Formal Method in Software Engineering (SE-313)

Course Teacher
Assistant Professor
Engr. Mustafa Latif

What Is Software Engineering?

- *The IEEE 610.12 definition is:*
- *"Software engineering is the application of a systematic, disciplined, quantifiable approach to the development, operation, and maintenance of software; that is, the application of engineering to software, and the study of such approaches."*

Software engineering includes:

- Methodologies to design, develop and test software to meet customers' needs.
- Software is engineered. That is, the software products are properly designed, developed and tested in accordance with engineering principles.
- Quality and safety are properly addressed.
- Mathematics may be employed to assist with the design and verification of software products. The level of mathematics employed will depend on the safety critical nature of the product. Systematic peer reviews and rigorous testing will often be sufficient to build quality into the software, with heavy mathematical techniques reserved for safety and security critical software.
- Sound project management and quality management practices are employed.
- Support and maintenance of the software is properly addressed.

3

Overview of Formal Methods

Introduction

- The term "formal methods" refers to various mathematical techniques used for the formal specification and development of software. They consist of a formal specification language and employ a collection of tools to support the syntax checking of the specification, as well as the proof of properties of the specification. They allow questions to be asked about what the system does independently of the implementation.
- The use of mathematical notation avoids speculation about the meaning of phrases in an imprecisely worded natural language description of a system. Natural language is inherently ambiguous, whereas mathematics employs a precise rigorous notation.

4

Overview of Formal Methods

Spivey [1] defines formal specification as:

Definition 3.1 (Formal Specification)

"Formal specification is the use of mathematical notation to describe in a precise way the properties that an information system must have, without unduly constraining the way in which these properties are achieved".

5

Overview of Formal Methods

- The term "formal methods" is used to describe a formal specification language and a method for the design and implementation of a computer system.
- Formal methods may be employed at a number of levels:
  - Formal specification only (program developed informally);
  - Formal specification, refinement and verification (some proofs);
  - Formal specification, refinement and verification (with extensive theorem proving).

6

Overview of Formal Methods

- The specification is written in a mathematical language, and the implementation may be derived from the specification via stepwise refinement. The refinement step makes the specification more concrete and closer to the actual implementation. There is an associated proof obligation to demonstrate that the refinement is valid, and that the concrete state preserves the properties of the abstract state. Thus, assuming that the original specification is correct and the proof of correctness of each refinement step is valid, then there is a very high degree of confidence in the correctness of the implemented software.

7

Overview of Formal Methods

- Stepwise refinement is illustrated as follows: the initial specification S is the initial model M0; it is then refined into the more concrete model M_1, and M_1 is then refined into M_2, and so on until the eventual implementation M_n = E is produced.

$$S = M_0 \sqsubseteq M_1 \sqsubseteq M_2 \sqsubseteq M_3 \sqsubseteq \ldots\ldots\ldots \sqsubseteq M_n = E.$$

8

Overview of Formal Methods

- Requirements are the foundation of the system to be built, and irrespective of the best design and development practices, the product will be incorrect if the requirements are incorrect. The objective of requirements validation is to ensure that the requirements reflect what is actually required by the customer (in order to build the right system). Formal methods may be employed to model the requirements, and the model exploration yields further desirable or undesirable properties.

9

Overview of Formal Methods

- Formal methods provide the facility to prove that certain properties are true of the specification, and this is valuable, especially in safety critical and security critical applications. The properties are a logical consequence of the mathematical definition, and the requirements may be amended where appropriate. Thus, formal methods may be employed in a sense to debug the requirements during requirements validation.

10

Overview of Formal Methods

- The use of formal methods generally leads to more robust software and increased confidence in its correctness. Formal methods may be employed at different levels (e.g. just for specification with the program developed informally). The challenges involved in the deployment of formal methods in an organization include the education of staff in formal specification, as the use of these mathematical techniques may be a culture shock to many staff.

11

Overview of Formal Methods

- Formal methods have been applied to a diverse range of applications, including the safety and security critical fields to develop dependable software. The applications include the railway sector, microprocessor verification, the specification of standards, and the specification and verification of programs.

12

Overview of Formal Methods

- Formal methods are potentially quite useful and reasonably easy to use. The use of a formal method such as Z or VDM forces the software engineer to be precise and helps to avoid ambiguities present in natural language. Clearly, a formal specification should be subject to peer review to provide confidence in its correctness.

13

Overview of Formal Methods

Why Should We Use Formal Methods?

- To produce software adhering to high-quality standards.
- Quality problems with software may cause minor irritations or major damage to a customer's business including loss of life.
- To reduce the occurrence of defects in software products.

UK Defense Standard 00-55,00-56 "The Procurement of safety critical software in defense equipment", "Hazard analysis and safety classification of the computer and programmable electronic system elements of defense equipment" [4].[5]

14

Overview of Formal Methods

Why Should We Use Formal Methods.

- A 9% cost saving is attributed to the use of formal methods during the CICS project.
- The T800 project attributes a 12-month reduction in testing time to the use of formal methods.
- The Intel Corporation incurred a large financial cost in replacing microprocessors for its customers, as well as reputation damage because of the floating-point problem in its Pentium microprocessor.

15

Overview of Formal Methods

Industrial Applications of Formal Methods.

- Formal methods have been employed in several domains such as the transport sector, the nuclear sector, the space sector, the defence sector, the semiconductor sector, the financial sector and the telecom sector.
- IBM developed the VDM specification language at its laboratory in Vienna, and it piloted the Z and B formal specification languages on the CICS (Customer Information Control System) project at its plant in Hursley, England

16

Overview of Formal Methods

Industrial Tools for Formal Methods

Formal methods have been employed in several domains such as the transport sector, the nuclear sector, the space sector, the defence sector, the semiconductor sector, the financial sector and the telecom sector.

- IBM developed the VDM specification language at its laboratory in Vienna, and it piloted the Z and B formal specification languages on the CICS (Customer Information Control System) project at its plant in Hursley, England

17

Overview of Formal Methods

Industrial Tools for Formal Methods

The tools include

- Specialized editors which ensure that the written specification is syntactically correct;
- Refinement tools
- Automated code generators that generate a high-level language corresponding to the specification;
- Theorem provers to demonstrate the correctness of refinement steps and to identify and resolve proof obligations, as well proving the presence or absence of key properties
- Specification animation tools where the execution of the specification can be simulated

18

Overview of Formal Methods

Industrial Tools for Formal Methods

- The B-Toolkit is an integrated set of tools that supports the B-Method. It provides functionality for syntax and type checking, specification animation, proof obligation generator, an auto-prover, a proof assistor and code generation.

19

Overview of Formal Methods

Industrial Tools for Formal Methods

- The IFAD Toolbox is a support tool for the VDM-SL specification language, and it provides support for syntax and type checking, an interpreter and debugger to execute and debug the specification, and a code generator to convert from VDM-SL to C++.

20

Overview of Formal Methods

Industrial Tools for Formal Methods

- The Community Z Tools (CZT) is building a set of tools for editing, type-checking and animating formal specifications written in the Z specification language

21

Overview of Formal Methods

Approaches to Formal Methods

- Model Oriented (VDM, Z, etc)
  - The model-oriented approach to specification is based on mathematical models, where a model is a simplification or abstraction of the real world that contains only the essential details.
- Properties Oriented(Axiomatic Approach) (Algebraic Logic, Temporal Logic, etc).
  - The axiomatic approach focuses on the properties that the proposed system is to satisfy, and there is no intention to produce an abstract model of the system. The required properties and behavior of the system are stated in mathematical notation.

22

Overview of Formal Methods

The Vienna Development Method

- VDM was developed by a research team at the IBM research laboratory in Vienna.
- VDM is a model-oriented approach i.e an explicit model of the state of an abstract machine is given, and operations are defined in terms of the state.

23

Overview of Formal Methods

The Vienna Development Method

- Operations may act on the system state, taking inputs and producing outputs as well as a new system state.
- Operations are defined in a precondition and postcondition style.
- Each operation has an associated proof obligation to ensure that if the precondition is true, then the operation preserves the system invariant.

24

Overview of Formal Methods

The Z Specification Language

- Z is a formal specification language developed by Abrial at Oxford University in the early 1980s.
- It is model-oriented approach.
- An explicit model of the state of an abstract machine is given, and the operations are defined in terms of the effect on the state.

25

Overview of Formal Methods

The Z Specification Language

- Operations are defined in a precondition/postcondition style.
- Each operation has an associated proof obligation to ensure that if the precondition is true, then the operation preserves the system invariant.

26

Overview of Formal Methods

Proof and Formal Methods

- Proofs in formal methods are concerned with cross-checking the details of the specification, or checking the validity of the refinement steps, or checking that certain properties are satisfied by the specification.

27

Overview of Formal Methods

Proof and Formal Methods

- A formal mathematical proof consists of a sequence of formulae, where each element is either an axiom or derived from a previous element in the series by applying a fixed set of mechanical rules.

28

Overview of Formal Methods

Proof and Formal Methods

- The application of formal methods in an industrial environment requires the use of machine-assisted proof, since thousands of proof obligations arise from a formal specification,
- Automated Theorem provers are essential in resolving these efficiently.
- The limitation of automated theorem proving is overcome with human intervention to provide guidance or intuition that improves the effectiveness of the theorem prover.

29

Overview of Formal Methods

Model Checking

- Model checking is an automated technique such that given a finite-state model of a system and a formal property, (expressed in temporal logic) then it systematically checks whether the property is true or false in a given state in the model.
- An effective technique to identify potential design errors, and increases the confidence in the correctness of the system design.

30

Overview of Formal Methods

Finite-State Machines

- **Warren McCulloch and Walter Pitts published early work on finite-state automata in 1943.**
- **" A finite state machine (FSM) is an abstract mathematical machine that consists of a finite number of states. It includes a start state q_0 in which the machine is in initially; a finite set of states Q; an input alphabet Σ; a state transition function δ; and a set of final accepting states F (where $F \subseteq Q$)".**

31

Overview of Formal Methods

- The state transition function takes the current state and an input and returns the next state. That is, the transition function is of the form:

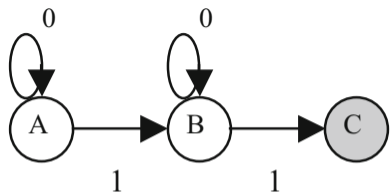

Fig. 3.1 Deterministic finite-state machine

$$\delta : Q \times \Sigma \rightarrow Q.$$

32

Overview of Formal Methods

Finite-State Machines

- A finite-state machine may be deterministic or non-deterministic,
- A deterministic machine (Fig. 3.1) changes to exactly one state for each input transition,
- A non-deterministic machine may have a choice of states to move to for a particular input.