COM 3360

September 29, 1999

Adaptive Object-Oriented Software Development: The Demeter Method — Syllabus Fall 1999
Professor Karl. J. Lieberherr

The syllabus is subject to change based on class reaction and projects selected by the students. The course progress file from last year

`http://www.ccs.neu.edu/research/demeter/course/f98/course-progress`

gives an accurate account how the material was covered last year. I will use the lectures from last year and adapt them to the state-of-the-art. A major change will be to view the subject matter from the point of view of UML collaborations.

The AP book is the 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.`

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/research/demeter/course/f99

This file is in:

admin/syllabus/

Viewgraphs for the lectures are in:

lectures/

1. Week: Introduction to adaptive software development.

   - 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: 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. 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. 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.

4. Week:

   More complex uses of visitors: multiple visitors and nested visitors. Garbage collection. Improving the reusability of software designs. Parameterized class definitions.

   AP book chapters: selections from 10-12.

5. Week: The Demeter Method for AP. Design notations for behavior. Developing a growth plan for implementation and testing. Design with maintenance in mind.

   AP book chapters: 13.

6. Week: Midterm. The Java Bean approach to an object-oriented component model. The coordination aspect. Exception handling design. Testing of object-oriented software.

7. Week: Graphical user-interface development with the Java framework AWT (Abstract Window Toolkit).

8. Week: Graphical user-interface development with the Java framework AWT (continued).

9. Week: 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.