Formal Specification and Verification

Wenhuan Lu
School of Computer Software
College of intelligence and Computing
Tianjin University

Overview

- **Semester Hours: 32 (Spring:9-16)**
- **Credits: 2**
- **Prerequisite**
  - **Set Theory**
  - **basic understanding of Mathematical Logic**

Goals of this course

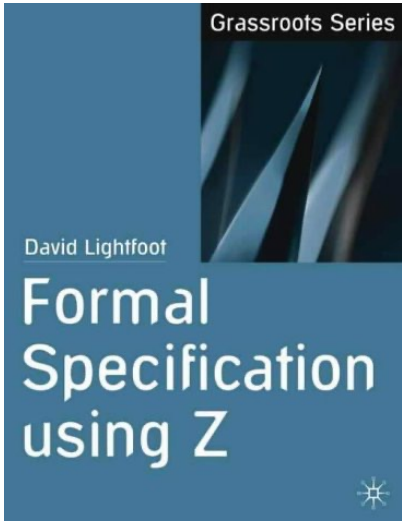
- **Learn the basics of the most widely used formal specification.**
- **Master Formal Specification Language.**
- **Have a basic knowledge on formal verification methods.**
- **Have a general idea about what kinds of problems can be effectively solved using formal methods.**

Formal Specification and Verification

- **Contents**
  - Formal Specification and Verification: What Is It and Why Study It?
  - Set Theory and Mathematical Logic (Classical Propositional Calculus and Classical Predicate Calculus)
  - Formal Specification Language Z
  - Specification and Verification Methods
- **Important Points**
  - Basic Ideas, Concepts, and Methodologies
  - Principles of Formal Specification and Verification Methods

Reference Books

- **Primary Text**
  - **D. Lightfoot, Formal Specification Using Z (2nd Edition), Palgrave, 2001**



- **Supplementary Text Material**
  - **J. Woodcock and J. Davies, Using Z: Specification, Refinement, and Proof, Prentice Hall, 1996**
  - **H. Habrias and M. Frappier (Eds.), Software Specification Methods, ISTE, 2006/ John Wiley& Sons, 2010**

Reference Books

- **Others**
  - **B. Potter, J. Sinclair, and D. Till, An Introduction to Formal Specification and Z, Prentice Hall, 1991, 1996(2nd edition)**
  - **P. Boca, J. Bowen and J. Siddiqi (Eds.), Formal Methods: State of the Art and New Directions, Springer, 2009**
  - **J. Bowen, Formal Specification and Documentation using Z: A Case Study Approach, International Thomson Computer Press, 1996**
  - **V. S. Alagar and K. Periyasamy, Specification of Software Systems, Spring-Verlag, 1998**

Important Notices and Requirements

- **Class-works**
  - Plus some points to scores of students who performed actively
- **After-class Readings and Reports**
  - Minus the same points from scores of students who submitted reports that are near resemblance
- **Qualification for Final Examination**
  - At least 2/3 of attendances and reports, respectively
- **Score**
  - Attendance + class-works/reports + final examination

1. Introduction

- **Major goal of software engineers**
  - **Develop reliable systems**
- **Formal Methods**
  - **Mathematical languages, techniques and tools**
  - **Used to specify and verify systems**
  - **Goal: Help engineers construct more reliable systems**
- **A mean to examine the entire state space of a design (whether hardware or software)**
  - **Establish a correctness or safety property that is true for all possible inputs**

1. Introduction

- **Past years of the formal methods**
  - **Obscure notation**
  - **Non-scalable techniques**
  - **Inadequate tool support**
  - **Hard to use tools**
  - **Very few case studies**
  - **Not convincing for practitioners**

1. Introduction

- **Nowadays**
  - **Trying to find more rigorous notations**
  - **Model checking and theorem proving complement simulation in Hardware industry**
  - **More industrial sized case studies**
  - **Researchers try to gaining benefits of using formal methods**
  - **…**

1. Introduction

- **Formal methods can be applied at various points through the development process**
  - **Specification**
  - **Verification**

2. Specification: What Is It?

- **Specification [IEEE Standard Computer Dictionary]**
  - **"A document that specifies, in a complete, precise, verifiable manner, the requirements, design, behavior, or other characteristics of a system or component, and, often, the procedures for determining whether these provisions have been satisfied."**

2. Specification: What Is It?

- **Specification [Dictionary of Computing, OUP]**
  - **"A formal description of a system, or a component or module of a system, intended as a basis for further development. The expression of the specification may be in text in a natural language, in a specification language, which may be a formal mathematical language, and by the use of specification stages of a methodology that includes a diagrammatic technique. Characteristics of a good specification are that is should be unambiguous, complete, verifiable, consistent, modifiable, traceable, and usable after development."**

2. Specification: What Is It?

- **Specification as descriptions**
  - A specification is a description, written in some notation (language), of the client's requirements, that may be functional requirements, (secure requirements), efficiency requirements, and implementation requirements.
- **Functional, efficiency, and implementation requirements**
  - Functional requirements address the input-output behavior of a system.
  - Efficiency requirements address the execution time of a system.
  - Implementation requirements address issue like the programming language to use, the software components to reuse, the targeted hardware platform, the operating system.

2. Specification: What Is It?

- **Specification as contracts**
  - **A specification constitutes a contract between the client and the specifier such that client must be able to understand the specification, in order to validate it.**
  - **A specification is also a contract between the specifier and the implementor such that the implementor understands the notation used for the specification, in order to implement it.**

3. Why Specification?

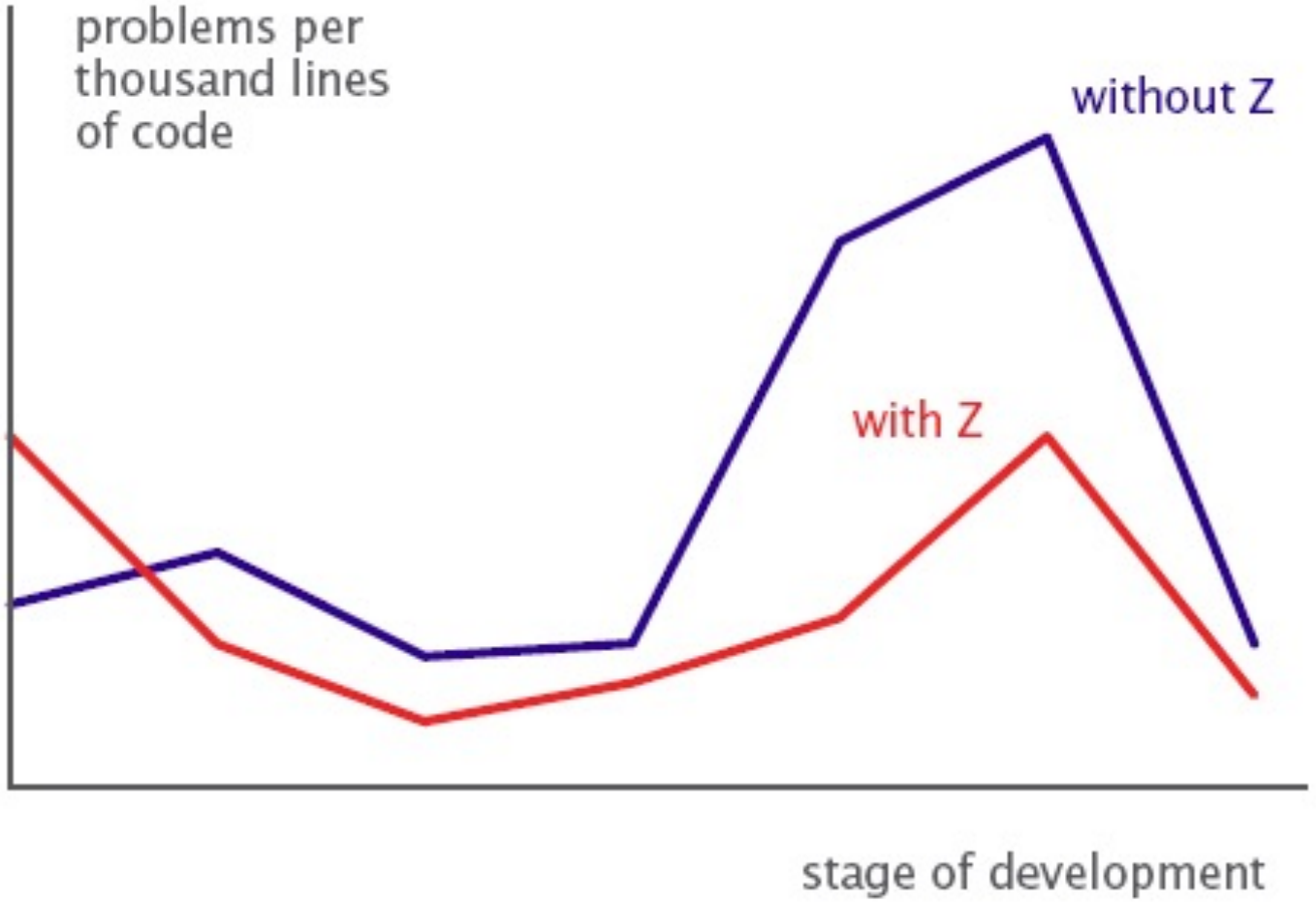
- **Goal of specifications**
  - **The goal of a specification is to capture the client's requirements in a concise, clear, unambiguous manner to minimize the risk of failure in the development process of a software system.**

3. Why Specification?

- **Merits of specifications**
  - A specification is the starting point of the development process.
  - It is much cheaper to change a specification than to change an implementation.
  - Even when the implementation is finished, the specification is very useful because conducting maintenance without a specification is a risky and expensive business, to modify a program, one must first know what it does.

3. Merits of Specifications

Qualitative results



4. Thinking…

An annual weekend event begins on the Friday evening and finishes on the Sunday afternoon. The date of the event is specified as: 'the last weekend in September'. What is the date of the Friday on which the event begins if the last day of September (30th) in that year is:

(a) a Monday;
(b) a Sunday;
(c) a Saturday;
(d) a Friday?

- **Suggest an unambiguous specification of the date of the event.**

A better specification would be, for example:

"The event takes place on the weekend which includes the last Sunday in September".

4. Thinking…

On Friday the first of the month a software engineer goes away leaving an undated message on her/his desk saying: 'Software engineer on leave until next Wednesday'. A colleague from another department passes the desk on Monday 4th of the same month and reads the message. When would the colleague expect the software engineer next to be back at work?

- **First ambiguity: What does 'next' Wed. mean, when you are reading on a Mon.—Wed. 6th or Wed. of next week, the 13th?**
- **Second ambiguity: Does on leave until Wed. mean that the software engineer's last day of leave is Wed. or does it mean that the software engineer will be back on Wed.?**
- **The colleague might reasonably expect the software engineering next to be back at work on any of: Wed. 6th, Thur. 7th, Wed. 13th, Thur. 14th.**

4. Thinking…

A video cassette recorder has a 'programming' facility which allows recordings to be made in the user's absence. Requests for recordings consist of: start-date, start-time, end-time and channel-number. Dates are specified by month-number and day-number. Times are given as hours and minutes using the 24-hour clock. If the end-time is earlier in the day than the start-time then the end-time is considered to be on the following day. Up to eight requests can be stored and are numbered one to eight.

How would you expect the recorder to behave in the following circumstances?

(a) start-date does not exist, for example, 31st April.
(b) start-date does not exist; not a leap year, 29 February.
(c) start- and end-times for different requests overlap (on same day).
(d) start-date is for New Year's Eve and end-time is earlier in the day than start-time.
(e) requests are not in chronological order. For example, request 1 is for a recording which occurs later than that of request 2.

Look at the user handbook for a video cassette recorder similar to the one described here. Does the handbook answer these questions?

5. Validation of Specifications

- **Validation of specifications**
  - A fundamental issue to make sure that the specification "matched" the client's needs(requirements). This activity is called validation.
  - Validation consists essentially of stating properties about the specification, and showing that the specification satisfies these properties. The more properties are stated, the more the confidence in the specification validity is increased.
  - Validation is an empirical process; a specification is deem valid until one finds a desired property that is not satisfied.
- **Notes**
  - We use the verb "match" instead of a stronger verb like "prove" or "demonstrate", in the definition of the validation concept.

6. Satisfaction of a Specifications

- **Satisfaction of specifications**
  - **It must be possible to demonstrate in some way that the implementation satisfies the specification of a system in order to show the confidence to the users of the system.**

6. Satisfaction of a Specifications

- **Formal refinement approach**
  - To progressively refine the specification until am implementation is reached.
  - If it is possible mathematically to probe that each refinement satisfies the specification, we say that development process is formal.
- **Informal test approach**
  - To implement the specification at first and then to test the implementation. Test cases are derived from the specification.
  - Such a development process is said to be informal.

7. Formal Notation and Semi-formal Notation

- **Formal notation**
  - **A notation is said to be formal if it has a formal syntax and semantics.**
  - **In general, "formal" in fact means "mathematical".**
- **Semi-formal notation**
  - **A notation is said to be semi-formal if it only has a formal syntax.**

8. Formal Specification

- **Formal specification [IEEE Standard Computer Dictionary]**
  - **A specification written and approved in accordance with established standards.**
  - **A specification written in a formal notation, often for use in proof of correctness.**
- **Formal specification [Dictionary of Computing, OUP]**
  - **A specification written and approved in accordance with established standards.**
  - **A specification written in a formal notation, such as VDM or Z.**

9. Why Formal Specification?

- **The role of mathematics**
  - **A major principles of mathematics are now stable and proven in use in scientific and engineering applications**
  - **Mathematics expressions have the advantage of being precise and unambiguous.**
  - **Mathematics expression are typically very concise; a great deal of meaning is concentrated in a relatively small number of symbols.**
  - **The mathematical notion of abstraction plays an important role in formal methods. Abstraction involves initially considering only the essential issues of a problem and deferring consideration of all other aspects until a later stage.**
  - **Deduction and conclusions expression in a mathematical form are capable of being proved, by application of established mathematical laws.**

10. Specification Language

- **Specification Language [IEEE Standard Computer Dictionary]**
  - **A language, often a machine-processable combination of natural and formal language, used to express the requirements, design, behavior, or other characteristics of a system or component.**
- **Specification Language [Dictionary of Computing, OUP]**
  - **A language that is used in expressing a specification. It has a formally defined syntax and semantics, and its design is based on a mathematical method for modeling or defining systems (e.g. set theory, equation and initial algebras, predicate logic).**

11. Specification Language Z: What Is It?

- **The history of Z**
  - **The Z notation (formally pronounced ‘z_d’ notation), named after Zermelo-Fraenkel axiomatic set theory.**
  - **Z war originally proposed by Jean-Raymond Abrial in France in 1977 with the help of Steve Schuman and Bertrand Meyer.**
  - **It was developed further at the Programming Research Group at Oxford University, led by C. A. R. Hoare, where Abrial worked in the early eighties (he arrived in Oxford on Sept. 1979).**
  - **Abrial answers the question “Why Z?” with “Because is the ultimate language!”**

11. Specification Language Z: What Is It?

- **The features of Z**
  - **Z is a state-based specification language.**
  - **Z is a based on the standard mathematical notation used in axiomatic set theory, lambda calculus, and first order predicate calculus.**
  - **All expressions in Z notation are typed, thereby avoiding some of the paradoxes of naive set theory.**
  - **Z contains a standardized catalog (called the mathematical toolkit) of commonly used mathematical functions and predicates.**

11. Specification Language Z: What Is It?

- **The features of Z**
  - Although Z notation uses many non-ASCII symbols, the specification includes suggestions for rendering the Z notation symbols in ASCII.
  - As well as the mathematical notation, there is a ‘scheme’ notation to aid in the structuring of the mathematics for large specification by packaging the mathematical notation into boxes that may be used and combined subsequently.

Example

Reserve$_3$

ΔLIB_3

ΞLIB_2

p?: PERSON

t?: TITLE

$p? \in members$

$p? \notin ran\ (reserved\ t?)$

$t? \in ran\ title$

$reserved' = reserved \oplus \{t? \mapsto reserved\ t? \frown \langle p? \rangle\}$

$title' = title$

$heldFor' = heldFor$

AProjected

$a : \mathbb{N}$

$\exists b : \mathbb{N} \bullet a < b$

---

12. Case Studies: CICS

- The CICS project
- CICS: Customer Information Control System
  - The on-line transaction processing system of choice for large IBM installations
- In the 1980s Oxford University and IBM Hursley Labs formalized parts of CICS with Z
- There was an overall improvement in the quality of the product
- It is estimated that it reduced 9% of the total development cost

12. Case Studies: CICS

- **This work won the Queen's Award for Technology**
  - **The highest honor that can be bestowed on a UK company.**