**Graduate Course Action Form**

**Department/Program:** Electrical and Computer Engineering/Computer Science

**Course Prefix/Number:** CSC/ECE 517

**Previous Prefix/Number:****

**Date of Last Action:** April 9, 1992

**Course Title:** Object-Oriented Languages and Systems

**Abbreviated Title:** OBJ_ORIEN_LANG_SYS

**Scheduling:**
- Fall [x]
- Spring [ ]
- Summer [ ]
- Every Year [x]
- Alt. Year Odd [ ]
- Alt. Year Even [ ]
- Other [ ]

**Credit Hours:** 3

**Offered by Distance Education Only:** [ ]

**Contact Hours:**
- Lecture/Recitation: 3
- Seminar: [ ]
- Laboratory: [ ]
- Problem: [ ]
- Studio: [ ]
- Independent Study/Research: [ ]
- Internship/Practicum/Field Work: [ ]

**Grading:**
- ABCDF [x]
- S/U [ ]

**Instructor (Name/Rank):** Edward F. Gehringer, Associate Professor

**Graduate Faculty Status:**
- Associate [ ]
- Full [x]

**Anticipated Enrollment:**
- Per semester: 60
- Max. per section: 80
- Multiple sections: Yes [ ]
- No [x]

**Prerequisite(s):** CSC 316, Data Structures for Computer Scientists or equivalent.

**Corequisite(s):**

**Pre/Corequisite For:**

**Restrictive Statement:**

**Curricula/Minors:**
- Required: Computer Science, Computer Engineering, Electrical Engineering
- Qualified Elective: [ ]

**Proposed Effective Date:** August 16, 2004

**Approved Effective Date:**

**Catalog Description:** Object-oriented languages and systems built with object-oriented software components. Object-oriented design methodologies, such as CRC cards and the Unified Modeling Language (UML). Requirement analysis. Design patterns. Agile methods. Object-oriented programming environments, such as the Eclipse platform. Platforms for Web services, such as J2EE. Project required.

**Recommended By:**

Department Head/Director of Graduate Programs

**Endorsed By:**

Chair, College Graduate Studies Committee

**Approved By:**

Dean of the Graduate School
Course Justification

Object-oriented software is now in the mainstream of the software industry. Students get a good deal of object-oriented programming in their undergraduate curriculum, but design issues are not covered in detail, although they are at least as important as programming skill. Agile methods and Web services are growing in importance to students in their careers. Knowledge of this field is a great advantage to the practitioner working in prototype development or large software systems.

Proposed Revisions, with Justifications

The existing course description reads:

Object-oriented languages and the systems that can be built with object-oriented software components. Systems concepts, including inheritance, dynamic binding, type systems, and genericity; implementation in specific languages, e.g., Smalltalk, C++, and Eiffel. Object-oriented programming environments and object-oriented databases. Project required.

The new description reads:

Object-oriented languages and systems built with object-oriented software components. Object-oriented design methodologies, such as CRC cards and the Unified Modeling Language (UML). Requirement analysis. Design patterns. Agile methods. Object-oriented programming environments, such as the Eclipse platform. Platforms for Web services, such as J2EE. Project required.

This revision is needed because the technology has changed over the past twelve years. Many of the topics once covered in this course (e.g., inheritance and dynamic binding) are now covered in undergraduate curricula. As software systems have grown larger, object-oriented analysis and design have become much more important. Web services is an important topic that did not even exist 11 years ago. Now it is important for students to have experience in writing components for systems that operate over the Web.

Student Learning Objectives

Upon completion of this course, students will be able to—

• analyze the requirements for a large software system in terms of the classes and objects required,
• design a system by defining the interaction between these classes and objects,
• use design patterns to fit the system together in well understood ways,
• use an advanced programming environment to implement such a system,
• write a component-based application in an object-oriented language to deliver services over the world-wide Web.

Enrollment for Last Five Years

<table>
<thead>
<tr>
<th>Semester</th>
<th>CSC 517</th>
<th>ECE 517</th>
<th>Combined</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Section Total</td>
<td>Students Total</td>
<td>Credit Hrs Total</td>
</tr>
<tr>
<td>SPRG '99</td>
<td>2</td>
<td>79</td>
<td>225</td>
</tr>
<tr>
<td>FALL '99</td>
<td>2</td>
<td>91</td>
<td>258</td>
</tr>
<tr>
<td>FALL '00</td>
<td>2</td>
<td>77</td>
<td>228</td>
</tr>
<tr>
<td>FALL '01</td>
<td>3</td>
<td>88</td>
<td>255</td>
</tr>
<tr>
<td>FALL '02</td>
<td>3</td>
<td>46</td>
<td>138</td>
</tr>
<tr>
<td>FALL '03</td>
<td>2</td>
<td>47</td>
<td>135</td>
</tr>
<tr>
<td>Total:</td>
<td>14</td>
<td>428</td>
<td>1239</td>
</tr>
</tbody>
</table>
New Resources Statement

No additional teaching or laboratory resources will be required by the changes described in this action.

Consultation with Other Departments

These changes affect the curricula of Computer Science and Electrical & Computer Engineering. They have been approved by the Graduate Curriculum committees of the two departments.
Syllabus (old)

North Carolina State University

CSC 591O/NTU SE-730P:
Object-Oriented Languages and Systems
Fall 1992

Preliminary Information

Instructor: Edward F. Gehringer (Office: Daniels 206, (919) 515-2066. Office Hours: MW 2:00–3:00, TTh 1:00–2:00, E-mail: efg@ncsu.edu, Fax: (919) 515-7382. Please do not fax homework unless it is late).

Class Meetings: On campus, class meets 4:05–5:20 Tuesday and Thursday. The first class will be Thursday, August 20. Class will not meet on October 13 (Fall Break) or November 26 (Thanksgiving). The last class meeting will be on Thursday, December 3. The last day to drop this course without a grade is Friday, October 25. In general, off-campus students should view the tape of each lecture one week after on-campus students.


Teaching Assistant: E. Craig Hyatt (tentative), echyatt@eos.ncsu.edu.

Important note: It is a requirement that I have each student’s e-mail address. Please e-mail yours to me. IBM employees should become registered gateway users. Communication is much more efficient if everyone is reachable by e-mail.

Course Goals

The goals of this course are (i) to teach the student to decompose programming problems in an object-oriented framework; (ii) to present the concepts of inheritance, dynamic binding, genericity, and conformance in the context of three object-oriented languages, (iii) to introduce object-oriented programming environments, such as Smalltalk browsers, inspectors, and other debugging tools and documentation aids (e.g., those of Eiffel), (iv) to consider implementation issues in object-oriented systems, e.g., compiling efficient code from dynamically bound programs, and memory management of a large object space, and (v) to introduce object-oriented databases and explain how they differ from more traditional models.

Course Policy

Homework: There will be one or two programming assignments to be in each of the three languages. In addition, each student will choose a semester project from a list supplied by the instructors, or another similar topic suggested by the student with approval of the instructors. The semester projects will be due on December 4.

Homework Policy: Homework provides students with an opportunity to exercise and demonstrate mastery of newly gained knowledge and the ability to communicate the results of that mastery. Therefore, in the evaluation of homework assignments we will be concerned not only with the
correctness of results, but also with the style in which they are presented. Students should strive to present their results in a clear, concise, and understandable fashion. Programs, in particular, should be elegantly presented. Points may be deducted for sloppy, ungrammatical, or misspelled homework.

**Exams:** Two tests and a final exam will be given. The first test will occur in early-to-mid October, and the second test will occur in mid-November. The final exam will be given on Tuesday, December 8 at 1:00.

**Grading Policy:** The final course grade will be computed as follows:

- 20% - Programming assignments
- 20% - Project
- 30% - Tests
- 30% - Final Exam

Homework should be postmarked by the specified day. Late homework will be penalized at the rate of 10% per weekday, excluding holidays. (The penalty is computed as 10% of the point value of the homework, not 10% of the student’s score.) **Off-campus students, please be aware that late homeworks greatly increase the grading workload (it takes at least twice as long to grade homeworks and tests one-by-one). Submit your homeworks on time, and see to it that your site coordinator mails them promptly.**

**Collaboration:** Students should do all assignments independently unless specifically told otherwise. The University policy on academic misconduct will be enforced rigorously by the staff of this course. Guidelines on what constitutes plagiarism and cheating will be passed out on the first day of class. It is the responsibility of all students to be familiar with these guidelines.

**Syllabus:**

**General concepts**

<table>
<thead>
<tr>
<th>Concepts illustrated with Smalltalk</th>
<th># of 75-minute lectures</th>
</tr>
</thead>
<tbody>
<tr>
<td>Object-oriented design methodology</td>
<td>1</td>
</tr>
<tr>
<td>Introduction to the Smalltalk language</td>
<td>4</td>
</tr>
<tr>
<td>Programming environments: browsers, inspectors, notifiers</td>
<td>1</td>
</tr>
<tr>
<td>Structuring an application with the model/view/controller paradigm</td>
<td>1</td>
</tr>
<tr>
<td>Building a Smalltalk application</td>
<td>2</td>
</tr>
</tbody>
</table>

**Concepts illustrated with C++**

<table>
<thead>
<tr>
<th>C++: a typed object-oriented language</th>
<th># of 75-minute lectures</th>
</tr>
</thead>
<tbody>
<tr>
<td>Static and dynamic binding</td>
<td>4</td>
</tr>
<tr>
<td>Building libraries for reusability</td>
<td>1</td>
</tr>
</tbody>
</table>

**Concepts illustrated primarily with Eiffel**

<table>
<thead>
<tr>
<th>Programming with assertions</th>
<th># of 75-minute lectures</th>
</tr>
</thead>
<tbody>
<tr>
<td>Genericity, inheritance, and delegation</td>
<td>2</td>
</tr>
</tbody>
</table>

**Other linguistic concepts**

<table>
<thead>
<tr>
<th>Models of inheritance and type systems</th>
<th># of 75-minute lectures</th>
</tr>
</thead>
<tbody>
<tr>
<td>Conformance of types</td>
<td>1</td>
</tr>
</tbody>
</table>

**Implementation issues**
An interpreter for an o-o language, and its memory model 1
Memory management (ref. counting, g.-c., generation scavenging) 2

Object-oriented databases
The relational model vs. the o-o model 1
Persistence of objects 1
Concurrency, recovery, and distribution 1
Implementation issues 1
CSC/ECE 517: Object-Oriented Languages and Systems

Instructor: Edward F. Gehringer, 2301 Partners I, 515-2066, efg@ncsu.edu.

Prerequisites: CSC 316, Data Structures for Computer Scientists, or equivalent. In general, an undergraduate degree in computer science is sufficient for those coming from other universities.

Student Learning Outcomes: Upon completion of this course, students will be able to—

• analyze the requirements for a large software system in terms of the classes and objects required,
• design a system by defining the interaction between these classes and objects,
• use design patterns to fit the system together in well understood ways,
• use an advanced programming environment to implement such a system,
• write a component-based application in an object-oriented language to deliver services over the world-wide Web.

Required Textbooks:


Course Organization and Scope:

Introduction to Smalltalk and O-O Design 1 week
The Smalltalk language and environment 2 weeks
Uniform Modeling Language 1 week
Analysis, design, and implementation of an application 2 weeks
Design patterns 1 week
Java user-interface development 1 week
Web services in J2EE 2 weeks
Agile methodologies and Extreme Programming 2 weeks
Concurrency in Java 1 week
Web security and Java 1 week

Homework, quizzes, tests: Three programming assignments 20% of grade
Semester project 20% of grade
Two midterm exams 30% of grade
Final exam 30% of grade
Incomplete grades and late assignments: Incomplete grades will be given when a student cannot complete the course due to unforeseeable circumstances beyond his/her control. Homework will be accepted up to 4 days late (with weekends and class holidays not counting as days). A penalty of 10%/day will be assessed for lateness. If serious reasons prevent a student from completing an assignment on the scheduled date the opportunity to make-up will be offered according to the official academic policies, see http://www.ncsu.edu/policies/academic_affairs/pols_regs/REG205.00.4.php.

Academic integrity statement: Work in this course is to be done under the Academic Integrity Honor Pledge:

"I have neither given nor received unauthorized aid on this test or assignment."

Students must abide by the Code of Student Conduct, http://www.ncsu.edu/policies/student_services/student_discipline/POL11.35.1.php

Students are required to work independently on homework assignments unless otherwise specified by the instructor (e.g., on ungraded homework problems or team projects). The decision as to whether a student cheated depends on the intent of an assignment, the ground rules specified by the instructor, and the behavior of the student. Two guidelines help an instructor decide if cheating has occurred:

* Program plagiarism will be suspected if an assignment that calls for independent development and implementation of a program results in two or more solutions so similar that one can be converted to another by mechanical transformation.

* Cheating will be suspected if a student who was to complete an assignment independently cannot explain both the intricacies of his or her solution and the techniques used to generate that solution.

It is unreasonable to expect a complete definition of cheating; each case is important enough to be given careful, individual scrutiny. It is, however, helpful to have guidelines and precedents. Here are some examples of cases which are clearly cheating and clearly not cheating.

Cheating:

* Turning in someone else's work as your own (with or without his or her knowledge). Turning in a completely duplicated assignment is a flagrant offense.

* Allowing someone else to turn in your work as his or her own.

* Several people writing one program and turning in multiple copies, all represented (implicitly or explicitly) as individual work.

* Using any part of someone else's work without the proper acknowledgment.

* Stealing an examination or a solution from the instructor. This is an extremely flagrant offense.

Not Cheating:

* Turning in work done alone or with the help of the course staff.

* Submission of one assignment for a group of students if group work is explicitly permitted (or required).

* Getting or giving help on how to do something on Unix or another operating system used in a course project.

* Getting or giving help on the meaning of diagnostic messages from the compiler.

* High-level discussion of course material for better understanding.
* Discussion of assignments to understand what is being asked for.

The instructor and course staff will not condone cheating. When cheating is suspected, instructors will take reasonable action to establish whether it has occurred. If it has, the instructor or the Office of Student Conduct will apply appropriate disciplinary policy. All violations and penalties will be reported to the Office of Student Conduct, in accordance with NCSU policy. Records are kept by the OSC for a period of two years after the violation; they do not result in a permanent notation on a student's record. However, the student always has the right to bring the case before the Office of Student Conduct, if, for example, (s)he feels this will result in a more sympathetic hearing. Failure to request that the case be brought before the Office of Student Conduct will be deemed a waiver of this right.

A list of possible disciplinary actions is given below:

**Actions within the course:**
* No credit, reduced credit, or negative credit for the assignment.
* Loss of a letter grade for the course.
* Makeup assignment over the same material; no credit.
* Forced drop in the course.

**Actions by the University:**
* Failure in the course.
* Suspension from the University for a designated period.
* Expulsion from the University.

The following policies apply to all cases of cheating and plagiarism:
* For a first offense, the penalty will always be more severe than the penalty for failing to turn in the assignment (or take the exam) in question.
* For either repeated offenses or a flagrant offense by any student, the instructor shall refer the incident directly to the Office of Student Conduct for action.

**Students with disabilities:** Reasonable accommodations will be made for students with verifiable disabilities. In order to take advantage of available accommodations, students must register with Disability Services for Students at 1900 Student Health Center, Campus Box 7509, 515-7653. For more information on NC State's policy on working with students with disabilities, please see this page (http://www.ncsu.edu/provost/hat/current/appendix/appen_k.html).