Lecture 4: Formal SE

Kenneth M. Anderson
Foundations of Software Engineering
CSCI 5828 - Spring Semester, 2000

Today’s Lecture

• Introduction to Formal Software Engineering
  – Discuss Models
  – Discuss Formal Notations

Formal Software Engineering

• Software
  Computer programs and their related artifacts

• Engineering
  The application of scientific principles in the context of practical constraints

• Formal
  The use of models, techniques, and tools that are grounded in mathematics

Some Important Points

• *Formal* does not mean *Hard*
• *Formal* does not mean *Good*
• *Informal* does not mean *Bad*
  – unless it means *ad hoc*
What Are “Formal Methods”?  

1. Writing a formal specification  
2. Proving properties about the specification  
3. Constructing a program by mathematically manipulating the specification  
4. Verifying the program by mathematical argument  

Formal SE is Broader  

Not just specification and verification of programs…  

- Architecture  
- Analysis/Testing  
- Reliability and Performance Engineering  
- Configuration Management  
- Process Management, etc.  

Model/Specification/Formalism  

- Model  
  - An abstract representation  
- Specification  
  - A formal expression of a model or of a property of a model  
- Formalism  
  - A mathematical notation for writing specifications; a specification language  

Specification and the Life Cycle  

- Requirements  
- Design  
  - High level and Low level  
- Implementation  
- Test  

* Specification is used in All Activities*
Specification/Modeling Styles

- Operational
- Declarative
  - Axiomatic
  - Algebraic
- Structural/Relational

Choice of style dictated by focus of concerns

Logical Foundations

- Predicate/Propositional Logic
- Temporal Logic Systems
- Lambda calculus, etc.

Propositional Logic

- A proposition is a statement that is either true or false, but not both
- Propositional Logic is the language of propositions
  - It consists of well-formed formulas constructed from atomic formulas and logical connectives
  - The meaning of a proposition is determined by the truth values assigned to its assertions
Example

- P = “program does not terminate”
- Q = “alarm rings forever”
- P \Rightarrow Q (If the program does not terminate then alarm rings forever)

\[
\begin{array}{c|c|c}
P & Q & P \Rightarrow Q \\
T & T & T \\
T & F & T \\
F & T/F & T \\
\end{array}
\]

Library Example

- S: a book is on the stacks
- R: a book is on reserve
- L: a book is on loan
- Q: a book is requested
- Constraints
  - A book can be in only one of three states S, R, and L
  - If a book is on the stacks or on reserve then it can be requested

Library Example, continued

- Constraints specified as propositions
  - A: S \iff \neg(R \lor L)
  - B: R \iff \neg(S \lor L)
  - C: L \iff \neg(S \lor R)
  - D: Q \Rightarrow (S \lor R)
- Prove “if a book is on loan then it is not requested” is a logical consequence

(One Possible) Solution

- Proof by Contradiction
  1. L \Rightarrow \neg Q
  2. \neg(\neg L \lor \neg Q)
  3. L \land Q
  4. L
  5. \neg(S \lor R)
  6. Q
  7. (S \lor R)
  8. Contradiction

- Steps
  - This is our goal
  - We negate it…
  - …and get this
  - Conjunction Elim, 3
  - Biconditional Elim, 4, C
  - Conjunction Elim, 3
  - Modus Ponens, 6, D
  - 5 and 7 contradict
Another Solution

- Direct Proof
  - $Q \rightarrow (S \lor R)$
  - $\neg Q \lor (S \lor R)$
  - $L \iff \neg (S \lor R)$
  - $\neg L \iff (S \lor R)$
  - $\neg Q \lor \neg L$
  - $L \rightarrow \neg Q$
  - $\neg L \leftrightarrow \neg (S \lor R)$
  - $\neg (S \lor R)$
  - $\neg L \leftrightarrow (S \lor R)$

- Steps
  - D
  - Logical Equivalence of 1
  - C
  - Double Negation of 3
  - Substitution, 2, 4
  - $\lor$ is communicative
  - Logical Equivalence of 6

Predicate Logic

- Propositional Logic cannot specify the relationships between objects
  - It can only assert that particular properties hold or do not hold within a set of propositions
- Predicate Logic has the power to do so
  - consists of
    - constants, predicates, variables, and functions

Example use of predicate logic

- Consider lines and points on a plane
  - (1) two lines meet at a unique point
  - (2) there is a unique line through any two points
  - line(x) = x is a line
  - point(x) = x is a point
  - lies_on(x, y) = point x is contained in line y

Example, continued

- domain distinction
  - (a) $\forall x \bullet (\text{point}(x) \lor \text{line}(x))$;
  - (b) $\forall x \bullet (\neg (\text{point}(x) \land \text{line}(x)))$;
- incidence
  - $\forall x, y \bullet (\text{lies}_\text{on}(x, y) \Rightarrow (\text{point}(x) \land \text{line}(y)))$;
Mathematical Foundations

- Set theory
- Graph theory
- Automata theory
- Abstract algebra
- Probability and statistics

Analysis of Specifications

- Static Analysis
  \textit{Examines} specification text to reveal properties
- Dynamic Analysis
  \textit{Executes} specification text to reveal properties

\textit{Choice of analysis dictated by focus of concerns and choice of specification style}