**Software Modelling and Design**

**Syllabus**

**Textbook** for the course: **Bernd Bruegge, Allen H. Dutoit**. *Object-oriented Software Engineering Using UML, Patterns, and Java*, 3rd ed, Pearson, 2013

**Aims and objectives**

**Knowledge and Understanding**

Having successfully completed the module, you will be able to demonstrate knowledge and understanding of:

A1. The software development process

A2. Requirements elicitation and structuring

A3. Requirements modelling in UML and set-theoretic notations, and their validation

A4. The use of UML design notations

A6. Concepts of software architecture, design patterns and their applicability

A7. Characteristics of appropriate APIs, Tools, IDEs

**Subject Specific Intellectual and Research Skills**

Having successfully completed the module, you will be able to demonstrate knowledge and understanding of:

B1. Specify, analyse and organise requirements for a software product

B2. Model, analyse and validate such software requirements using UML and set-theoretic notations

B3. Apply appropriate UML design patterns and notations to the design of components of a product

B4. Select and use appropriate APIs and Tools in mapping these designs to code

**Subject Specific Practical Skills**

Having successfully completed the module, you will be able to demonstrate knowledge and understanding of:

D1. Apply an appropriate software engineering process and tools to the task of structuring, modelling and validating requirements for a software product

D2. Apply appropriate software engineering techniques and tools from a suitable requirements model

**Syllabus**

- From problems (requirements) to solutions (programs): process  
- Software process models, e.g. waterfall, agile  
- Requirements elicitation, modelling, analysis and validation  
- Structuring narrative requirements  
- Structure: class diagrams, state: state machines  
- Modelling, analysis and validation with UML and Event-B  
- Use of tools for analysis and validation of models  
- Modelling and analysis case studies  
- UML design notations: use cases, scenarios: sequence diagrams  
- Design patterns, generics  
- Architecture  
- APIs, Tools