Requirements and Automated Refinement

Douglas R. Smith
Kestrel Institute
Palo Alto, California

www.kestrel.edu

Refinement: Requirements → Code
Generating correct-by-construction code from requirements

- Requirements languages and processing tools
- Libraries of Domain Theories

- Libraries of Design Theories
  - system architectures
  - algorithm classes
  - data structures
  - protocol theories
  - ...

- Inference support
  - analysis
  - checking consistency
  - propagating design constraints
  - supporting application of design theories
  - ...

Kestrel Institute
Constructing Refinements

1. Library of Refinements
   - Global Search
   - Global Search Algorithm
   - Set
   - Sequence
   - Resource
   - Transportation Resource

2. Library of Refinement Generators
   - Rewrite Simplification
   - Context-dependent Simplification
   - Finite Differencing
   - Case Analysis
   - Partial Evaluation

Global Search  →  Scheduling_0

Global Search Algorithm  →  Scheduling_1

Global Search Algorithm  →  Scheduling_2

Context-dependent Simplification  →  Scheduling_3

Finite Differencing  →  Scheduling_4

Specification Carrying Software

\[ A = \langle P_A, \models_A, S_A \rangle \]

\[ B = \langle P_B, \models_B, S_B \rangle \]

\[ f \]

\[ f_P \]

\[ f_S \]

\[ f_P(b) \models_A \alpha \]

\[ b \models_{f_S(\alpha)} \]

Kestrel Institute
Open System Composition

colimit resolves the apparent circularity to produce the concurrent composition

System Composition
Cross-Cutting Constraints

Idea:
1. Express a cross-cutting constraint in temporal logic (or as an automaton)

2. Enforce the constraint by generating & adding code so that the new system refines the constraint