IIA1317
Object Oriented Analysis, Design and Programming

Nils-Olav Skeie
Associate Professor, PhD
Agenda

- Background,
- Introduction,
- Analysis and Design,
- Course Information,
- Assignments,
- Learning objectives.
Software development

• 1985-1989: Mycron: (office computers),
  – CCPM/Unix/MSDOS/Windows: C and assembler.
• 1989-1991: Datek: (air traffic),
  – MSDOS/Network: C and assembler.
• 1991-2003: Skarpenord (level gauging systems for the maritime industry);
  – BallastMaster v1 (1999-2003): RTOS: C: Some OOAD: ≈ 300.000 lines of code,
  – Windows, Linux and Solaris: C, Some OOAD: ≈ 300.000 lines of code,
• 2003-2006: Ph.D.: Measurement system:
  – Windows: C#, OOAD, ≈ 50 classes.
• 2006-2015: Telemark University College:
  – Master course (5 credits) in OOADP with UP, UML and C#.
• 2003-2015: KROHNE Skarpenord: (part time)
  – CARGOMASTER v5: Windows: maintenance/extensions: ≈ 1.5 mill. lines of code,
  – System Architect; control systems for the maritime industry.
• 2015-……: University College of Southeast Norway:
  – Master course (5 credits) in OOADP with UP, UML and C#.
• 2012-……: SMART research group:
  – Windows: C#, OOAD, ≈ 100 classes.
Introduction
Software Engineering

• Systematic methods for:
  – Development Process
  – Collecting requirements,
  – Analysis,
  – Design,
  – Development
  – Implementation,
  – Testing,
  – Maintenance,
  – Configuration,

of software applications
Can we structure and document software applications before or without writing/developing the code?

- Analyze the system?
- Define the static and dynamic behavior?
- Define the main classes/objects?
- Inheritance?
- Documentation?
Specification / Requirements / Analysis

- Specification,
- Working domain,
- Understanding,
  - Specification,
  - Working domain,
  - Working process,
  - User needs,
  - User wishes.
Course Overview

- Programming repetition,
  - C# and Git as source control,
- Development Process
  - UP, (Unified Process), iterative development process,
- Collect Requirements,
  - FURPS+; Functional and Non-function requirements,
- Analysis
  - Use case; UML (Unified Modeling Language) and text documents,
  - Domain model; UML
  - System Sequence Diagram; UML

- Design
  - Patterns; tool for assigning responsibilities to classes/objects,
  - Interaction diagram; UML
  - Class and object diagrams; UML
- Implementation
  - C#
- Testing
  - Test plan and test cases
- Configuration.
The critical design tool for software development is a mind well educated in design principles. It is not the UML or any other technology.
## Course schedule

- **Software Engineering;**
  - Development Process,
  - Collect requirements,
  - Analysis,
  - Design,
  - Implementation,
  - Testing,
  - Documentation.

- **Assignments;**
  - Approved / Not Approved.

### Schedule 2017

<table>
<thead>
<tr>
<th>Week</th>
<th>Topic</th>
<th>Chapters</th>
<th>Assignments</th>
<th>Note</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>OOADP Introduction</td>
<td>1</td>
<td>Install Visual Studio and Git</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>UP introduction and Version Control Sys.</td>
<td>2 - 5</td>
<td>Install Visual Studio and Git</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>C# training (exercises in front)</td>
<td>Exercises</td>
<td>Specification</td>
<td>No in front</td>
</tr>
<tr>
<td>5</td>
<td>C# training (exercises in front)</td>
<td>Exercises</td>
<td>#1 in front</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>C# training (exercises in front)</td>
<td>Exercises</td>
<td>#2 in front</td>
<td>Class</td>
</tr>
<tr>
<td>7</td>
<td>Requirements, use cases, and domain model. Midsemester evaluation</td>
<td>6 - 7</td>
<td>Hand in #1</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>Domain model, System Sequence Diagram (SSD), and UML tools</td>
<td>7 - 8</td>
<td>#3 in front</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>Interaction and Class diagrams</td>
<td>9</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>GRASP patterns, Design Model and Programming (Exceptions/Threads)</td>
<td>9</td>
<td>Hand in #2</td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>Use case, Inheritance, and Package</td>
<td>10 - 11</td>
<td>#2 in front</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>GRASP and GoF patterns</td>
<td>12 - 13</td>
<td>#3 in front</td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>Activity, State and Deploy diagrams</td>
<td>14</td>
<td>Hand in #3</td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>Data structures and algorithms</td>
<td>15</td>
<td></td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>Easter holiday (No teaching)</td>
<td>No teaching</td>
<td></td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>No teaching</td>
<td>No teaching</td>
<td>Work with #4</td>
<td></td>
</tr>
<tr>
<td>17</td>
<td>Logical architectures, inheritance, and software development.</td>
<td>16 - 17</td>
<td>Work with #4</td>
<td></td>
</tr>
<tr>
<td>18</td>
<td>No teaching</td>
<td>No teaching</td>
<td>Hand in #4</td>
<td></td>
</tr>
<tr>
<td>19</td>
<td>No teaching</td>
<td>No teaching</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Final test: 29-MAY-17</td>
<td>3 hours</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Assignments

• Specification for a self elected project
• Assignment #1;
  – Programming in C#,
• Assignment #2;
  – Analysis using UP and UML,
• Assignment #3;
  – Designing using UP, UML and patterns,
• Assignment #4;
  – Develop an application based on the specification in the start of the semester,
  – Using UP, UML, patterns and C#,
  – Include a video of the UI and process.
Learning objectives

• Knowledge; The candidate will be able to:
  – explain the meaning of a development process in software development,
  – describe the concept of analyzing and designing object-oriented software using Unified Modeling Language (UML),
  – give an overview of software testing and software source control.

• Skills; The candidate will be able to:
  – use Unified Process (UP) as a software development process in software development,
  – analyze and design an object-orientated software application for the process industry,
  – implement the results from the design phase as a software application.