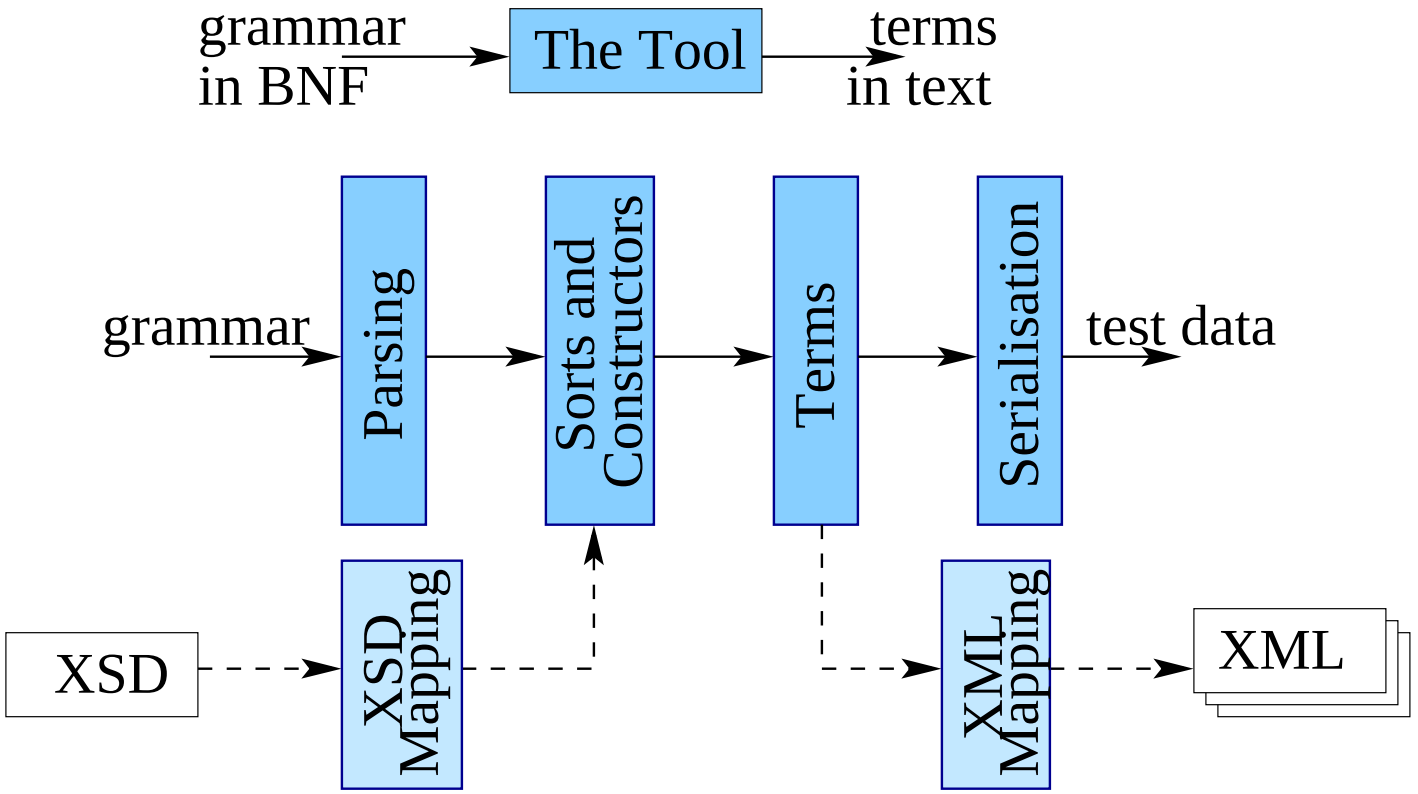
```
Tree
 = Nil
 | Node(Tree,Tree)
 ;
```

Constructors of all sorts

```
Nil
Node-1(Nil,Nil)
...
```

Terms as objects
+Serialisation

Solution proposition



Problems underway

- XSD is not meant to be implemented
  - (as a whole)
- `YACC`ification
- how to deal with XML attributes
- implementing control mechanisms
- ...

Conclusion

The buzzwords are:

- Test data generation
- Combinatorial testing
- Controlled explosion
- Differential testing
- The .NET Framework