Principles of Stepwise Refinement

Heiko Mantel

Deduction and Multiagent Systems Lab
German Research Center for Artificial Intelligence (DFKI)
Saarbrücken, Germany

PRL Seminar
November 22, 1999
Important Research Areas (incomplete)

• tool support for formal methods [VSE group 98, 00, Inka group 99]
• real world case studies […]
• proof support [AutexierMantelStephan 98]
• managing developments [AutexierHutterMantelSchairer 99, ??]
• modularization [MantelGaertner ??]
• refinement [Mantel ??, …]
• specification techniques [MantelGaertner ??, …]
Existing Refinement Concepts (incomplete)

- Hoare 72
- GoguenThatcherWagner 78, SannellaTarlecki 88, Reif 91
- Back 88
- LynchTuttle 87
- AbadiLamport 91
- Schellhorn 99
- …
Overview (Part I)

Part I: Structure the jungle
Part II: What are the important problems?
Part III: The solutions

- Introduction
- Modeling systems by sets of traces
- What is refinement? — basic principles
- Example: refinement in TLA
- Future directions
Modeling System Executions

• A state $s$ is a mapping from variables to values.

• A trace $\sigma$ is a sequence of states (often infinite).

\[ \sigma : \quad s_0 \rightarrow s_1 \rightarrow s_2 \ldots \]

• A system can be modeled by a set of traces.

• A specification corresponds to a set of systems.
  Usually, sets of traces are considered instead of sets of sets.

• Safety and liveness properties [AlpernSchneider 85]

• Theorem: Every property is the intersection of a safety and a liveness property. [AlpernSchneider 85]
Principles of Refinement

- Refinement of the set of traces $S^c \leadsto_D S^a$
  
- Design

- Refinement of single traces $S^c \leadsto_{obs} S^a$

  $obs(\sigma^a): s_0^a \rightarrow s_1^a \rightarrow s_2^a \ldots \ldots$

  $= \quad = \quad =$

  $obs(\sigma^c): s_0^c \rightarrow s_1^c \rightarrow s_2^c \ldots \ldots$

  Computation/Representation
Classes of Observation Functions

- refinement of computation

\[ \ldots s_i^a \rightarrow s_{i+1}^a \]
\[ \ldots s_j^c \rightarrow \ldots s_{j+n}^c \]

- refinement of representation/state space

\[ \sigma^a: \quad s_0^a \rightarrow s_1^a \rightarrow s_2^a \rightarrow \ldots \]
\[ \sigma^c: \quad s_0^c \rightarrow s_1^c \rightarrow s_2^c \rightarrow \ldots \]
What is my point?

- For other specification formalisms use the intuition of these principles (Design/Computation/Representation).
- The principles can be combined!
- Looking at the basic principles helps in understanding a possibly complicated refinement concept.
- The basic principles help in identifying which properties are preserved under refinement.
- Given properties which should be preserved one can construct the appropriate refinement concept.

⇒ example

⇒ development process
The Development Process

- Requirements
- Architecture 1
- Architecture 2
- System

Properties:
- Properties 0
- Properties 1
- Properties 2
- Properties n
Which Properties are Preserved?

- Properties are preserved if they are closed under the refinement concept.
  - for design: closure under subsets
    (prerequisite for Alpern/Schneider framework)
  - for computation/representation: closure under realization for the respective observation function
    (is fulfilled if property is stated in terms of the observable parts of traces)

- Some properties are not preserved under refinement but one can compute properties which hold for the refined system.
  (e.g. representational refinement)
• syntax: $t'$, $F_f$, $\Box F$, $WF(F)$, $SF(F)$, ...

• semantic: $\| F \|$ is a set of traces

• calculi: natural deduction and sequent calculi

• implementations: VSE-2, Isabelle, ...

• some features:
  – composition principle [AbadiLamport 95]
  – refinement concept [AbadiLamport 91]
The concept of state incorporates an infinite set of state objects.

TLA does not allow arbitrary sets of traces (RTLA does). Sets of traces fulfill the certain conditions:

- closure under stuttering
- closure under arbitrary values of other variables
- closure under arbitrary values of hidden variables

\[
\langle e_0, i_0 \rangle \rightarrow \langle e_1, i_1 \rangle \rightarrow \langle e_2, i_2 \rangle \ldots \ldots
\]

TLA \quad \langle e_0 \rangle \rightarrow \langle e_1 \rangle \rightarrow \langle e_2 \rangle \ldots \ldots

Values of internal variables are not reflected in the semantic of TLA.
Refinement in TLA

A concrete specification $S^c$ refines an abstract specification $S^a$

iff

$S^c \subseteq S^a$

$\implies$ looks like refinement of design principle
Principles of Refinement in TLA

- refinement of design ✓
- refinement of computation ✓
  - equivalence under stuttering
  - in combination with hiding
- refinement of state space ✓
  - hiding
- refinement of representation ×
Addition of auxiliary variables may be necessary.

Theorem: If the machine-closed specification $S^c$ implements the internally continuous, fin specification $S^a$, the there is a specification $S^c_h$ obtained from $S^c$ by adding a history variable and a specification $S^c_{hp}$ obtained from $S^c_h$ by adding a prophecy variable such that there exists a refinement mapping from $S^c_{hp}$ to $S^a$. 
How to apply a Refinement Step?

- Invent and Verify
  - A specification is invented and then verified against a more abstract specification.
  - Intelligence is required in inventing specifications and in verifying them.
  - Specify the goal hoping that a way to it can be found.

- Transformational Approach
  - Specification is modified by application of transformation rules.
  - Intelligence is required in choosing a rule (and in verifying side conditions).
  - Apply known transformation hoping to achieve the goal.
Future Directions

- identify the basic principles in other refinement concepts
- compare refinement concepts in the framework
- prepare 2nd talk in this series
  - What are the important problems?
- prepare 3rd talk in this series
  - The solutions
- use the techniques in a real world project