

ARCHITECTURE, DESIGN, AND PATTERNS (SE-C-03)

AIMS:

Software architecture is an active area with growing interest by researchers and practitioners from the disciplines of software engineering and software design. In the strict sense, software *architecture* is “a description of the subsystems and components of a software system and the relationships between them.”

The increasing volatility of requirements and need to respond in timely fashion to changes taking place in the business or application domain, requires software design to produce systems that can be extended and modified relatively easily. There is an increasing need in industry for software to support evolution of requirements, the rapid addition of services, and a wide variety of customer needs.

A number of architecture modelling notations and tools as well as architectural styles have been proposed to provide the foundation for developing robust, scalable, and reliable software. Explicit focus on architecture in conjunction with emerging best practices in analysis and design has shown great potential to improve the current state-of-the-art in software product development.

The course addresses this issue as a central design goal, and introduces the student to a variety of modelling and design techniques that address this need in the context of object-oriented software development. The course covers all aspects of designing software: from architectural issues (styles, patterns, views) to design patterns that can be described as “a common solution to a common problem in a given context” on a lower level of abstraction.

LEARNING OUTCOMES:

Upon successful completion of this module, the student will be able to:

- clearly appreciate the impact of abstraction, modelling, architecture, and patterns in the development of a software product;
- critically discuss and explore key concepts in software architectures, designs, and patterns;
- critically discuss and explore architectural and design alternatives and be able to generate reasonable alternatives for a problem, and choose among them;
- to recognize the appropriate pattern for a problem and to create an appropriate one;
- apply practical skills to generate and deploy software architectures and designs based on traceable requirements.

SYLLABUS CONTENT:

- Theory of software architecture.
- Analogy with architecture in general.
- Elements of software architecture: architectural styles (ABAS – Attribute Based Architectural Style); architectural patterns (event-based, layered, pipes & filters ...); architectural description languages.
- Interaction between various types of requirements and architecture.
- Master plans vs. piecemeal growths.
- Architecture analysis and evaluation: SAAM (Software Architecture Analysis Model), scenario-based evaluation.
- Architecture process and organization.
- Model driven development.
- From architecture to design.
- Reusing architectures.
- Design patterns.
- Framework and tools.

PREREQUISITES: None

RECOMMENDED ASSESMENT: Coursework and unseen paper