An Analysis Framework and Additive Software Analysis

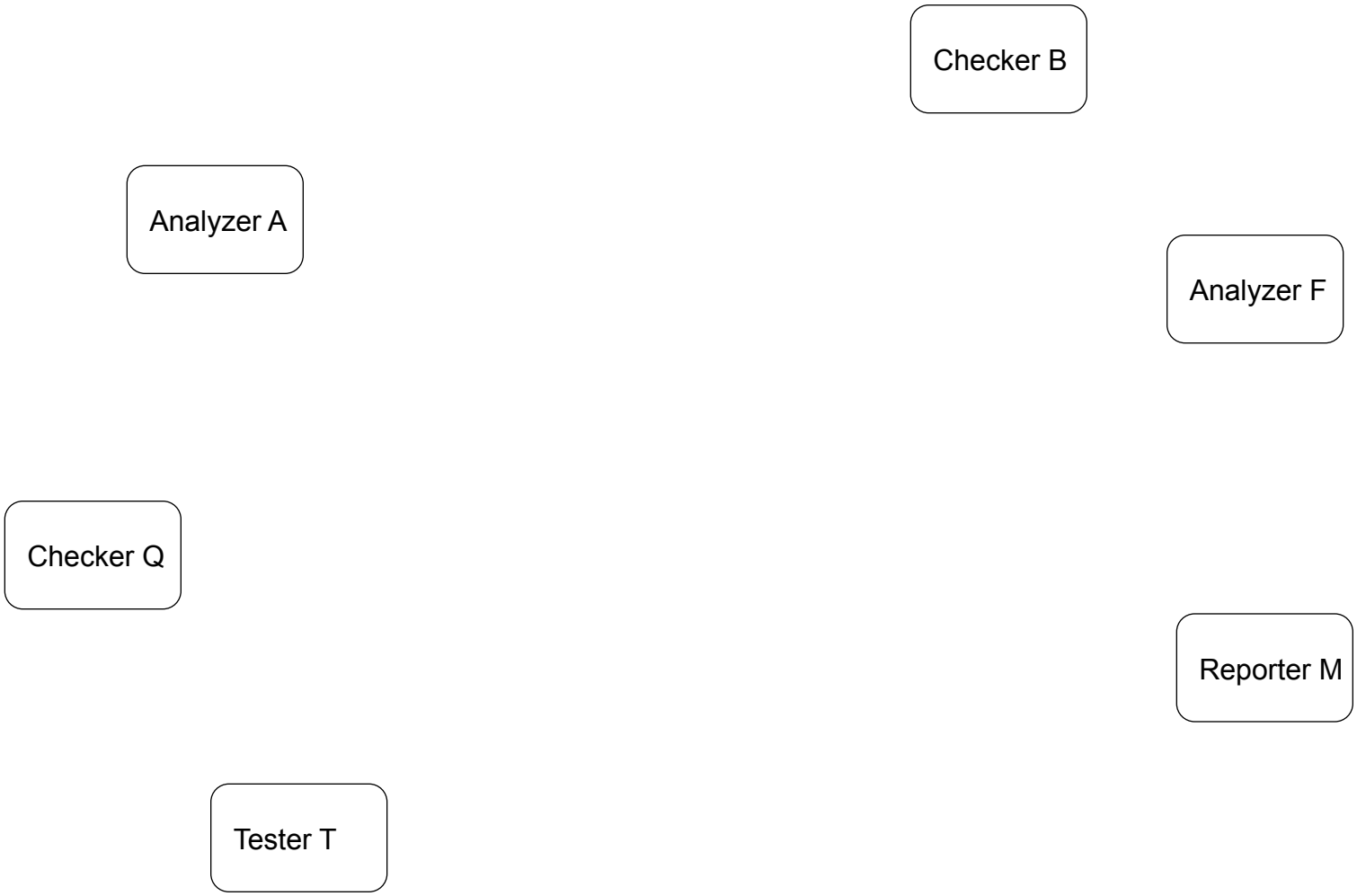
Paul E. Black
Software Quality Group
Software and Systems Division

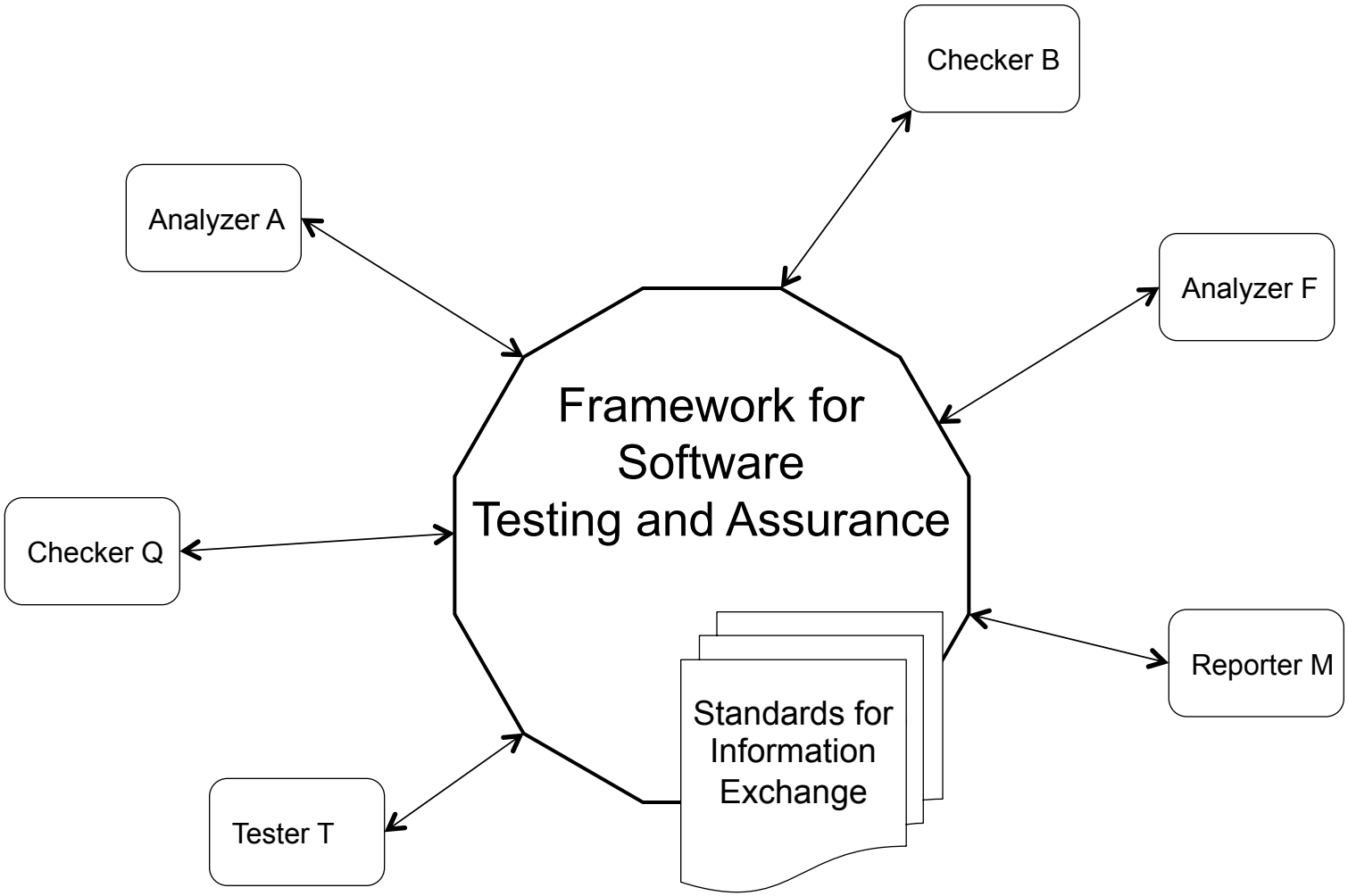
16 July 2016

Certain trade names and company products are mentioned in the text or identified. In no case does such identification imply recommendation or endorsement by the National Institute of Standards and Technology (NIST), nor does it imply that the products are necessarily the best available for the purpose.

Outline

- **A Framework for Software Assurance**
- **Additive Software Analysis**


Checker B
Analyzer A
Analyzer F
Checker Q
Reporter M
Tester T

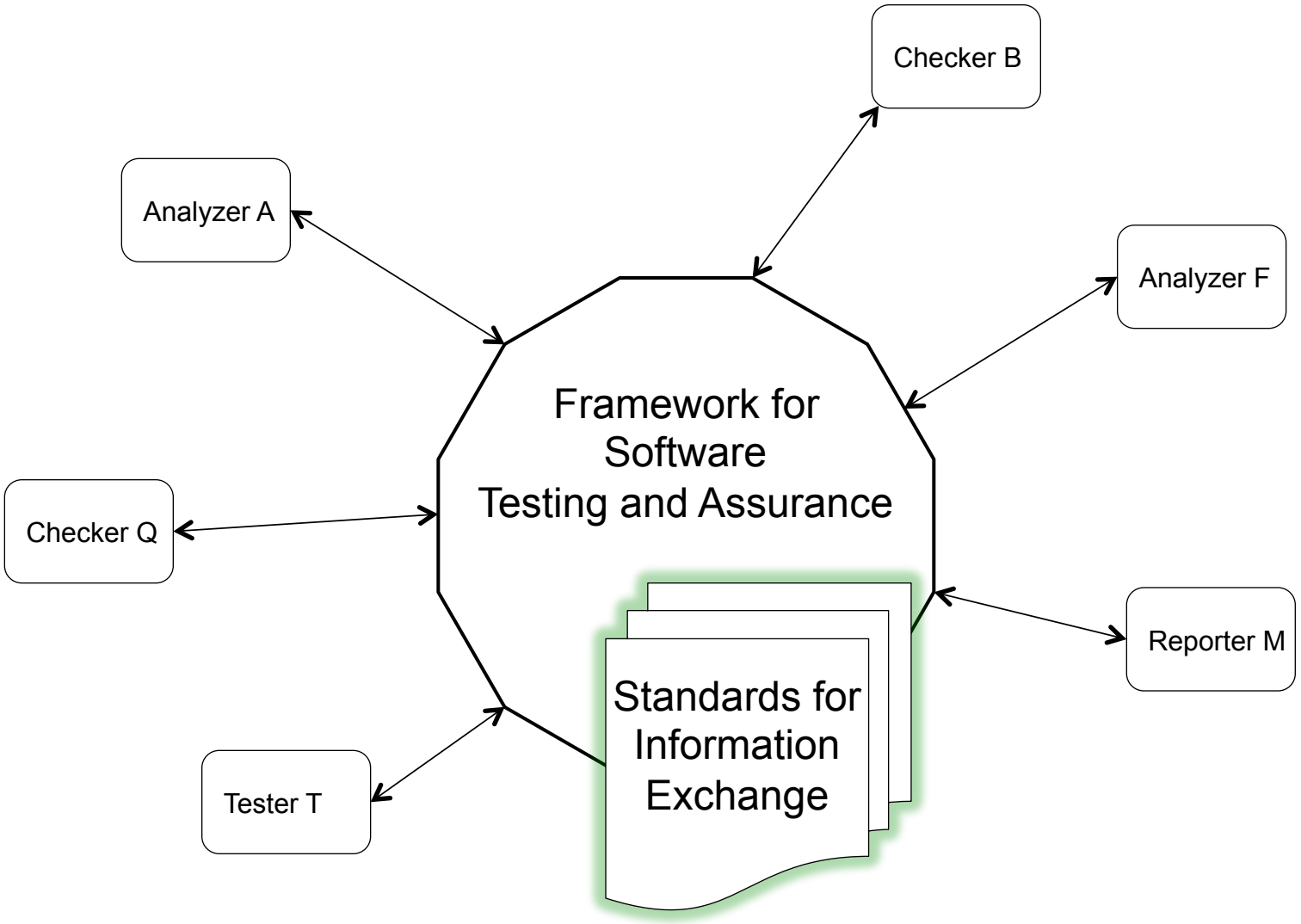

Checker B
Analyzer A
Analyzer F
Framework for
Software
Testing and Assurance
Checker Q
Reporter M
Standards for
Information
Exchange
Tester T

Functions of a Framework

- **Aggregate tool outputs.**
- **Allow software assurance checkers to interoperate.**
- **Pass program information between tools.**

Benefits of a Framework

- **Modular and distributed development.**
  - **Existing modules may be replaced by superior ones.**
  - **Facilitate synergy between groups of researchers.**
- **Enable development of “hybrid” tools.**
  - **A tool uses a static analyzer module to find problematic code locations, then uses a constraint satisfier module and a symbolic execution engine to create inputs that trigger failures.**


Checker B
Analyzer A
Analyzer F
Framework for
Software
Testing and Assurance
Checker Q
Reporter M
Standards for
Information
Exchange
Tester T

Possibly Useful Information

- **Location in code**
  - **File name, class file, method/function name, line number, etc.**
- **Variables visible at a location**
- **Possible variable values at a location**
  - **Intervals? enumerations? relations (e.g. x < y)**
- **Data flows**
- **Paths**
- **Stack traces**

Additional Information

- **Origin of binary chunk in source code**
- **Warnings of possible problems**
- **Assertion, pre- & postcondition, invariant**
- **Function signatures**

Much of This Already Exists

- **LLVM**
- **Clang**
- **gcc**
- **Rose compiler infrastructure**
- **findbugs**
- **Yasca**
- **TOIF, SAFES**
- **Code Dx**

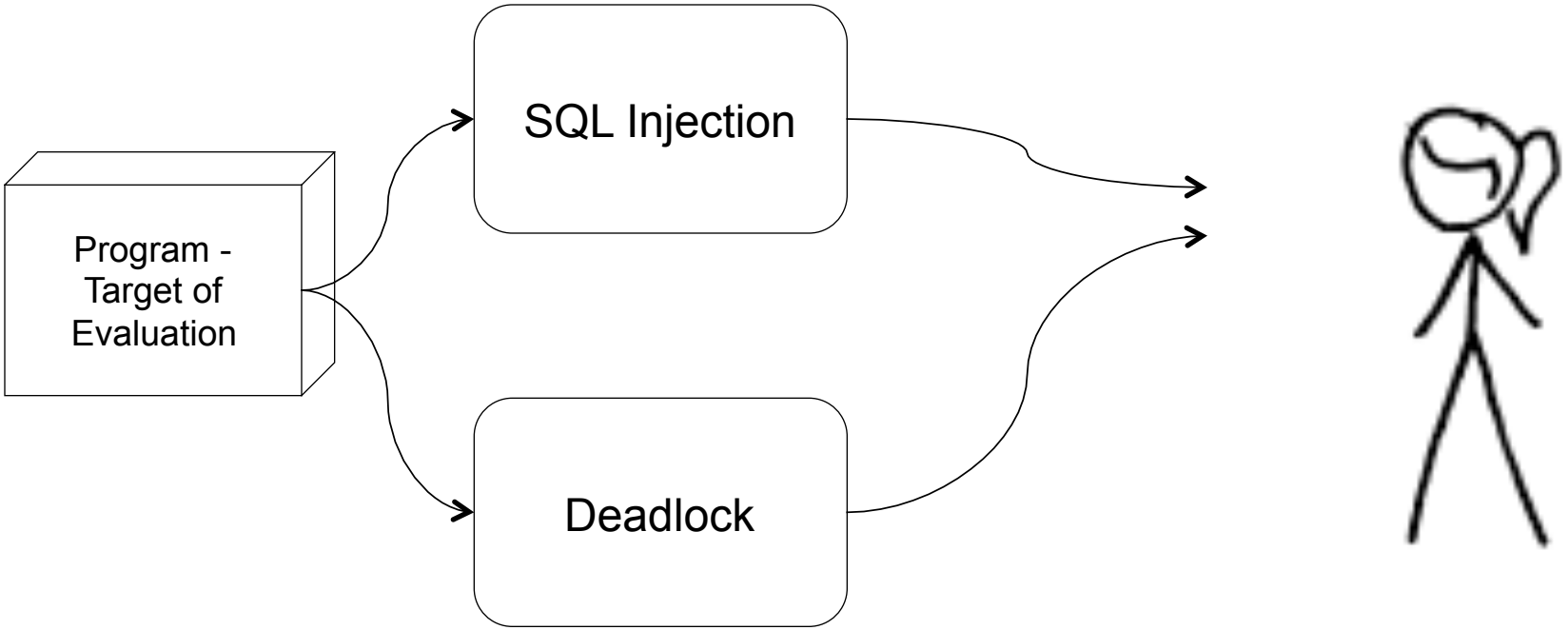
XKCD cartoon used with permission. Permanent link is http://xkcd.com/927/

ADDITIVE SOFTWARE ANALYSIS

Case 1: More Information

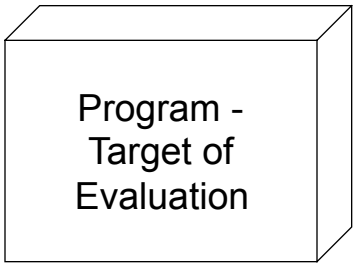
Each analyzer or checker added gives the programmer more information.

Case 1: More Information



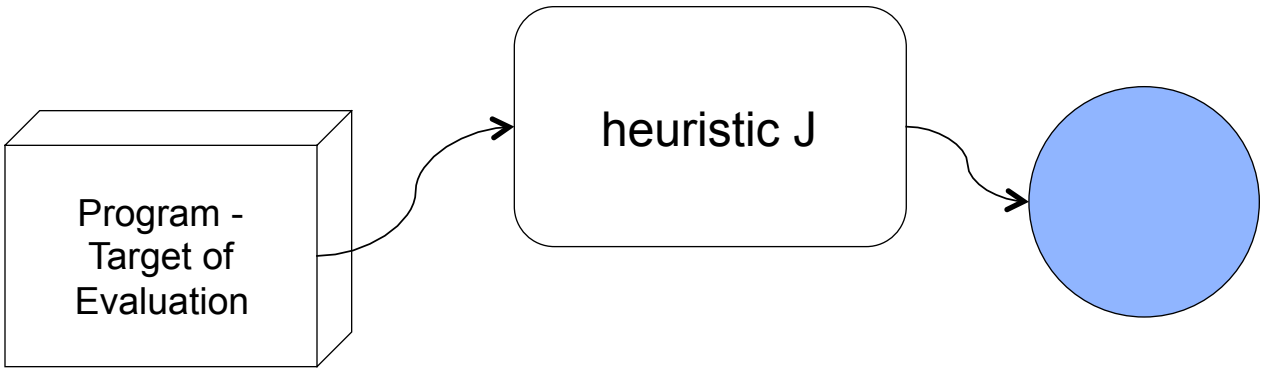
Each analyzer or checker added gives the programmer more information.

Case 2: Confirmation



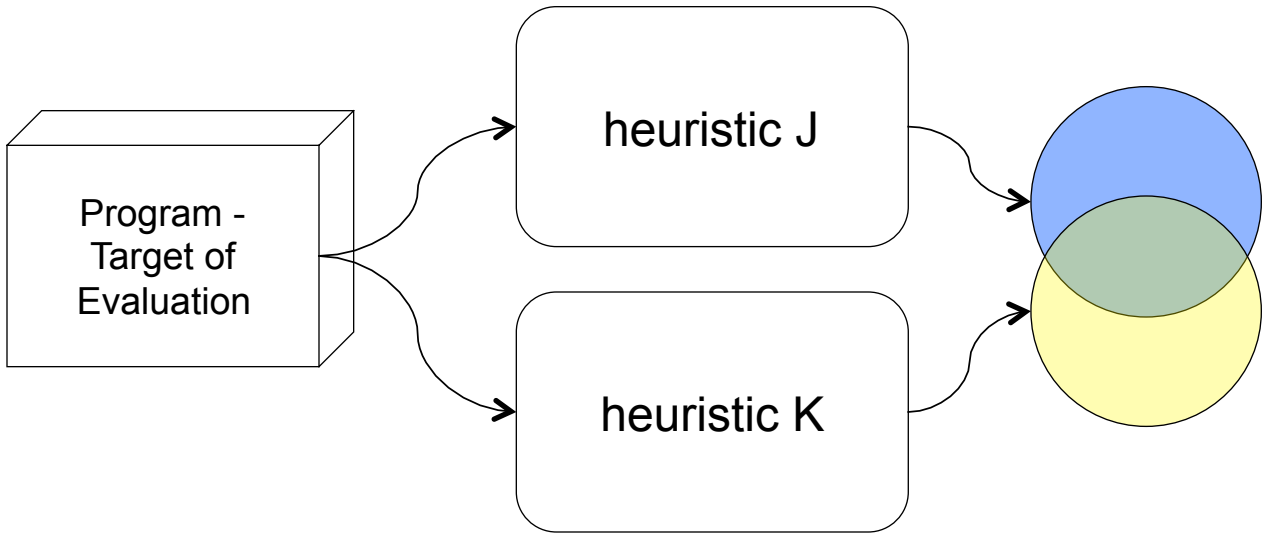
Results are correlated or compared to provide better information than either one alone.

Case 2: Confirmation



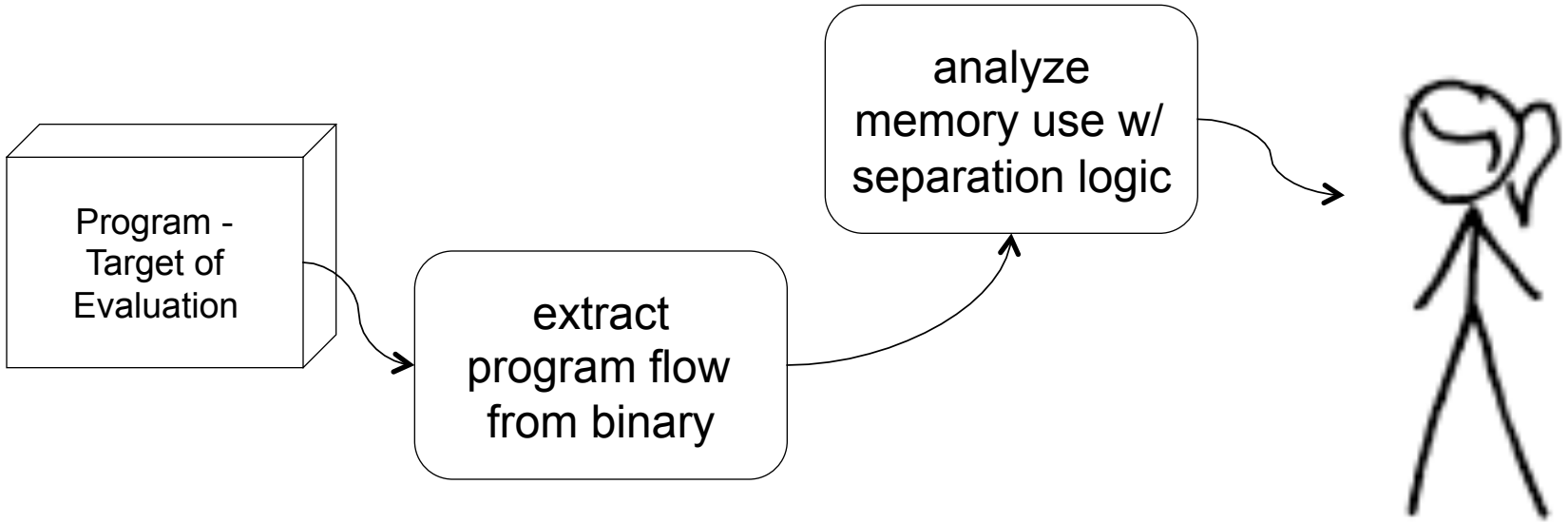
Results are correlated or compared to provide better information than either one alone.

Case 2: Confirmation



Results are correlated or compared to provide better information than either one alone.

Case 3: Synergy



Another example: tie static analysis with execution monitoring and constraint solving to get a hybrid analyzer.

Additive Software Analysis Benefits

- **Checkers and analyzers work together.**
- **Foster an "ecosystem" for tools.**
- **Growing set of problematic and virtuous programming patterns and idioms may be checked by tools.**

Possibly Useful Information

- **A descriptive taxonomy of checkers.**
  - **Inputs needed.**
  - **Languages/constructs handled.**
  - **Checking/analysis performed.**
  - **Outputs provided.**
- **A catalog of publically-vetted checkers and analyzers.**
- **A publicly accessible repository of checkers.**

Framework and Additive Software Analysis Together Are Powerful