CEN 6085 Software Architecture and Patterns

Credits: 3 credits

Text book, title, author, and year: Taylor, R.N., Medvidovic, N., and Dashofy, N. *Software architecture: Foundation, theory, and practice*, Wiley, 2010. E.B.Fernandez, Class notes for software architecture

Reference materials: E.B.Fernandez, "*Security patterns in practice: Building secure architectures using software patterns*". To appear in the Wiley Series on Software Design Patterns.
Selected papers

Specific course information

Catalog description: *Software architecture* is concerned with the selection of components (elements), their relationships, their interaction, and their constraints in the construction of complex software systems. We study reusable abstractions (patterns) that can be used to guide the quality aspects of a system. We use UML as a language to describe architectures. Software architecture is fundamental for the development of high-quality complex systems.

Prerequisites: Working knowledge of object-oriented concepts, in particular UML modeling. General concepts of software engineering.

Required, elective, or selected elective: Core graduate course.

Specific goals for the course: Understanding of the structure of complex software systems and their proper development.

Learn some approaches to develop complete computer systems

Learn how to integrate non-functional specifications (security, reliability, performance) in the design process.

Obtain a perspective of how a variety of mechanisms can work together to define the quality of a system

Develop ability to evaluate and compare diverse systems or mechanisms with respect to their architectural quality

Increase proficiency in using UML models to describe architectures

Brief list of topics to be covered:

1. **Motivation and objectives**, functional and non-functional requirements. Role of the software architect. Basic concepts.

2. **Types of patterns and their templates**. Pattern classifications and pattern diagrams. Reference architectures.

3. **Domain Analysis**. Use cases and requirements. Analysis patterns.

4. **Architectural patterns for distributed systems**: Layers, Broker, Client-Dispatcher- Server, Pipes and filters, Blackboard. Components and middleware

5. **Interaction patterns**: Observer, MVC, Adapter.

6. **Security and reliability**: patterns and misuse patterns. Secure domain models.

7. **Service-oriented architectures:** patterns for web services and their standards

8. **Cloud computing and its effect on software architectures**.

9. **Model-Driven Architecture** (MDA). Metamodels.