

**MUST**  

---

**Wisdom & Virtue**

MIRPUR UNIVERSITY OF SCIENCE AND TECHNOLOGY (MUST), MIRPUR  
DEPARTMENT OF SOFTWARE ENGINEERING

# Formal Methods in Software Engineering

Lecture [2] : Formality Levels and Logic

*Engr. Samiullah Khan*

*(Lecturer)*

## *Topics discussed in Today's Lectures*

- Formal Methods Intro
- Formal Methods Techniques
- Formality Levels
- Logic
- First Order Predicate Calculus



# Formal Methods

- Formal methods are introduced to transform the problem from the **informal** space to the formal space where:
  - It becomes easier for **computational methods** and technologies to be adopted **to solve** the underlying problem
- These are used to describe the problem in a way that will **help in finding the solution**
- Initially, it is widely used with SE to specify the target system to be able to:
  - Design
  - Develop
  - Validate the underlying system



# Formal Methods

- Formal methods are **practical** and **precise way** of solving problems
- It is important to find suitable way to define & describe the underlying problem so that it becomes **easier to find solution**
- These methods can be viewed as **formal way** to describe problem or to **model** system
- These methods includes all applications of (primarily) **discrete mathematics** to SE problems
- These involves **modeling** and **analysis procedures** which are derived from mathematical foundation



# Formal Methods Techniques

- These are **mathematically based** techniques for specification, development & verification of software system.
- These can include **graphical languages**. For example, **DFDs** are most well-known graphical technique for specifying function of a system
  - DFDs can be considered a **semi-formal method**
  - Researchers have explored techniques for treating DFDs in a completely formal manner
- **Petri nets** provide another well-known **graphical technique**, often used in distributed systems, which are a **fully formal technique**
- Another formal method is the **Finite state machines**, which are commonly presented in **tabular form**



# Formality Levels

- Based on the requirements/specification detailed level, Formality level can be varied from:
  - Application to application
  - Domain to domain,
- Figure 1-2 shows different levels of formalization spectrum
- Specification language (i.e **Z language**) is used as a **set of formula** in a formal language to describe underlying system

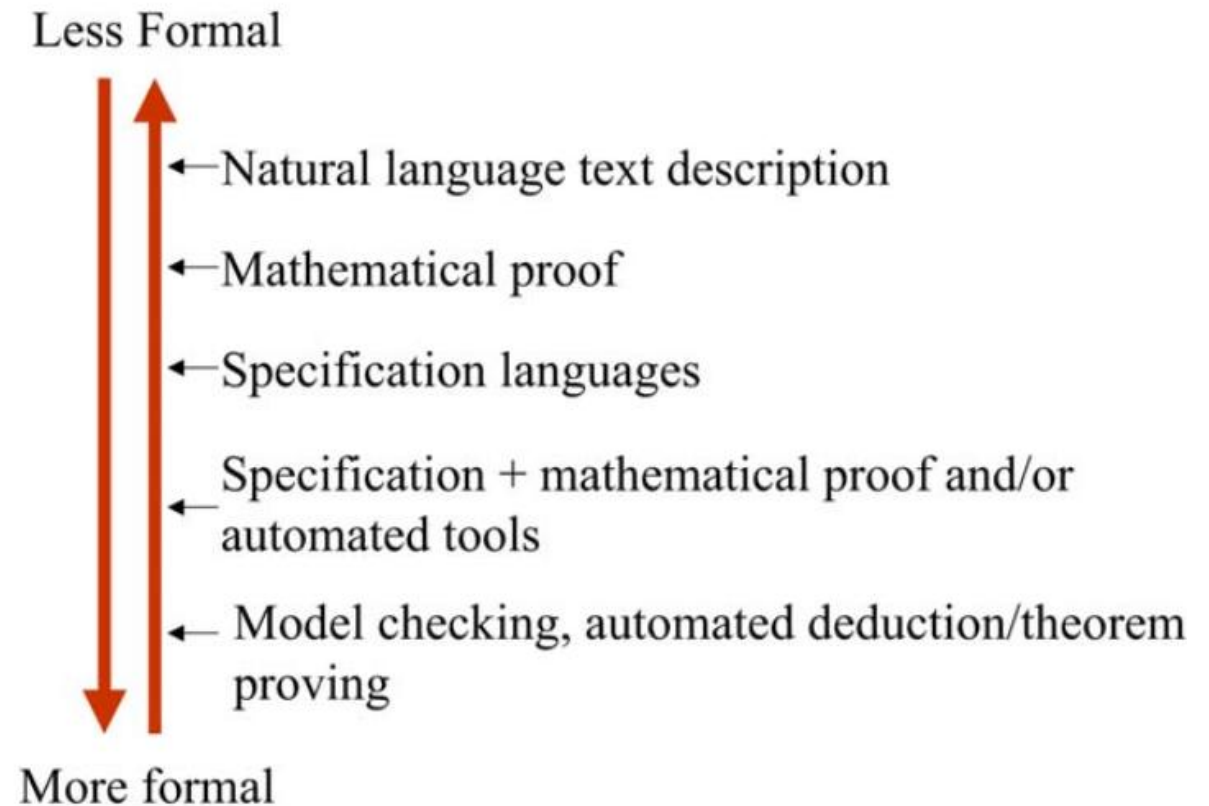


Figure 1-2. Formalization Spectrum



# Logic

- Logic or **propositional calculus** is based on statements, which have truth values (true or false)
- A **proposition**, is any declarative sentence, which is either true (T) or false (F)
- We refer to T or F as the truth value of the statement
- Calculus provides a means of determining the truth values associated with statements formed from “**atomic**” statements



# Logic - Example

- If **p** stands for “pressure is high in pipe P1” and **q** for “pipe P1 is leaking” then we may form statements such as shown in table 1-1.

Symbolic Statement	Translation
$p \vee q$	p or q
$p \wedge q$	p and q
$p \Rightarrow q$	p logically implies q
$p \Leftrightarrow q$	p is logically equivalent to q
$\neg p$ (also $\sim p$ )	Not p

Note that  $\vee$ ,  $\wedge$ ,  $\Rightarrow$ , and  $\Leftrightarrow$  are all binary connectives. They are sometimes referred to, respectively, as the symbols for disjunction, conjunction, implication and equivalence. Also  $\neg$  is unary and is the symbol for negation.



# Logic

- If propositional logic provide us with the means to assess the truth value of compound statements, **then we need some rules for how to do this**
- For example, the calculus states that “ $p \vee q$ ” is true if either  $p$  is true or  $q$  is true (or both are true)
- Similar rules apply for all the ways in which the building blocks “statements” can be combined
- The language of predicate calculus requires: Variables and Constants.



# First Order Predicate Calculus

**Table 1-2. First Order predicate calculus**

Symbol	Meaning
$\vee$	or
$\wedge$	and
$\neg$	not
$\Rightarrow$	logically implies
$\Leftrightarrow$	logically equivalent
$\forall$	for all
$\exists$	there exists



# First Order Predicate Calculus

example:  $\forall X.\text{man}(X) \Rightarrow \text{mortal}(X)$ , means all men are mortal.  $\exists X.\text{Tank}(X)$ , means there is at least one tank.

It is possible to form a new proposition from old one. For example,  $p$ : "There is Pump with 300 rpm in Plant Model Plant-1." The negation of  $p$  is  $\neg p$ , which is defined as: "There is no Pump with 300 rpm in Plant Model Plant-1." Another example: if  $p$ : " $1 + 4 < 5$ ",  $q$ : " $1 + 4 = 5$ ", then  $\neg p \wedge \neg q$ : " $1 + 4 > 5$ ".



THANKS