COM 1205

March 21, 2001

Software Design and Development — Syllabus Winter 2001
Professor Karl. J. Lieberherr

The syllabus is subject to change based on class reaction.

`http://www.ccs.neu.edu/research/demeter/course/topics-covered/topics-covered`

contains a good overview of computer science topics covered in this course. Six of nine important areas are touched. Three fundamental computer science processes are touched: theory, abstraction and design. And 5 of 12 recurring computer science concepts are touched.

The AP book is the recommended text book: "Adaptive Object-Oriented Software: The Demeter Method" which you can browse at:

`http://www.ccs.neu.edu/research/demeter/biblio/dem-book.html.`

You don't have to buy the book; instead you can print selected chapters on your home printer and other chapters you can just browse on-line.

This course provides state-of-the-art techniques and concepts for object-oriented software design and development with a focus on aspect-oriented techniques

`http://www.ccs.neu.edu/home/lieber/AOP.html.`

You learn the Demeter Method for object-oriented software development which will hopefully make you at least 4 times as productive as a Java software developer compared to software development without the method. In the first few weeks you learn the method which you apply in the remaining weeks to your project. The project will be done in groups.

Assume that you could do your project in 200 hours of Java programming without using a good method. Instead of working hard, we will work smart. We will spend 50 hours on learning the Demeter Method and then you spend 50 hours on doing your project in Java using the Demeter Method. Since you will be at least 4 times as productive, you will still finish your 200 hour project in 50 hours only. Instead of spending 200 hours on the course, which would be excessive, you can accomplish the same in a total of 100 hours AND learn very useful object-oriented technology.

Designing and programming will be done in a structure-shy, grammar-based, object-oriented style. All assignments and the project will be written directly or indirectly in Java.

This course does not have Java as a prerequisite, however you are expected to learn a small subset of Java from the recommended text book or from another good Java book of your choice.

The WWW page for the course is:

http://www.ccs.neu.edu/home/lieber/com1205/w01

This file is in:

admin/syllabus/

Viewgraphs for the lectures are in:

lectures/

The lectures gradually introduce you to programming adaptively. First we program adaptively in pure Java using the DJ library. Hw 1 and hw 2 exercise this knowledge. In later homeworks and the project we use a small extension to Java to write our adaptive programs as behavior files. Structure will be defined by class dictionaries similar to XML schemas. Behavior files allow us in many cases to keep information about one concern in one file. The code will be spread automatically into multiple Java classes.

1. Week: Introduction to adaptive software development. The bus simulation example.

    - Pattern: Structure-shy Traversal.
      http://www.ccs.neu.edu/research/demeter/adaptive-patterns/pattern-lang-conv file: Structure-Shy-Traversal.html
    - Pattern: Selective Visitor.
      http://www.ccs.neu.edu/research/demeter/adaptive-patterns/pattern-lang-conv file: Selective-Visitor.html
    - Pattern: Class Graph (Class dictionaries and Unified Modeling Notation.)
      http://www.ccs.neu.edu/research/demeter/adaptive-patterns/pattern-lang-conv file: Class-Graph.html
    - Overview of pattern language, see: http://www.ccs.neu.edu/research/demeter/adaptive-patterns/pattern-lang-conv file: pattern-language-for-AP.html

    AP book chapters: selections from 1-4. UML Distilled: chapter 1: Introduction, and class diagram chapter 4: The essentials.

2. Week:

    - Pattern: Structure-Shy Object
      http://www.ccs.neu.edu/research/demeter/adaptive-patterns/pattern-lang-conv file: Structure-Shy-Object.html
    - Pattern: Growth Plan
      http://www.ccs.neu.edu/research/demeter/adaptive-patterns/pattern-lang-conv file: Growth-Plan.html

    Writing simple adaptive programs using traversals and visitors. Introduction to DJ. Class dictionary design.

    AP book chapters: selections from 5-8. UML Distilled: class diagram chapter 5: Advanced Concepts

3. Week:

    Strategy graphs. Decoupling classes: Law of Demeter Relationship to adaptive software. DJ continued. The class dictionary notation (graphical and textual). Design rule checking of class dictionaries. Class dictionaries as customizers for adaptive programs.

    AP book chapters: selections from 8-11. UML Distilled: chapter 6: Interaction Diagrams

4. Week:

    Visitor pattern. Adaptive Programming. Traversal strategies overview. More complex uses of visitors: multiple visitors and nested visitors. Improving the reusability of software designs. Parameterized class definitions.

    AP book chapters: selections from 10-12. UML Distilled: chapter 9: Activity Diagrams.

5. Week: Traversal strategies in detail. The Demeter Method for AP. Design notations for behavior. Growth Plan pattern. Developing a growth plan for implementation and testing. Design with maintenance in mind.

    AP book chapters: 13. UML Distilled: chapter 2: An Outline Development Process

6. Week: Midterm. Testing of adaptive, object-oriented software. UML Distilled: chapter 11: UML and Programming

7. Week: Class dictionary design and class dictionary transformations. The 11 kinds of class dictionaries. Project program design and implementation.

8. Week: Project program design and implementation.

9. Week: Project program design and implementation. Collaborations and adapters. Personalities. Further design patterns for adaptive object-oriented software development.

    http://www.ccs.neu.edu/research/demeter/adaptive-patterns/

10. Week: A comparison of object-oriented software development methodologies.
11. Exam week: Final.