CS 680 Distributed Software Development
Spring 2010

Instructor: Giuseppe (Peppo) Valetto
E-mail: valetto@cs.drexel.edu
Office: University Crossings 106
Phone: 215-895-1518
Office Hours: TBA
Class Hours: TBA
Classroom: TBA

Introduction
The main goal of this course is to familiarize students with methods of software production that involve distributed collaborative work. These methods represent a dominant trend in contemporary software engineering, and present developers, technical leads, project managers, and scholars alike with a number of interesting challenges, in terms of how the development effort can be organized and conducted in effective ways. Examples of distributed software development abound, from global software development (GSD) efforts carried out by multi-national corporations and software houses, to joint ventures between multiple industrial entities (which at times take the form of outsourcing or off-shoring) to Open Source Software (OSS) communities getting together to build some new Free/Open Source software product, to consulting and service projects in which third-party developers join forces with resident work force for ad hoc collaborations. All together, projects of these kinds account for a very substantial and ever-growing portion of all software development efforts today.

We will look at this topic from multiple perspectives:

• the software development paradigms and processes that can be adopted for distributed software development, and how they may impact the work of developers and the outcome of the project;
• the organizational and collaboration models that can be observed in distributed software development, and how they relate at the same time to the product to be developed and the process that is adopted;
• the tools that are commonly used to support distributed development work, and their significance as enablers of long-distance collaboration and coordination among developers.

Objectives
• Provide students with an awareness of the distributed software development trend and related challenges
• Expose students to a variety of distributed development models, which are likely to be of great significance in any software-related career in the near future
• Analyze the consequences of distribution on collaboration and coordination in development teams
• Explore the latest Software Engineering research on the topic of Distributed Software Development, and engage students in significant amount of critical discussion on that research and its impact on the field.

Prerequisites
The course is open to graduate students in CS and SE, as well as (subject to instructor’s approval) senior undergraduate students with adequate GPA and background. There is no official pre-requisite for this class. Practically speaking, however, to do well in the course, the following must be true:

• You need to have a solid understanding of basic software engineering concepts and principles, similar to what is covered in Drexel CS 451
• You need to have some familiarity with mainstream tools for configuration management, such as CVS or SVN, and state-of-the-art programming and development environments, such as Eclipse, NetBeans, Sourceforge, Trac etc.
• You also need to have acquired substantial development experience, in one or more contemporary programming languages, such as Java, C#, C++, Python etc.

Notice that you may have acquired any or all of the above from a combination of courses, or even outside any academic course, for example on the job. Therefore, to discuss or verify your eligibility, or for any other inquiries about prerequisites above, don’t hesitate to contact the instructor.

Topics
Lectures are intended to be very interactive, class participation is required and strongly encouraged, and critical thinking and personal contribution to the in-class discussion are part of what is expected of students.

Major topics discussed in this course include the following:
• Challenges of distributed software development: how is it different from co-located development?
• The socio-technical nature of large-scale software development, according to Brooks, Conway and Parnas
• The promise and reality of globally distributed software development
• Software development life cycles for distributed projects
• The Open Source Software model and its eco-system
• Open Source Software dissected: how does it work inside?
• Software development organizations and distributed teams
• Tools and environment for collaborative distributed development
• Investigating distributed software projects by means of their software repositories

This list is tentative, and is subject to change, depending on class needs.

Textbook
There is no official textbook for this class.
We will have weekly readings lifted from research and best practice publications on the topics covered in each class, as indicated by the instructor. Those readings will create the opportunity for discussion and will be used to introduce the class topic and/or highlight specific aspects of interest.

**Class organization**

The course will be run in a highly interactive fashion. A significant portion of each class will be devoted to presentation and discussion of relevant subjects, driven by students and facilitated by the instructor. The instructor will also introduce new subjects and explain particularly important and/or complex aspects of the covered material.

**Course components**

1. **Reading assignments:**
   Students are supposed to carry out preparatory readings from the textbook and other sources indicated by the instructor as part of their assignments, before the class meets. Their reflections on the material will fuel the in-class discussion. Each week, and for each of the reading assignments, a student will be chosen as the Designated Presenter, and will assume the responsibility to prepare an in-class presentation about that particular reading item.

2. **Class participation:**
   Engaging discussion of the lecture and reading material is considered a fundamental component of this class. Participation in the discussion on reading assignments, specifically, occurs through two different channels: in-class discussion, as well as online discussion, through the class discussion board. With respect to online discussions, student contribution to the online threads will be graded on a 0-3 scale: 0 means no participation; 1 means marginal contribution; 2 means satisfactory contribution; 3 means exceptional contribution. Satisfactory participation requires not simply posting comments, but stimulating the discussion by proposing questions, answering questions and replying to other’s comments, pointing additional or related material, etc. The Designated Presenter is supposed to integrate the above contributions in her presentation.

3. **Group project:**
   A substantial part of the course work is a multi-week collaborative project that is intended to bring together many of the various aspects learned in class. The project will enable students to acquire in-depth familiarity with one or more facets of the material covered in class. Students will be asked to report the results of their project work and discuss them in class.

4. **Group position paper:**
   The same groups working on the project will compose a short technical paper, on a topic related to some of the more conceptual aspects of software design discussed in class. The group will present their position paper in class as part of their final requirements, aiming at stimulating exchange of ideas with the rest of the class and the instructor on the chosen topic.

5. **Take-home exams:**
The course will include at least one individual exam (either midterm, or final, or both). The exams will take the form of take-home questions and essays, which will require of the students some independent research of prominent literature in the field of distributed software development, and expanded exploration of issues covered in class.

6. **Quizzes and in-class exercises:**
   In-class practical and problem-solving exercises will be carried out from time to time, with the goal to highlight and explicate specific issues that have been addressed during that class. Some of those exercises and problems will have a programming connotation: bring your laptops! Other in-class activities may include quizzes. The work done during those in-class activities will be rewarded with credit: students who are absent or unprepared to carry out those activities will not receive credit.

**Class Policy**

- **Do’s and Don’t’s**
  - Be punctual
  - Switch off cell phones
  - No laptops for personal use

- **Adhere to standard academic honesty rules.**
  - See: [http://www.drexel.edu/provost/policies/academic_dishonesty.asp](http://www.drexel.edu/provost/policies/academic_dishonesty.asp)

- **Late policy**
  You will submit all assignments through the BB Vista WWW facilities. Submissions with a timestamp later than the due date and time stated in the assignment specifications are deemed late. They will be dealt with as follows:
    - Graded with a 20% penalty if submitted within 1 day from the official assignment deadline
    - Not graded if submitted later than 1 day

*Force majeure* exceptions (e.g. for documented illness) will be considered and an extension can be negotiated with the instructor for those cases. In those cases, it is the student’s responsibility to contact the instructor BEFORE the official assignment deadline, if at all possible.