

The Science of Software

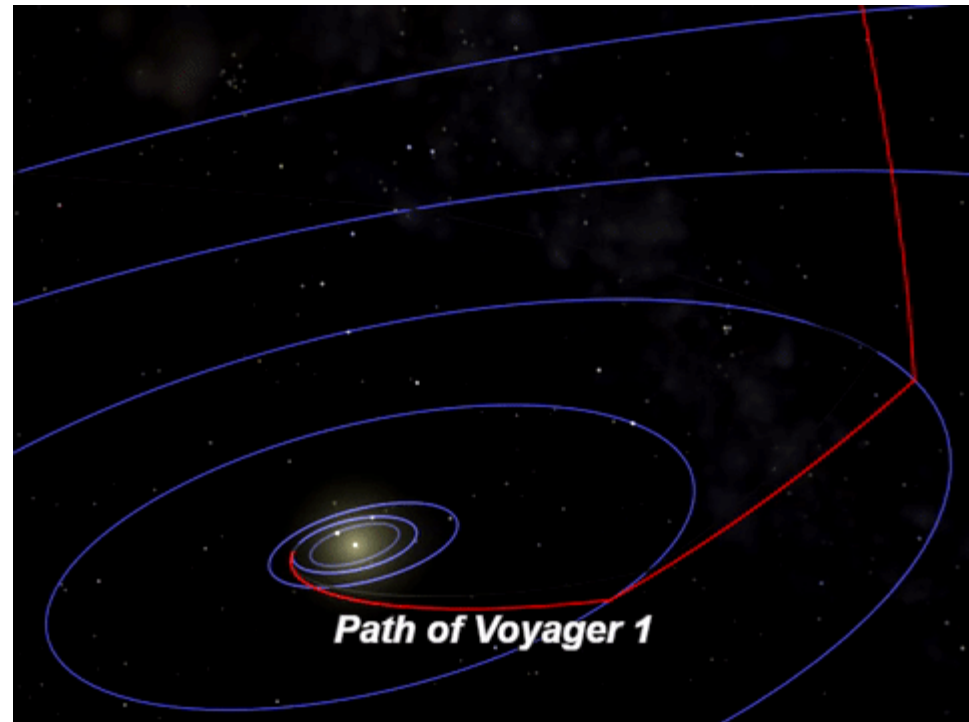
(a.k.a. Formal Methods)

Stavros Tripakis

Aalto University

What is science?

science => **predictions**



What is the science of software?

- What predictions can we make about the programs we write?
- Can I predict that my program will:
 - Terminate?
 - Never throw an exception?
 - Produce the right result?
 - Always?
 - Sometimes?
 - ...



```
78 // Trim the string and replace slashes with forward slashes
79 $hex_str = trim(preg_replace('/\\\\\\\\/', '/', $image_src), '/');
80
81 $SESSION['_CAPTCHA']['config'] = serialize($captcha_config);
82
83 return array(
84     'code' => $captcha_config['code'],
85     'image_src' => $image_src
86 );
87
88
89
90
91 if (function_exists('hex2rgb')) {
92     function hex2rgb($hex_str) {
93         $hex_str = preg_replace('/[^\0-9A-Fa-f]/', '', $hex_str); // Gets a proper
94         if (strlen($hex_str) == 6) {
95             $color_val = hexdec($hex_str);
96             $rgb_array['r'] = 0xFF & ($color_val >> 0x10);
97             $rgb_array['g'] = 0xFF & ($color_val >> 0x8);
98             $rgb_array['b'] = 0xFF & ($color_val >> 0x0);
99         } elseif (strlen($hex_str) == 3) {
100             $rgb_array['r'] = hexdec(str_repeat(substr($hex_str, 0, 1), 2));
101             $rgb_array['g'] = hexdec(str_repeat(substr($hex_str, 1, 1), 2));
102             $rgb_array['b'] = hexdec(str_repeat(substr($hex_str, 2, 1), 2));
103         } else {
104             return false;
105         }
106     }
107 }
108
109 // Draw the image
110 if (isset($_GET['code'])) {
111     $code = $_GET['code'];
112     $image_src = $SESSION['_CAPTCHA']['image_src'];
```

(

software science \neq comput^{ing} science

software science \subsetneq comput^{er} science

)

What is the mathematics of the science of software?

- Language
- Truth
- ...



logic

Formal (mechanized) logic and proof

- specification and verification

Isabelle2016-1 - AlgorithmFeedback.thy

File Edit Search Markers Folding View Utilities Macros Plugins Help

AlgorithmFeedback.thy (C:\cygwin64\home\stavros\stavros_svn\syrein\Isabelle\TranslateHBD\)

```
Hoare (invariant As_init w  $\sqcap$  ( $\lambda$ As. Suc 0 < length As))
  [ :As $\rightsquigarrow$ As'.  $\exists$ Bs. Suc 0 < length Bs  $\wedge$  ( $\exists$ Cs. perm As (Bs @ Cs)  $\wedge$  As' = FB (Parallel_list Bs) # Cs) : ] (Sup_le
apply (rule hoare_demonic, safe)
apply (simp add: Sup_less_def)
apply (rule_tac x = "invariant As_init (Suc (length Cs))" in exI, safe)
apply (rule_tac x = "(Suc (length Cs))" in exI, simp_all)
apply (simp add: invariant_def, safe)
by (drule perm_length, simp)

lemma [simp]: "io_distinct As_init  $\implies$  Suc 0  $\leq$  length As_init  $\implies$ 
  Hoare (invariant As_init w  $\sqcap$  ( $\lambda$ As. Suc 0 < length As))
    [ :As $\rightsquigarrow$ As'.  $\exists$ A B Bs. perm As (A # B # Bs)  $\wedge$  As' = FB (FB A ;; FB B) # Bs : ] (Sup_less (invariant As_init) w
apply (rule hoare_demonic, safe)
apply (simp add: Sup_less_def)
apply (rule_tac x = "invariant As_init (Suc (length Bs))" in exI, safe)
apply (rule_tac x = "(Suc (length Bs))" in exI, simp_all)
apply (simp add: invariant_def, safe)
apply (drule perm_length, simp)
done

theorem CorrectnessTranslateHBD: "io_distinct As_init  $\implies$  length As_init  $\geq$  1  $\implies$ 
  Hoare (io_distinct  $\sqcap$  ( $\lambda$ As. As = As_init)) TranslateHBD ( $\lambda$ S. in_out_equiv S (FB (Parallel_list As_init)))"
apply (simp add: TranslateHBD_def)
apply (simp add: hoare sequential)
```

Examples

- src/HOL/ex/Seq.thy
- src/HOL/ex/ML.thy
- src/HOL/Unix/Unix.thy
- src/HOL/Isar_Examples/Drinker.thy
- src/Tools/SML/Examples.thy

Release notes

- ANNOUNCE
- README
- NEWS
- COPYRIGHT
- CONTRIBUTORS
- contrib/README
- src/Tools/jEdit/README

Tutorials

- prog-prove: Programming and Pro
- locales: Tutorial on Locales
- classes: Tutorial on Type Classes
- datatypes: Tutorial on (Co)datatype
- functions: Tutorial on Function Def
- corec: Tutorial on Nonprimitively Co
- codegen: Tutorial on Code Generat
- nitpick: User's Guide to Nitpick
- sledgehammer: User's Guide to Sledge
- eisbach: The Eisbach User Manual
- sugar: LaTeX Sugar for Isabelle do

Reference Manuals

- main: What's in Main
- isar-ref: The Isabelle/Isar Referenc
- implementation: The Isabelle/Isar
- system: The Isabelle System Manu
- jedit: Isabelle/jEdit

Old Manuals

- Original jEdit Documentation

470,4 (21323/21323)

(isabelle,isabelle,UTF-8-Isabelle)Nmr o UG 976/1119MB 11:45 AM

From software to systems



Courtesy <http://www.fastcodesign.com>

Thanks to Christos Cassandras for recommending this video

CONTRIBUTED ARTICLES

How Amazon Web Services Uses Formal Methods

By Chris Newcombe, Tim Rath, Fan Zhang, Bogdan Munteanu, Marc Brooker, Michael Deardeuff

Communications of the ACM, Vol. 58 No. 4, Pages 66-73

10.1145/2699417

[Comments \(1\)](#)

VIEW AS:



SHARE:



Since 2011, engineers at Amazon Web Services (AWS) have used formal specification and model checking to help solve difficult design problems in critical systems. Here, we describe our motivation and experience, what has worked well in our problem domain, and what has not. When discussing personal experience we refer to the authors by their initials.

At AWS we strive to build services that are simple for customers to use. External simplicity is built on a hidden substrate of complex distributed systems. Such complex internals are

SIGN IN for Full Access

» [Forgot Password?](#)

» [Create an ACM Web Account](#)

[SIGN IN](#)

ARTICLE CONTENTS:

[Introduction](#)

[Key Insights](#)

[Precise Designs](#)