Agile Formal Methods

Reiner Hähnle

6th International KeY Symposium
Nomborn
15th June 2007

Agile Methods

"Agile software development is a conceptual framework for undertaking software engineering projects that embraces and promotes evolutionary change throughout the entire life-cycle of the project."

Agile Methods

"Agile software development is a conceptual framework for undertaking software engineering projects that embraces and promotes evolutionary change throughout the entire life-cycle of the project."

Some Examples

- Extreme Programming (1996)
- Feature Driven Development (1999)

Agile Methods

"Agile software development is a conceptual framework for undertaking software engineering projects that embraces and promotes evolutionary change throughout the entire life-cycle of the project."

Some Examples

- Extreme Programming (1996)
- Feature Driven Development (1999)
- And, inevitably: Agile Unified Process (2001)

Agile Methods: Principles

Partial List of Agile Method Principles

- Rapid, continuous delivery of useful and working software
- Working software is the principal measure of progress
- Even late changes in requirements are welcome
- Regular adaptation to changing circumstances
- Close, daily, cooperation between business people and developers
- Continuous attention to technical excellence and good design
- Simplicity

Formal vs. Agile Methods

Most people associate Formal Methods with heavy design methods!

Formal Methods with Agile Character

Recent Formal Methods are more agile than older ones

- Design-by-Contract (Eiffel, JML, Spec#)
- Extended Static Checking based on Contracts (ESC/Java, Boogie)
- Automatic Test Generation (see Christoph's talk)

Formal and Agile Methods

Formal Methods align very well with some Agile Method Principles!

Formal and Agile Methods

Formal Methods align very well with some Agile Method Principles!

- Rapid, continuous delivery of useful and working software
- Working software is the principal measure of progress
  Automatic test generation — Bug finding

Formal and Agile Methods

Formal Methods align very well with some Agile Method Principles!

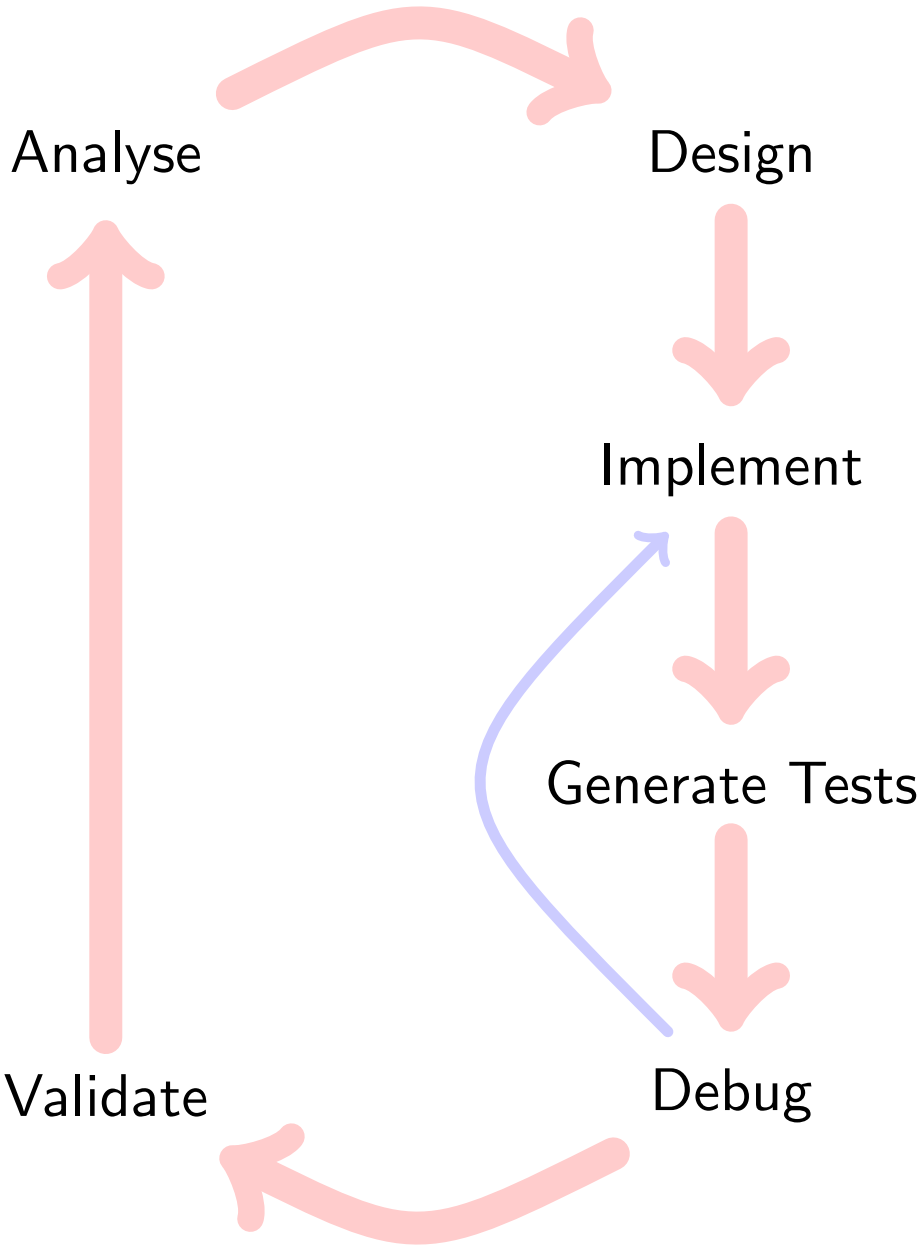
- Rapid, continuous delivery of useful and working software
- Working software is the principal measure of progress
  Automatic test generation — Bug finding
- Continuous attention to technical excellence and good design
  Precise specification — Verification

Formal and Agile Methods

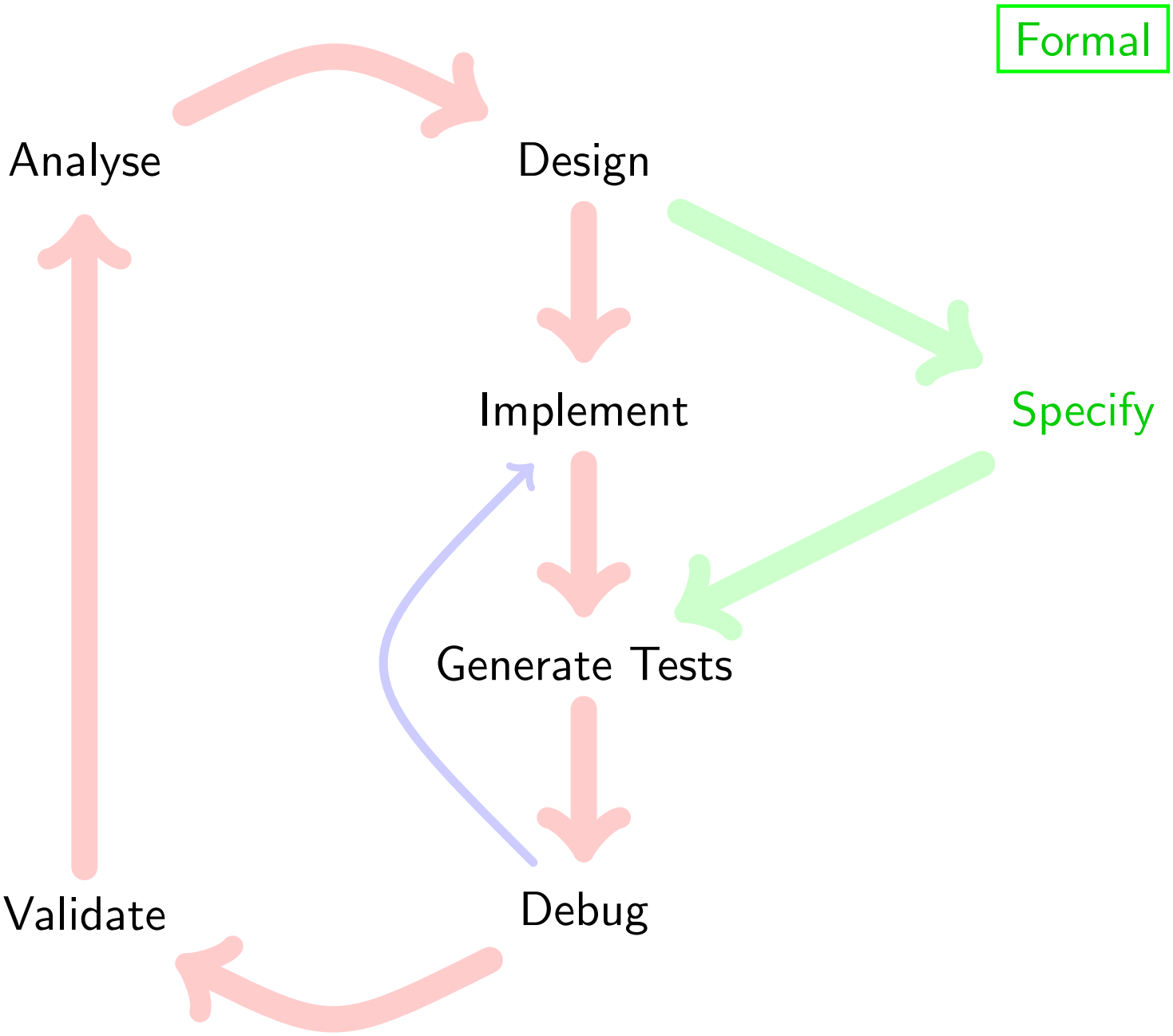
Formal Methods align very well with some Agile Method Principles!

- Rapid, continuous delivery of useful and working software
- Working software is the principal measure of progress
  Automatic test generation — Bug finding
- Continuous attention to technical excellence and good design
  Precise specification — Verification
- Simplicity
  Is a prerequisite for feasibility of verification!

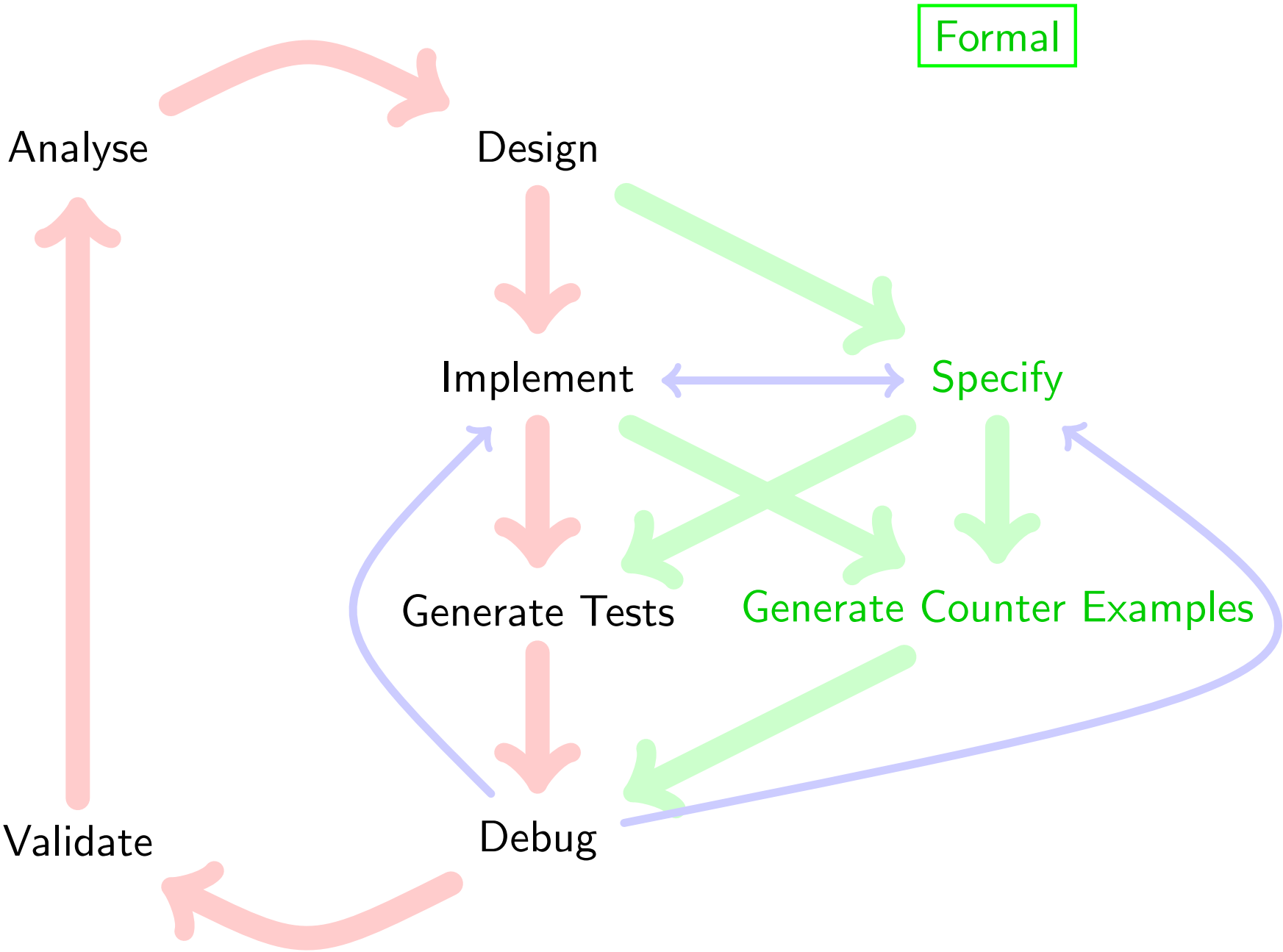
Towards an Agile Formal Method



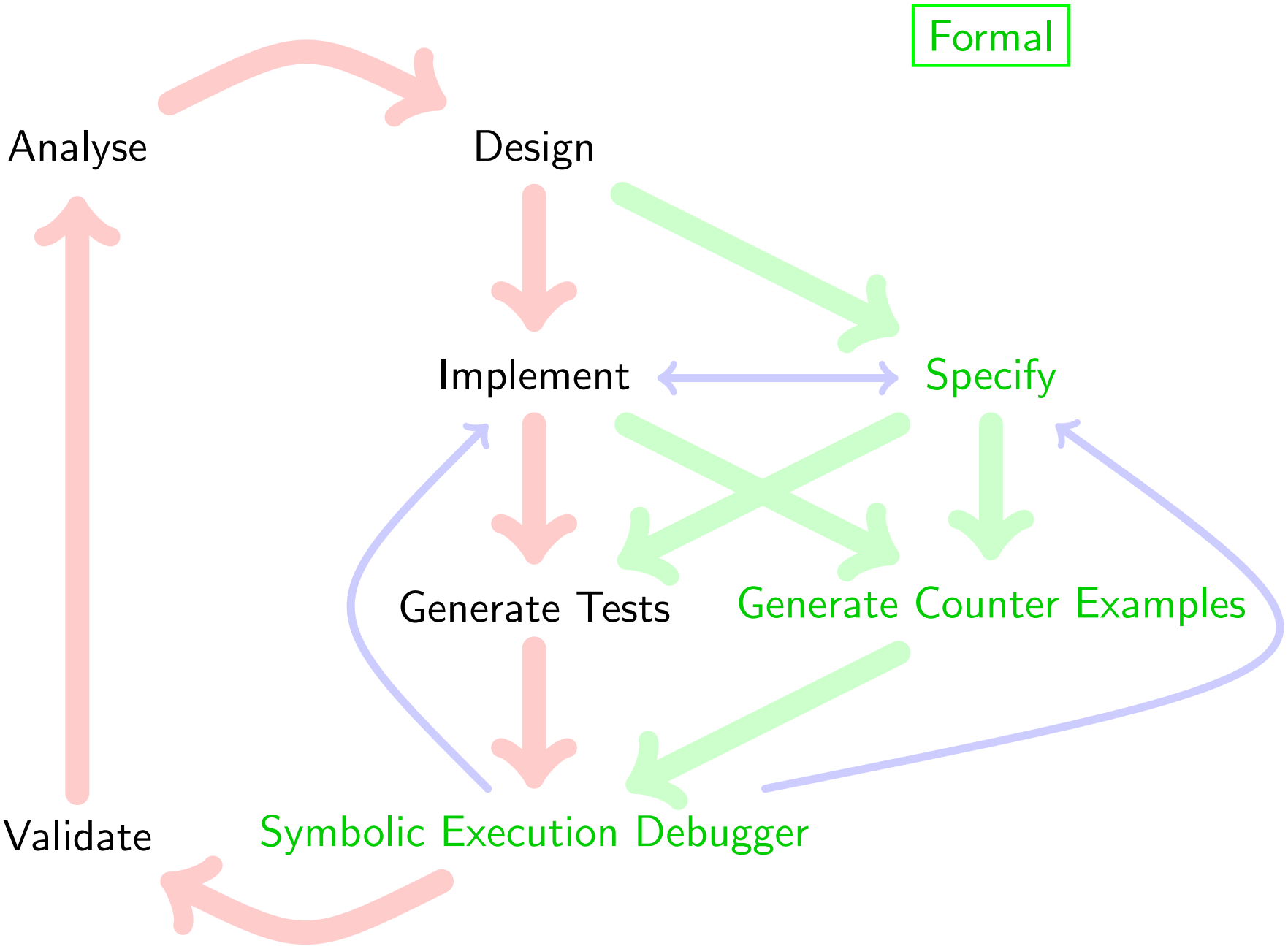
Towards an Agile Formal Method



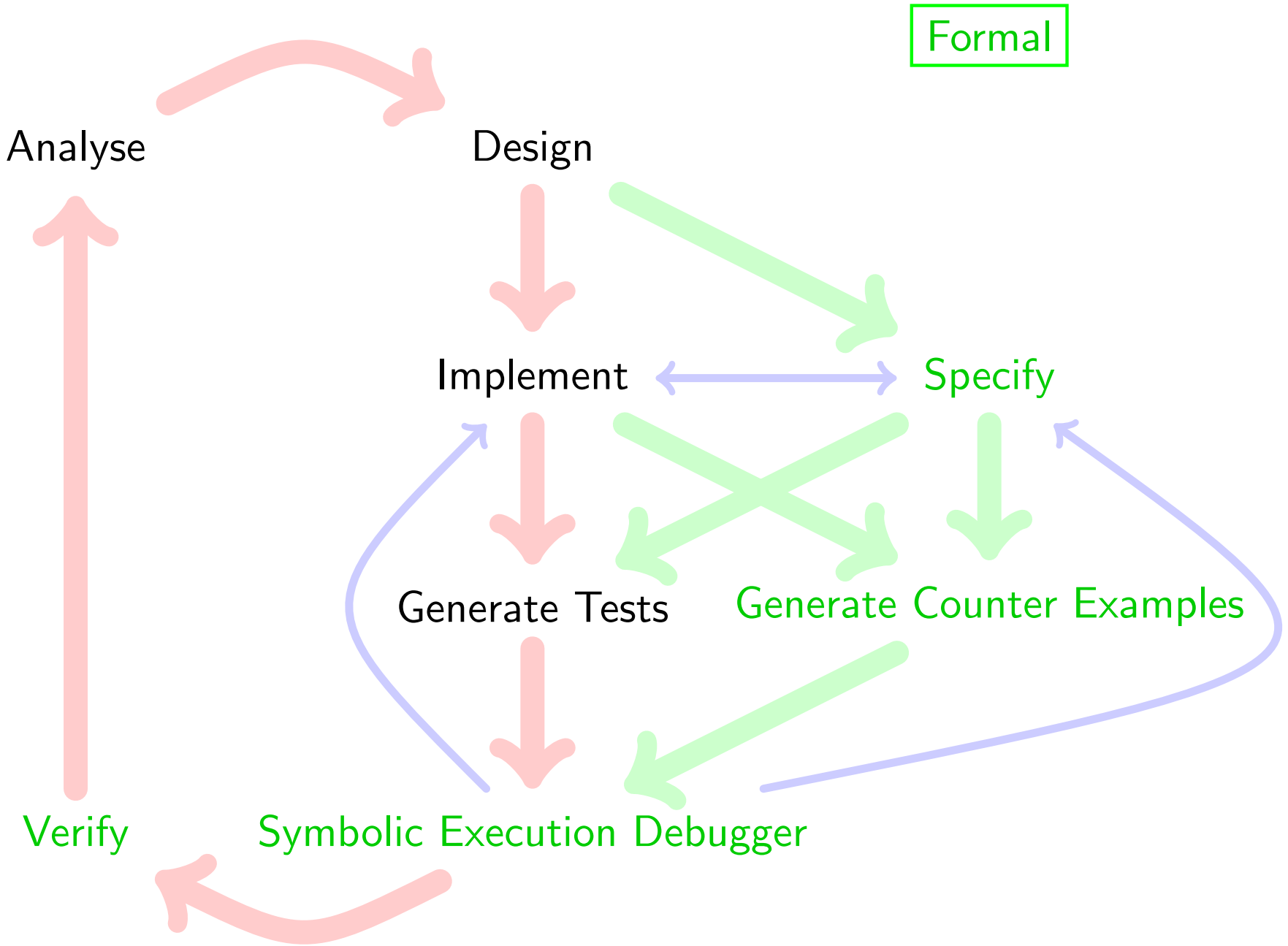
Towards an Agile Formal Method



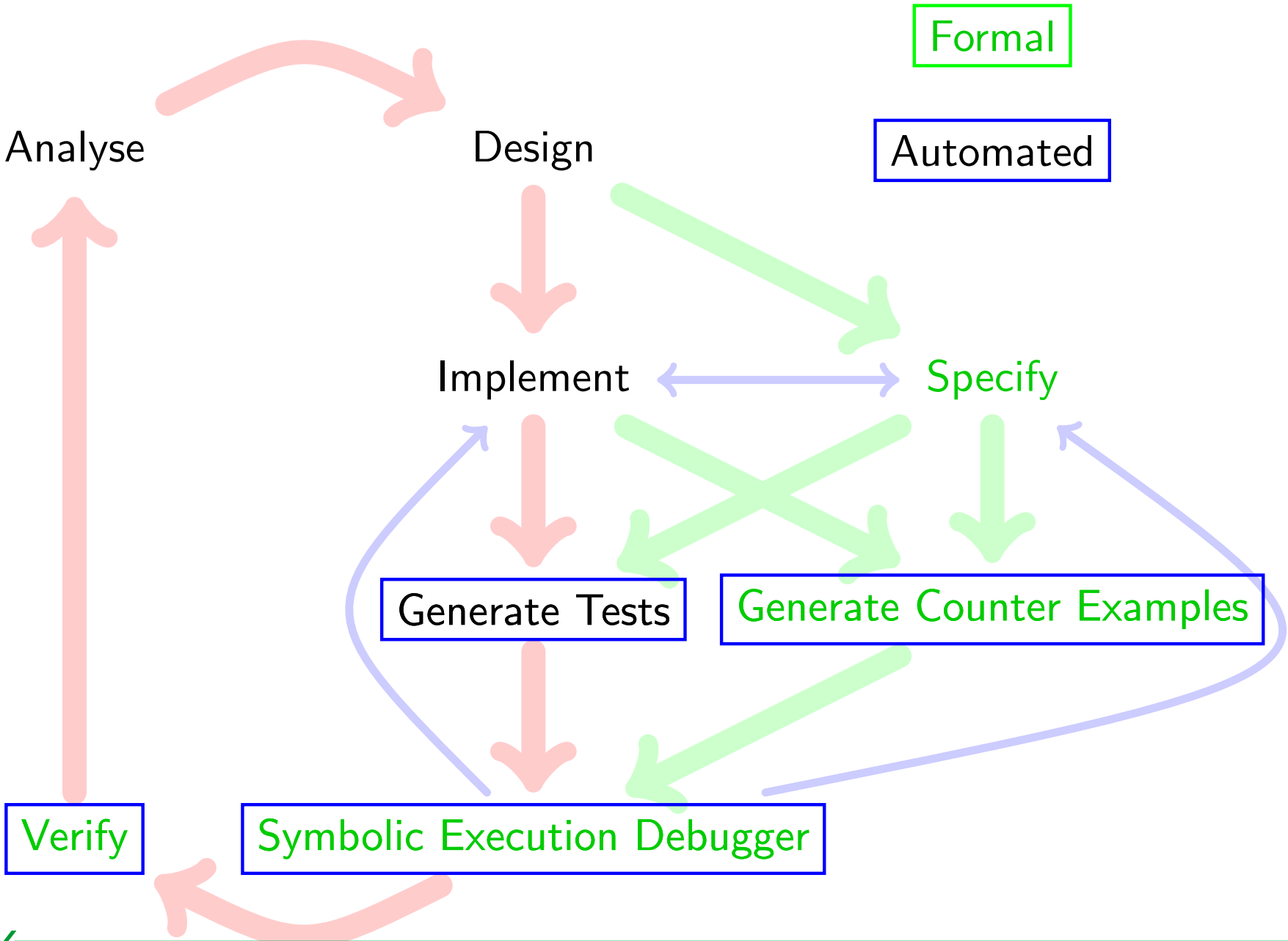
Towards an Agile Formal Method



Towards an Agile Formal Method



Towards an Agile Formal Method



Agile Formal Methods: Prerequisites

- Tight integration into **one** tool, preferably Eclipse
  - source code/specification editor
  - test generation
  - counter example generation
  - symbolic execution debugging
  - verification
- High degree of automation
  Full automation for everything but verification
- Full coverage of target language

Agile Formal Methods: Prerequisites

- Tight integration into **one** tool, preferably Eclipse
  - source code/specification editor
  - test generation
  - counter example generation
  - symbolic execution debugging
  - verification
- High degree of automation
  Full automation for everything but verification
- Full coverage of target language

KeY seems very suitable to achieve this!