Live-Verification While Programming:

Dafny

Presented by: Philippe Heiler

Image Source: https://www.microsoft.com/

Introduction

- Dafny was developed by K. Rustan M. Leino at Microsoft in 2008
- Part of Microsoft Research RiSE *(Research in Software Engineering)*
- Live Verification Language

- Cross Language Compiler to
  - JavaScript, Go, .NET Languages

- Supports
  - Generic Classes
  - Dynamic Allocation
  - Specification Constructs
    - Pre- and Post-Conditions
    - Ghost State
    - Dynamic Frames
    - Proof of Termination

Microsoft

RiSE

Image Sources:
http://ase-conferences.org/ase/past/ase2007/images/RustanLeino.png
https://www.microsoft.com/

Pre- and Post- Conditions

- Pre- and Post- Conditions (*requires* and *ensures*) are checked before compilation
  - Works similar to Typing constraints in other programming languages like C# or Java
  - Dafny analyzes possible code paths, to validate the solvability of the given specifications

- This assures that certain specifications are complied before and after method execution
  - Preconditions need to be ensured before the Methods Body
  - Postconditions need to be ensured after the Methods Body

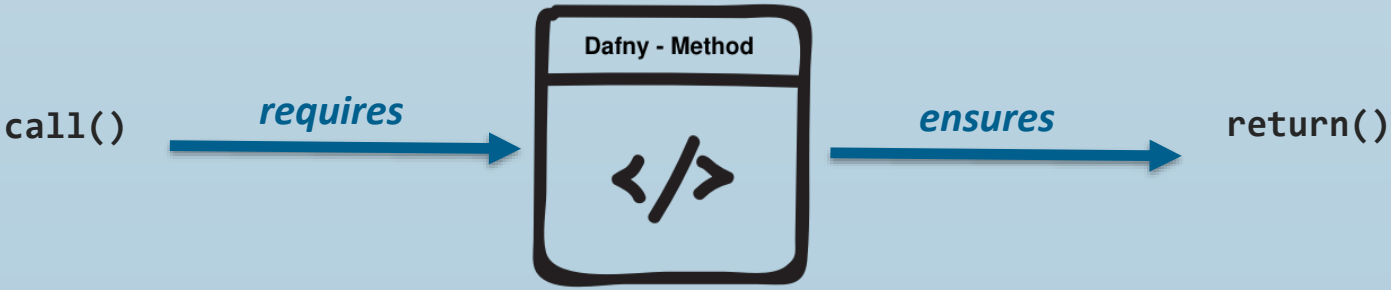


Image Source: https://www.iconfinder.com/data/icons/online-marketing-hand-drawn-vol-2/52/development__code__script__web__tag__root__browser-512.png

Ghost State

- Everything that is specified as ghost is only considered by the Verifier, and will be completely ignored by the Compiler
- Variables can be initialized as Ghost Variables
- Ghost Methods are called Functions
- Ghost Variables and Functions are prohibited to influence normal Variables and Methods
  - This is also checked by the Verifier

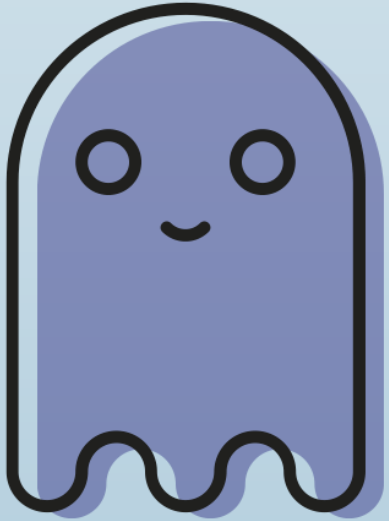


Image Source: https://cdn3.iconfinder.com/data/icons/sweet-halloween/512/Sweet_Halloween_Ghost-256.png

Dynamic Frames

- Dynamic Frames assure that only the selected objects can be accessed by the current Method
- The ‘modifies’ statement assures, that only the selected objects can be altered by the current Method
- The ‘reads’ statement assures, that only the selected objects can be read by the current Method
- The selection is recursive complete, which means every object that is a parameter of the selected object (*and so on*) will also be accessible.

Image Source: https://www.flickr.com/photos/d_pham/8072408810/

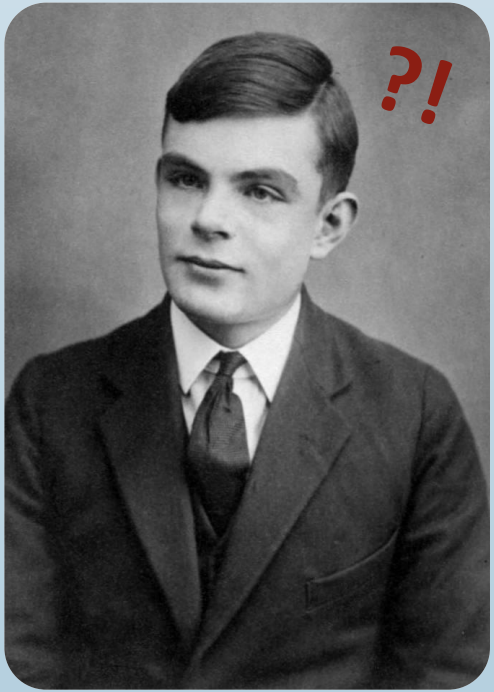
Proof of Termination

- Dafny can check if a Method with Loops and / or Recursion will terminate eventually
- The termination proof will only take seconds, even if the programs termination itself could need years
  - E.g. recursive Ackermann or Fibonacci

Image Source: https://pixabay.com/photos/pocket-watch-time-of-sand-time-3156771/

Proof of Termination

- Dafny can check if a Method with Loops and / or Recursion will terminate eventually
- The termination proof will only take seconds, even if the programs termination itself could need years
  - E.g. recursive Ackermann or Fibonacci



WAIT A MINUTE !?

(Pun not intended)

Isn't that the Halting-Problem ?

Why can Dafny solve NP Complete ??

Image Source: https://de.wikipedia.org/wiki/Alan_Turing#/media/Datei:Alan_Turing_Aged_16.jpg

Proof of Termination

- Dafny can check if a Method with Loops and / or Recursion will terminate eventually
- The termination proof will only take seconds, even if the programs termination itself could need years
  - E.g. recursive Ackermann or Fibonacci
- Dafny will analyze the termination with the help of the 'decreases' variable and tries to figure out if the selected variable decreases towards it's halting anchor
- This works for Loops and Recursion, but there could still be algorithms that can not be proven to terminate
  - This will result in a verification timeout and a warning

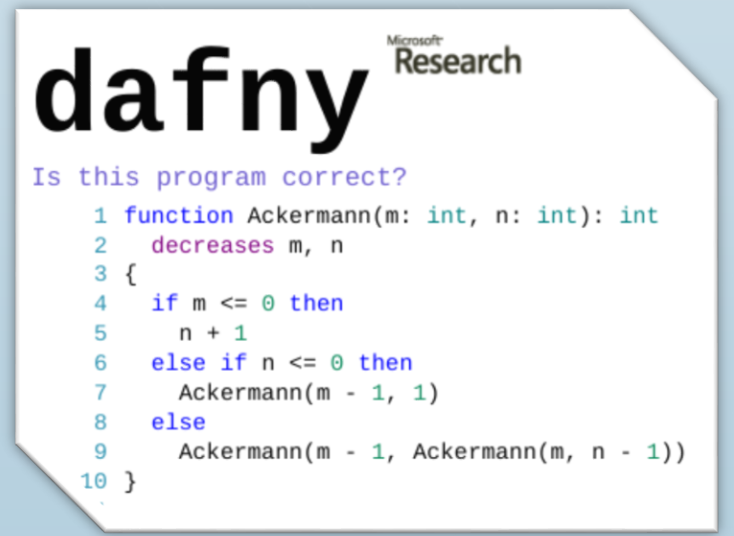


Image Source: https://rise4fun.com/Dafny/

How ? Magic ?

- How can Dafny verify all those things ?
- Dafny uses an Intermediate Verification Language called Boogie *(Microsoft)*
- Boogie is able to translate specifications defined by Dafny into Mathematical Formulas, the so called 'Theories'
- Those Mathematical Formulas will then be solved by the SMT-Solver Z3 *(Microsoft)*
- This will result in a feedback if (e.g.) a Pre- or Post- Condition can be fulfilled or not

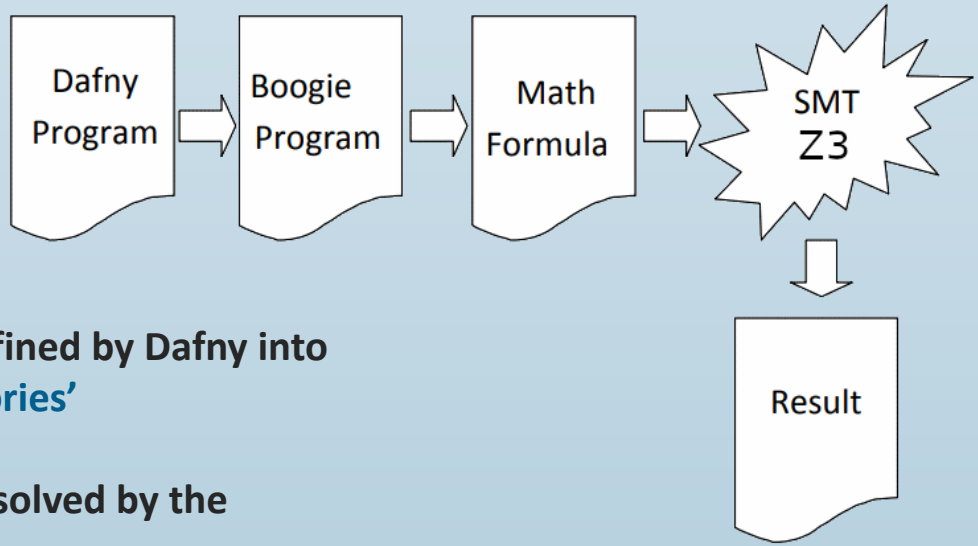
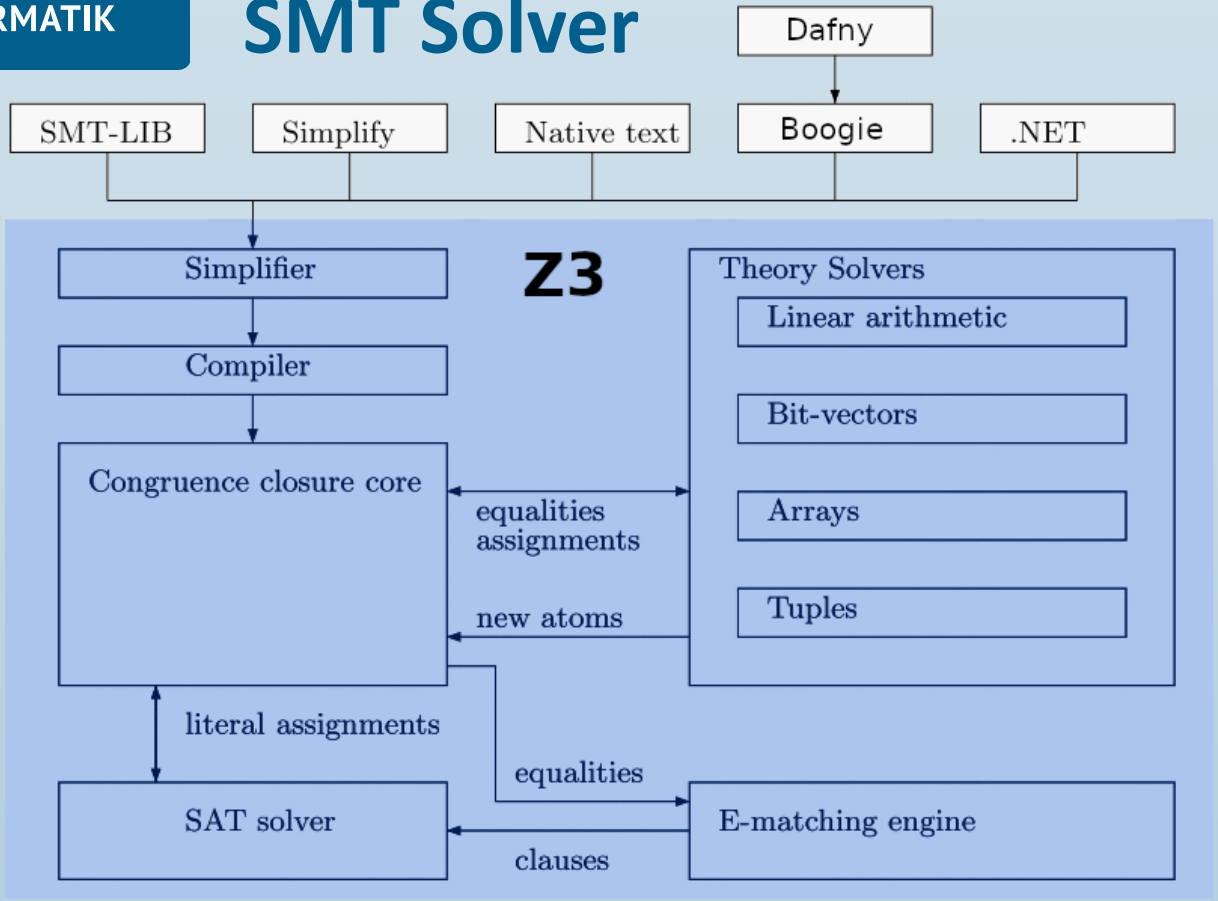


Image Source: https://ethz.ch/content/dam/ethz/special-interest/infk/chair-program-method/pm/documents/Education/Theses/Alexandru_Dima_RS_report.pdf

SMT Solver



Satisfiability
Modulo
Theories

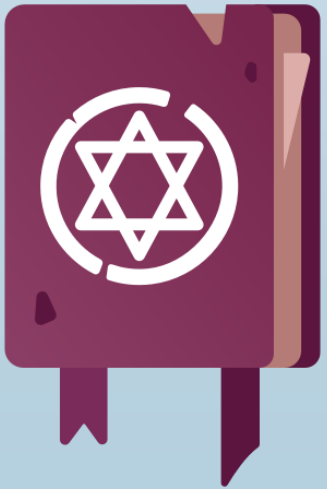


Image Sources: http://algos.inesc-id.pt/projectos/projects/projects/nanotime/ref12.pdf
https://cdn3.iconfinder.com/data/icons/fantasy-and-role-play-game-adventure-quest/512/Spell_Book-256.png

Specification - Classes

- Abstract Classes are called Traits
- Classes may extend other classes or Traits
- Consist of Constructor, Variables and Methods
- Variables can be initialized as Ghost Variables

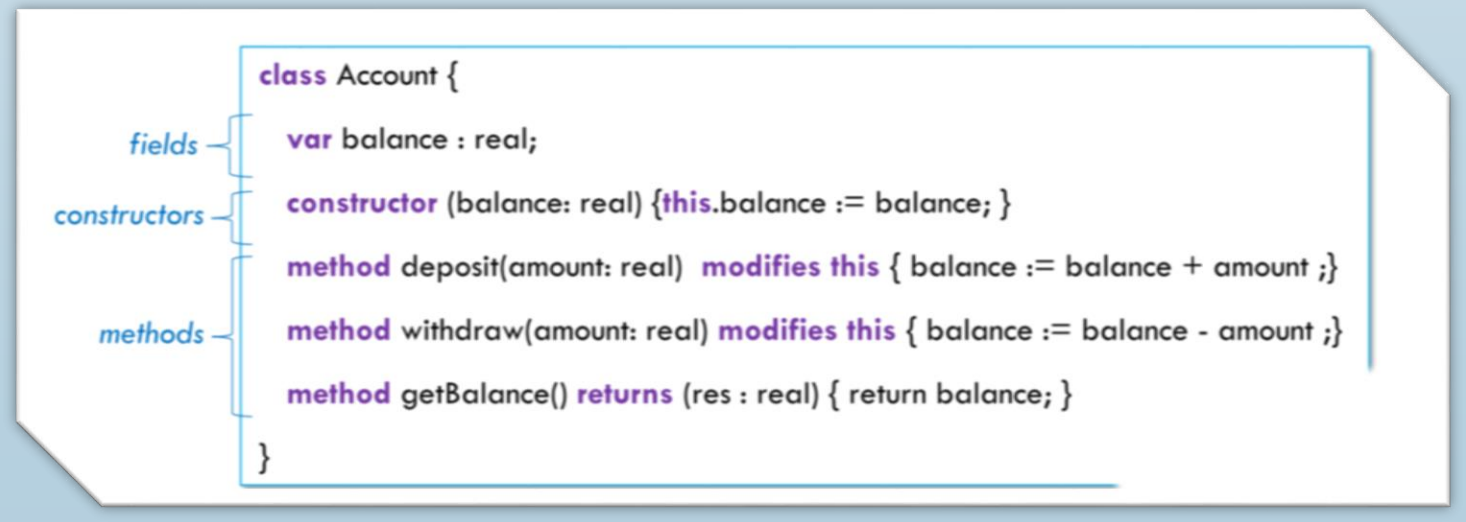


Image Source: https://www.dcc.fc.up.pt/~nam/web/resources/vfs20/DafnyQuickReference.pdf

Specification - Methods

- Has multiple Input and Output Parameters
- Uses Pre- and Post- Conditions
- Uses Dynamic Frames to manage Memory Access
- Proofs Termination of Loops and Recursive Methods
- Ghost Methods are called Functions

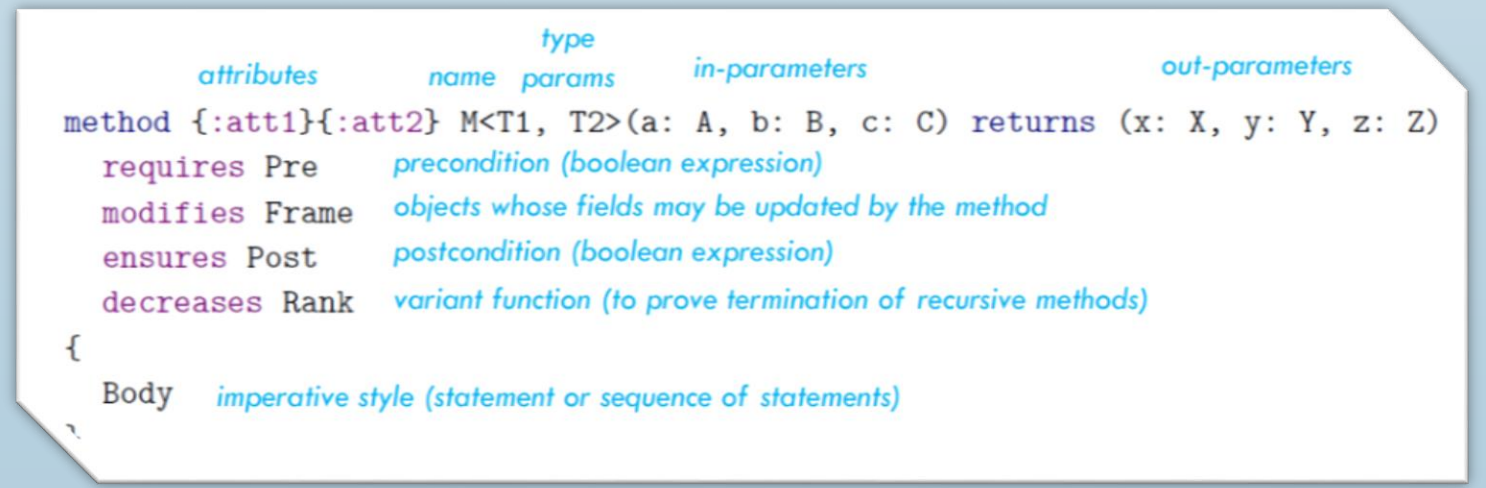


Image Source: https://www.dcc.fc.up.pt/~nam/web/resources/vfs20/DafnyQuickReference.pdf

Example: Queue Datastructure


```
1   method Dequeue()
2     requires Valid();
3     requires 0 < |contents|;
4     modifies footprint;
5     ensures Valid() ∧ fresh(footprint − old(footprint));
6     ensures contents = old(contents)[1..];
7   {
8     var n := head.next;
9     head := n;
10    contents := n.tailContents;
    }
```

Not Empty

Dynamic Frame

Swinging Pivot Restriction

Old Queue but *WITHOUT* the first entry

Function Body

Image Source: https://www.flickr.com/photos/hktang/4243300265

Literature

Introduction and Theory:

- ‘*Live-Verification while Programming: Dafny*’ (Philippe Heiler, 2020) ☺
- ’*Dafny: An automatic program verifier for functional correctness*’ (K. Rustan M. Leino, 2010)
- ’*Specification and Verification of Object-Oriented Software*’ (K. Rustan M. Leino, 2010)

Programming Documentation:

- ‘*Dafny Quick Reference*’ (J. Pascoal Faria, 2020)
- ’*Dafny Reference Manual*’ (Richard L. Ford et al. 2017)

Try out Dafny:

- ‘*Dafny GitHub Page*’
  (https://github.com/dafny-lang/dafny/releases)
- ‘*Dafny Online Verifier*’
  (https://rise4fun.com/Dafny/)

Image Source: https://pixabay.com/de/photos/literatur-buch-weisheit-bibliothek-3324023/