

Software Engineering and Formal Specification

Course code: **CSIT-31012** - (Credit Hours: 3)

Dr. Nadeem Akhtar

Objectives:

At the end of the course students would be able to:

1. Describe the costs and benefits of formal methods
2. Verify attributes of formal models
3. Demonstrate formal correctness of simple procedure
4. Construct formal models of sequential software systems
5. Implement sequential software systems based on formal models
6. Represent software systems with both state-based and process algebra models
7. Specify software systems formally, and reason about specifications, and verify their properties.
8. Connect specifications to programs through refinement and decomposition.
9. Use theorem proving tools.
10. Use model checking tools.

Course Outline:

- Software Engineering: A Preview
- Introduction
- The software Life-cycle
- Software: Its Nature and Qualities
- Software Representative Qualities
 - Correctness, Reliability, and Robustness
 - Performance, Efficiency, Usability, Verifiability, Maintainability,
 - Repairability, Evolvability, Reusability, Understandability
 - Interoperability, Productivity
- Software Engineering Principles
 1. Rigor and Formality
 2. Separation of Concern
 3. Modularity
 4. Abstraction
 5. Anticipation of Change
 6. Generality
 7. Incrementality
- Formal methods, formal specifications, formal verification.
- Formal languages, methods and techniques.
- Introduction to the use of mathematical models for specification and validation.
- Finite state machine models, models of concurrent systems, verification of models, and limitations.
- Transformational development

- Specification analysis and proof
- Program verification
- Objects and types: Sets and set types
- Tuples and Cartesian product types
- Bindings and schema types, Relations and functions
- Properties and schemas, Generic constructions, Syntactic conventions, Schema references.
- Schema texts, Predicates, Schema expressions, Generics, Sequential Systems.
- Analyzing correctness (e.g. static analysis, simulation, model checking, etc.).
- Formal analysis.
- Analyzing well-formedness (e.g. completeness, consistency, robustness, etc.).
- Comparative Formal Methods.
- Formal Proofs.
- Introduction to Coloured Petri Nets (CP-Nets)
- Introduction to Event-B

Reference Materials:

1. **Fundamentals of Software Engineering, Second edition**
Carlo Ghezzi, Mehdi Jazayeri, Dino Mandrioli
2. **Software Engineering: A Practitioner's Approach, Eighth edition**
Roger Pressman
Publisher: McGraw-Hill. 2015
3. **Artificial Intelligence, A Modern Approach, Third edition**
Stuart J. Russell and Peter Norvig
Publisher: Pearson
4. **Modern Formal Methods and Applications**
Hossam A. Gabbar
Publisher: Springer-Verlag 2006