

# Lack of Software Specifications: A Sustained Sustainability and Productivity Crisis

Hridesh Rajan

With Tien Nguyen, Gary T. Leavens, Robert Dyer, and Vasant G Honavar

# Sustainability and productivity challenge

- To produce critical software infrastructure so it is:
  - of highest quality and free of defects,
  - produced ethically and within budget, and
  - maintainable, upgradeable, portable, scalable, secure.
- Pervasiveness of software infrastructures in such critical areas as power, banking and finance, air traffic control, telecommunication, transportation, national defense, and healthcare need us to address this challenge.

Software specifications\* can help achieve this sustainability and productivity challenge.

\* Software specifications: formal, often machine readable, description of software's intended behavior, e.g. {Pre} S {Post} behavioral specifications

# Sustainability and productivity challenge

- If specifications are widely available, a wide variety of techniques for addressing the sustainability and productivity crisis can be enabled.
  - Maintenance of code can become easier
  - Lower cost of code understanding & total lifecycle cost
  - Specification-guided code optimization
  - Prevent introducing new bugs during maintenance
  - Code reuse
  - Specification-guided synthesis
  - Modular analysis and verification, scalable tools

# Sustainability and productivity challenge

- If specifications are widely available, a wide variety of techniques for addressing the sustainability and productivity crisis can be enabled.

Despite these benefits  
useful, non-trivial  
specifications aren't widely  
available

# Sustainability and productivity challenge

- If specifications are widely available, a wide variety of techniques for addressing the sustainability and productivity crisis can be enabled.

Why aren't software specifications widely available?

# Sustainability and productivity challenge

- If specifications are widely available, a wide variety of techniques for addressing the sustainability and productivity crisis can be enabled.

Cost  
Education  
Tools  
Libraries

# Sustainability and productivity challenge

- If specifications are widely available, a wide variety of techniques for addressing the sustainability and productivity crisis can be enabled.

**Cost**  
**Education**  
**Tools**  
**Libraries**

# Sustainability and productivity challenge

- If specifications are widely available, a wide variety of techniques for addressing the sustainability and productivity crisis can be enabled.

## **Unspecified libraries are root cause**

- increase cost of specification
- make education harder
- make tool support difficult
- make specifying libraries harder

Big Code:

large number of readily available projects

How can we effectively  
leverage these projects?

Aim: multi-step specification inference

- Bootstrap using **consensus**
- Propagate using **similarity**
- Last mile using **decomposition**

Consensus

Similarity

Decomposition

Challenge:

separate **project specific constraints**  
(chaff)

mixed with **API-specific preconditions**  
(wheat)

Consensus

Similarity

Decomposition

Our initial work<sup>1</sup>:

use **consensus** across  
**large number of projects**  
to infer API-specific preconditions

[1] FSE'14 – H. Nguyen, Dyer, T. Nguyen, Rajan

Consensus

Similarity

Decomposition

## Key Ideas

Preconditions can be mined from **guarded conditions** at the call sites of the code using the APIs

```
void m(...) {  
    ...  
    if (pred)  
        lib.api();  
    ...  
}
```

Preconditions mined from **multiple projects** in a **large-scale code corpus** can be used to filter out chaff

Consensus

Similarity

Decomposition

Challenge:

not all APIs are widely used

Consensus

**Similarity**

Decomposition

Key Ideas

**Similar code** should have  
**similar specifications**

Cluster pairs of (*code, spec*)  
from previous phase to find  
**similar code/specs**

Consensus

**Similarity**

Decomposition

Challenge:

unique API code would still be un-specified

Consensus

Similarity

Decomposition

## Key Ideas

Revision history and slicing can help **decompose** code into fragments

Simple fragments may already be specified by prior phases – compose new specification from those pieces

Consensus

Similarity

**Decomposition**

Lack of specifications,  
a critical hurdle for high assurance SE,  
can be overcome by leveraging big code mining,  
esp. **consensus**, **similarity**, and **decomposition**.