

# CIS 4339 Project in Computer Science

---

<b>Course:</b>	CIS 4339
<b>Course Title:</b>	Project in Computer Science
<b>Time:</b>	
<b>Place:</b>	
<b>Instructor:</b>	Paul Wolfgang
<b>Instructor Phone:</b>	215-204-5155
<b>Office Hours:</b>	MWF 3:00 – 5:00
<b>Course Web Page:</b>	<a href="http://www.cis.temple.edu/~wolfgang">www.cis.temple.edu/~wolfgang</a>
<b>Prerequisites:</b>	C or better in: CIS 4298      Software Engineering and Senior Standing
<b>Textbooks:</b>	None assigned
<b>Course Description</b>	Team-oriented design and implementation of a large programming project. Students will propose topics for review and acceptance early in the semester. Students will provide written documentation of their completed projects and will demonstrate the operation of their completed projects in an oral presentation.
<b>Course Goals:</b>	Students will be organized into project teams ranging in size from 3 to 6. The project will follow a modified version of the Rational Unified Process known as the Unified Process for Education. <ul style="list-style-type: none"><li>• Inception – defines the scope of the project</li><li>• Elaboration – design product(s) and plans the project</li><li>• Construction – builds the product(s)</li></ul> The end of each phase is a Milestone with specified documentation and other artifacts.
<b>Grading:</b>	The artifacts at the end of phase will be graded. Students will have an opportunity to revise draft documentation prior to final grading. Grading will be based on technical content and quality of writing.
<b>Exam Dates:</b>	There are no exams. Final project presentations will be made in lieu of a final exam during the final exam week.
<b>Attendance Policy:</b>	Attendance is mandatory. Unexcused absence will result in reduction of final grade.

## Other Important Information

### Disability disclosure:

Any student who has a need for accommodation based on the impact of a disability should contact me privately to discuss the specific situation as soon as possible. Contact Disability Resources and Services at 215-204-1280 in 100 Ritter Annex to coordinate reasonable accommodations for students with documented disabilities. (Temple University Policy and Procedures Manual)

### **Academic freedom:**

Freedom to teach and freedom to learn are inseparable facets of academic freedom. The University has a policy on Student and Faculty and Academic Rights and Responsibilities (Policy #03.70.02) which can be accessed through the following link: [http://policies.temple.edu/getdoc.asp?policy\\_no=03.70.02](http://policies.temple.edu/getdoc.asp?policy_no=03.70.02).

### **Academic Honesty**

Academic cheating (such as plagiarism, copying during an exam, copying homework, stealing files and passwords, etc.) is strictly prohibited in this course. The penalty for the first offense will normally be an F in the course. A subsequent offense (in this or any other course) may also be referred to the University Disciplinary Committee.

No collusion what-so-ever during an exam will be tolerated. In particular not talking or other sharing of information (for example during open book exams) is permitted. Keep your eyes on YOUR paper.

IGNORANCE OF ACCEPTABLE GUIDELINES OF CONDUCT IS NO EXCUSE

[http://policies.temple.edu/getdoc.asp?policy\\_no=03.70.12](http://policies.temple.edu/getdoc.asp?policy_no=03.70.12)

### **Dates to Remember**

First Day of Class:	August 31, 2009
Last Day to Drop:	September 14, 2009
Last Day to Withdraw* :	November 2, 2009
Thanksgiving Recess:	November 26-29, 2009
Last Day of Class:	December 9, 2009
Final Project Presentation**:	To Be Announced
Final submittal of all deliverables:	December 14, 2009

\* Students may withdraw from a course only once. Students may withdraw from a total of five courses.

\*\* The Final Project Presentation will be either during the last week of class or during the exam period.

## Documentation Artifacts

### Documentation by Phase

The documentation is expected to be developed over the life of the project, but certain documents are required to be initially delivered (I), finalized (F), or revised/updated (R) at the end of particular phases as shown in the following table:

Document	Phase		
	Inception	Elaboration	Construction
Project Abstract	F		
Software Development Plan	I	F	
Requirements Specification	I	F	R
Users Manual		I	F
Design Document (Part I)		F	R*
Design Document (Part II)		I	F
Test Procedures		I	F
Test Report			F

\* Included with the Design Document (Part II)

## General Requirements

With the exception of the Project Abstract each document will contain:

Title Page: showing the name of the document, authors, release date, and revision number.

Table of Contents

System overview (copy/adapt the Project Abstract)

Document overview: summarize the purpose and contents of the document

Body: the content of this document

References to other documents

Glossary if required

Index

Each page will include a page number, document title, release date, and revision number in the header or footer.

## Project Abstract

### Purpose

The Project Abstract provides an initial description of the project's goals. It offers:

- a flavor of what the application will accomplish
- why the user should use this application
- why this application and approach are superior to the competition's efforts

### Requirements

The Project Abstract will be at least ½ page, but not more than one page long. A page is defined as an 8½" × 11" with 1" margins containing 11point single-spaced text. The Project Abstract will contain.

- A top-level description of the requirements.
- A conceptual design.
- Reference to similar products.

The abstract will **not** contain phrases such as:

- We are a team from Temple University ...
- This application fulfills a class requirement ...
- I am doing this because ...
- Our team has decided to ...

Examples can be found on software websites, advertising brochures, and in the introduction section of many manuals. Here is a short example:

*TestItAll sets a new standard for automating software testing and development. A series of user defined menus will enable testers to create easily regression suites without the need to learn a proprietary programming language. And on, and on, and on, for at least half a page of text.*

## Software Development Plan

### Purpose

The Software Development Plan describes the activities and tasks to be performed to develop the software product.

### Requirements

In addition to the general requirements the Software Development Plan will contain the following sections:

Activities	E.G. requirements gathering, top-level design, detailed design, test.
Tasks.	<p>A task is the performance of an activity leading to a specific product. E.G. Design of unit x. Associated with each task is</p> <ul style="list-style-type: none"><li>• predecessor tasks (what tasks must be complete before this task can start)</li><li>• an estimated effort</li><li>• estimated start date</li><li>• estimated finish data</li><li>• responsible individual</li><li>• successor tasks (what tasks cannot start until this task is complete)</li></ul>
Schedule	A graphical lay-out of the tasks in the form of a Gantt Chart.

## Requirements Specification

### Purpose

The Requirements Specification defines the functional and non-functional requirements for the product.

### Requirements

In addition to the general requirements the Requirements Specification will include the following:

Use-case diagram or some other diagram that identifies the external interfaces.

Use-case descriptions:

For each use-case define the triggering event and the interactions between the actor and the system. Normal and alternate flows should be described. A State Diagram may be useful in some cases.

Non-functional requirements:

Any constraints on the system. E.G. minimum/maximum response time.  
Minimum/maximum memory, etc.

# Users Manual

## Purpose

The Users Manual describes how to use the system.

## Requirements

In addition to the general documentation requirements the Users Manual will contain

Quick Start Guide	For the experienced user.
Installation	Basic installation, network considerations, uninstalls. Installation should be automated and seamless. The system must be responsible for determining if minimum requirements are satisfied.
Configuration	Single user, multi-user, network, resources. Automation is again the key to successful configuration.
Security	Passwords.
Database	Installation and maintenance if required.
Application Functions	System functionality, screen shots. Step-by-step operating procedures.
Backup and Recovery	
Error Messages	Messages and actions. Don't leave people hanging. When an error is reported, there must be a corrective action.
Troubleshooting	Troubleshoot the application -- not the operating system or network.
Support	Contacts, contracts

## Design Document (Part I)

### Purpose

The Design Document (Part I) describes the software architecture. This document will be a combination of diagrams and text that describes what the diagrams are showing.

### Requirements

In addition to the general requirements the Preliminary Design Document will contain:

A description the different components and their interfaces. For example: client, server, database.

For each component provide class diagrams showing the classes to be developed (or used) and their relationship.

Sequence diagrams showing the message flow for each use-case.

State diagrams for classes that have states.

If there is a database:

- Entity-relation diagram.

- Table design.

## Design Document (Part II)

### Purpose

The Design Document gives the complete design of the application.

### Requirements

In addition to the general documentation requirements the Design Document will contain

Updated copy of Part I

For each class define the data fields, methods.

- The purpose of the class.

- The purpose of each data field.

- The purpose of each method

  - Pre-conditions if any.

  - Post-conditions if any.

  - Parameters

  - Return value

  - Exceptions thrown\*.

This information should be in structured comments (e.g. Javadoc) in the source files. A documentation generation tool (e.g. Javadoc) should be used to generate the document.

\* At the top level, or where appropriate, all exceptions should be caught and a error message that is meaningful to the user generated. It is not OK to say ("xxxx has encountered a problem and will now close (OK?)" Error messages and recovery procedures should be documented in the Users Manual.

## Test Procedures

### Purpose

The Test Procedures describe the test approach and the tests to be performed.

### Requirements

In addition to the general documentation requirements this document will contain the procedures for the following tests:

#### Unit tests

For each method, one or more test cases.

A test case consists of input parameter values and expected results.

All external classes should be stubbed using mock objects.

#### Integration tests

Tests to demonstrate each use-case based on the use-case descriptions and the sequence diagrams. External input should be provided via mock objects and results verified via mock objects. Integration tests should not require manual entry of data nor require manual interpretation of results.

#### Acceptance test

Demonstration of all of the functional and non-functional requirements. Actual manual usage of the system and observation of results is preferred.

## **Test Report**

### **Purpose**

The Test Report is a record that the tests were run and a documentation of their results.

### **Requirements**

In addition to the general documentation requirements this document will contain:

Output from the unit test runs.

Output from the integration test runs.

A copy of the Acceptance Test procedure with notations indicating that the test was performed and the observed results.

List of known problems: describe each failed test.