



Software Quality Management: CIS 8300 Syllabus

Table of Contents

Catalog Description.....	1
Prerequisites.....	2
Overview.....	2
Learning Objectives.....	2
Intended Audience.....	3
Course Materials.....	3
Evaluation.....	6
In-Class Exercises.....	7
Apply a quality model.....	7
Define & apply checklists.....	7
Customize RUP.....	7
Debate.....	8
Case analysis.....	9
Homework.....	9
Define a software quality process model.....	9
Project quality assessment.....	10
Process assurance plan.....	10
Special topic.....	10
Examinations.....	11
Exam 1.....	11
Exam 2.....	11
How to Scan CIS Literature.....	11
Software.....	11
Literature Review.....	12
Workload Expectations.....	12
Student Behavior.....	13
Discrimination and Harassment.....	13
Official CIS Department Class Policies.....	13

As with any document, be aware that this may contain clerical errors. Please tell me if you spot one.

The instructor reserves the right to modify the syllabus as necessary to improve student learning and provide appropriate evaluation. Students will be notified of any such modification in-class and via the web site.

Catalog Description

The current university catalog description of this course can be obtained in the University's Catalog:



http://www.gsu.edu/es/catalogs_courses.html

A recent university catalog description follows:

In the past, software quality initiatives often focused on software testing. Now, software quality improvements derive from the design of software development processes—a quality process creates quality products. The course will cover methods and tools for achieving software quality assurance at various levels of a software system including at the module, subsystem, and system levels. The principles of software development and management are presented with special emphasis on the processes and activities of quality assurance. State of the art tools and techniques including development process modeling, manual and computer-assisted reviews, and ROI analysis of new processes. In addition, the role of standards, policies, and procedures are discussed, with examples drawn from IEEE, ISO, CMMI, RUP, and other process models and standards. The course will prepare students to methodically develop a software quality-assurance program. This course provides practical knowledge of a variety of quality assurance techniques, and an understanding of some of the tradeoffs between techniques.

Prerequisites

Required: [CSP](#) 1-8, CIS 8030

Overview

This class covers the principles of software development emphasizing processes and activities of quality assurance.

Learning Objectives

Upon successful completion of this course, you will accomplish the following objectives and outcomes. In particular, students who complete this course will gain "Ready for work" skills, including:

1. Defining quality assurance plans
2. Applying quality assurance tools & techniques

Additionally, much of the software is available for download, either from the instructor, or from the CIS agreements with [MSDNAA](#) and the [IBM Academic Initiative](#).

1. Understand the quality assurance context
 - a. Describe the challenges of software quality [Galín, Ch. 1]
 - i. IS quality: (Gibbs 1994) (Whittaker et al. 2002)
 - b. Define software quality [Galín, Ch. 2]
 - c. Define a software quality model [Galín, Ch. 3]
2. Understand SQA projects
 - a. Describe components of a software quality assurance system [Galín, Ch. 4]
 - b. Describe software quality plans [Galín, Ch. (scan 5),6]
 - c. Relate software quality to the software development life-cycle [Galín, Ch. 7,8]
 - d. Describe quality tools & techniques
 - i. **Demonstrated by** Apply a quality model, Define & apply checklists, Customize RUP
 - ii. Testing [Galín, Ch. 9,10]
 - iii. Quality procedures [Galín, Ch. 14,15]
 - iv. Runtime assurances, requirements monitoring(Fickas *et al.* 2005)
 - v. Open source(Cowan 2003)



- e. Describe a common software process model, and tailor it for increased quality.
 - i. RUP (Ericsson 2000); Tailoring RUP with IBM Rational Method Composer (Haumer 2005; Haumer 2006)
 - ii. **Demonstrated by** Customize RUP
3. Understand SQA management
 - a. Describe process controls [Galin, Ch. 20]
 - i. Agile, MSF, & CMMI (Anderson 2005)
 - b. Describe quality metrics [Galin, Ch. 21]
 - c. Describe quality costs [Galin, Ch. 22]
 - i. SQA economics: iDAVE (Boehm *et al.* 2004)
4. Understand SQA standards
 - a. Describe common standards [Galin, Ch. 23]
 - b. Describe IEEE standards [Galin, Ch. 24]
 - c. **Demonstrated by** Define a software quality process model
5. Understand SQA management organization
 - a. Describe management roles, organization, and activities [Galin, Ch. 25,26]
6. Know software quality management, comprehensively
 - a. Be able to analyze current issues in software quality assurance
 - i. Debate the issues
 - b. **Demonstrated by** Exam 1, Exam 2,
7. Know a software-quality management topic, in detail
 - a. **Demonstrated by** Topic article, Topic presentation
8. Demonstrate critical thinking, integrative reasoning, & communication skills
 - a. **Demonstrated by** all coursework, but especially Debate

Intended Audience

Anyone with a keen interest in software quality will do well in this course. It's mainly geared to produced IS Project Managers, Systems Analysts and Designers. However, Programmers, Database Administrators, and CIO's may benefit from this course.

Course Materials

The primary textbook, Galin, must be purchased. Other books and articles are available on-line, for free.

1. **Primary Textbook:** [Software Quality Assurance: From Theory to Implementation](#), Daniel Galin, Addison Wesley, 2003.
2. **Reading List:** On-line articles—see the Learning Objectives and the following References.



References

- Anderson, D.J. "Stretching agile to fit CMMI level 3 - the story of creating MSF for CMMI/spl reg/ process improvement at Microsoft corporation," 2005, pp. 193-201.
- Boehm, B., Huang, L., Jain, A., and Madachy, R. "The ROI of software dependability: The iDAVE model," *Software, IEEE* (21:3) 2004, pp 54-61.



- Cowan, C. "Software security for open-source systems," *Security & Privacy Magazine, IEEE* (1:1) 2003, pp 38-45.
- Ericsson, M. "Developing Large-Scale Systems with the Rational Unified Process," IBM Rational.
- Fickas, S., Robinson, W., and Sohlberg, M. "The Role of Deferred Requirements: A Case Study," International Conference on Requirements Engineering (RE'05), IEEE, Paris, France, 2005.
- Gibbs, W.W. "Software's chronic crisis," *Scientific American* (271:3) 1994, pp 72-81.
- Haumer, P. "IBM Rational Method Composer: Part 1: Key concepts," *The Rational Edge*:12), 15 December 2005.
- Haumer, P. "IBM Rational Method Composer: Part 2: Authoring method content and processes," *The Rational Edge*:1), 15 January 2006.
- Whittaker, J.A., and Voas, J.M. "50 years of software: key principles for quality," *IT Professional* (4:6) 2002, pp 28-35.

E-book from Books24x7

Consider the E-books as another resource; they are **free** to our students. See this note: <http://www2.cis.gsu.edu/cis/news/newandnoteworthy2.asp> Access from the GSU online library: <http://www.library.gsu.edu/ebooks/>; select the link **Books24x7**.

The following E-book from Books24x7 books may support your interests in the course materials. These are FYI only (For Your Information) and are **not required in class**, unless indicated by the readings.

- [Data Quality: The Accuracy Dimension by Jack E. Olsen Morgan Kaufmann Publishers © 2003](#)
- [Implementing the Capability Maturity Model by James R. Persse John Wiley & Sons © 2001](#)
- [Improving Data Warehouse and Business Information Quality: Methods for Reducing Costs and Increasing Profits by Larry P. English John Wiley & Sons © 1999](#)
- [Interpreting the CMMI: A Process Improvement Approach by Margaret K. Kulpa and Kent A. Johnson Auerbach Publications © 2003](#)
- [Measuring Information Systems Delivery Quality by Evan W. Duggan and Han Reichgelt \(eds\) Idea Group Publishing © 2006](#)
- [Practical Guide to Software Quality Management, Second Edition by John W. Horch Artech House © 2003](#)
- [Practical Insight into CMMI by Tim Kasse Artech House © 2004](#)
- [Software Process Improvement with CMM by Joseph Raynus Artech House © 1999](#)
- [Testing and Quality Assurance for Component-Based Software by Jerry Zeyu Gao, H.-S. Jacob Tsao and Ye Wu Artech House © 2003](#)

Schedule

The following table defines the schedule. However, the topics and readings may change according to the interests and abilities of the class. See the [Academic Calendar](#). *On the web, the underlined items link to supporting information.* Materials may be updated 24 hours prior to class; please check before attending class.

The following table refers to the preceding Learning Objectives. In particular, the Outcomes column refers to specific Learning Objectives, which include references to the course readings. You are responsible for reviewing the materials referenced in the Outcomes column prior to class.



Week	Learning Objectives	Outcomes	Key Events
1.	Understand the quality assurance context Introduction, Bad software, SQM	1.a	
2.	Understand the quality assurance context Quality models	1.b,1.c	Apply a quality model
3.	Understand SQA projects SQA planning and plans Review: Example SQA Plans	2.a,2.b	
4.	Understand SQA projects SQA & SLDC Reviews Demo: RUP web	2.c	Define & apply checklists
5.	Understand SQA projects Demo: Method Composer	2.e.i, 2.d.iii	Customize RUP Debate quality assurance process: distinct and separate vs. integrated within development process
6.	Understand SQA management Agile, MSF, & CMMI Exam 1 review	3.a,3.a.i	Define a software quality process model Example Debate Topics Debate development process for quality: comprehensive vs. agile process models
7.	Know software quality management, comprehensively		Exam 1
8.	Understand SQA management Metrics Demo: Quality measurement tools	3.b	
9.	Understand SQA management SQA Economics Demo: iDAVE	3.c,3.c.i	Debate quality improvement activity: requirements analysis vs. product testing
10.	Understand SQA projects Testing Demo: Quality testing tools	2.d.ii	
11.	Know software quality management, comprehensively Case study of plans and open-source projects	2.d.v	Case analysis Debate development model for quality: open-source vs. proprietary software quality
12.	Know a software-quality management topic, in detail		Topic presentation
13.	Know a software-quality management topic, in detail		Project quality assessment Topic presentation
14.	Holiday (no classes)		
15.	Understand SQA projects Runtime assurances Demo: runtime monitoring Exam 2 review	2.d.iv	
16.	Know software quality management, comprehensively		Exam 2
17.	Finals Week		Process assurance plan Topic article



Evaluation

Students are evaluated by the deliverables summarized in Table 1 and described in the In-Class Exercises, Homework and Examinations sections. The course credits are earned according to the following Table 1.

Table 1 Relative weights assigned to course deliverables.

Assignment	Percentage
Exam 1	25
Exam 2	25
Define a software quality process model (<i>group project</i>)	10
Project quality assessment (<i>group project</i>)	10
Topic presentation (<i>group project</i>)	5
Topic article (<i>group project</i>)	10
Process assurance plan (<i>group project</i>)	10
In-Class Exercises	5
Total	100

The following table overviews how credit will be assigned. Note that all group work includes a peer review, which can distinguish an individual's assigned points from the group's assigned points. (See **Self-Managed Teams** in the Workload Expectations section.)

Table 2 Grading standards.

Work quality	Percent
Absolutely fantastic, walk on water, overflow grade	110
Excellent answer on all counts	100
Excellent answer on most counts	90
Very good answer, but not excellent	80
Professionally done and adequate	70
Inadequate, needs work	60
Varying degrees of inadequacy	0 - 50

The following breakout depicts how grades will be assigned under this system.



Grade	Percentage
A	≥ 90
A-	≥ 87
B+	≥ 83
B	≥ 80
B-	≥ 77
C+	≥ 73
C	≥ 70
C-	≥ 67
D	≥ 60
F	< 60

In-Class Exercises

Apply a quality model

Given a UML specification (sample). You are to:

- Use the factors from a quality model to define testable quality requirements.

Define & apply checklists

Given a UML specification (sample). You are to:

- Conduct an inspection using checklists (see the RUP web site).
 - **Deliver answers to the following questions on paper or via email:**
 - What is the value of the inspection? What is the value of the checklists? Would you make improvements? (If so, specify.)
 - If you deem it helpful, extend the checklists with your own rules.

Customize RUP

- Select either RUP for Small Projects or RUP for Large Projects from the RUP Web.
- **Deliver answers to the following questions on paper or via email:**
 - Define a customization of the work breakdown structure of the delivery processes.
 - Define your own goals for a new RUP variant (e.g., Agile RUP, BigBoxStore RUP, etc.)
 - Given the limited time, focus on quality tasks
 - Which tasks would you add, remove, reorder? Why?
 - Delivery your itemized list of task changes, along with rationale
- You may use IBM Rational Method Composer (the best tool) or some other tool to specify the work breakdown structure (e.g., MS Project)



Debate

We use the debate format to provide a structured, thoughtful discussion of the issues. It is intended to provide an informal and enjoyable learning experience. Please, have fun with it. Feel free to show a sense of humor. However, ensure that the important points are discussed.

- Students will be partitioned into two groups, randomly selected
- Each group will be designated as the proponent for a particular perspective
- The week prior to the debate, each group should prepare a debate plan:
 - Review any relevant materials. The prior class reading should prepare you. However, you may want to do a brief literature scan. (See How to Scan CIS Literature.) It's up to you to determine how much material you need. Don't spend too much time researching the topic, unless that's your interest. In the end, the goal is to learn, have fun, and perhaps win the debate.
 - Discuss the issues within the group (via email, phone, discussion board, *etc.*)
 - Define the main discussion points
 - Prepare a few slides; about 5, no more than 10
 - Designate a lead speaker, or speakers
 - Plan on 10 minutes of opening remarks, by each group followed by nearly an hour of discussion
- The in-class schedule for Debate Day is typically:
 - Other course work (e.g., instructor presentation)
 - Debate 1 hour
- **Deliver** your slides and a talking points summary (if you have one) via a single email to the instructor

Example Debate Topics

Debate development process for quality: comprehensive vs. agile process models

Structure in development models ranges from the simple "hack and build" to the complex, lock-step of the waterfall, cf., [Choose the Right Software Method for the Job](#). Currently, developers are divided as to how much structure is appropriate, and how it effects quality. Many suggest that the structure of RUP, CMMI, ISO 9000 is good, while other react to those "overly complex approaches" with simpler structures of Agile or eXtreme.

- Which process model produces better quality software, the comprehensive models, such as RUP or CMMI, or the smaller, agile models, such as Agile or eXtreme?
 - Does it depend on project characteristics, such as size, type, domain, *etc.*?

Debate development model for quality: open-source vs. proprietary software quality

Proponents of open-source software have suggested that open-source software produces better software than commercial, proprietary development. In his essay on the open-source movement, "[The Cathedral and the Bazaar](#)," developer Eric Raymond wrote, "Given enough eyeballs, all bugs are shallow." (cf., [Building trust into open source CNET News.com](#)) Commercial software's quality problems are evidence. In response, Microsoft has redirected its efforts to improving quality (cf., [One year on, is Microsoft 'trustworthy'?](#), [The Trustworthy Computing Security Development Lifecycle](#)) Researchers have attempted to shed light on this controversy, e.g., [Is Open Source Software Development Faster, Better, and Cheaper than Software Engineering?](#), [High Quality and Open Source Software Practices](#). Why do you think?



- Which software development environment produces better quality software, the open-source model or the commercial, proprietary model?
 - Does it depend on project characteristics, such as size, type, domain, *etc.*?

Debate quality assurance process: distinct and separate vs. integrated within development process

Some people suggest that software quality should be a distinct and separate process from the software development process (e.g., ch. 9 [Practical Insight into CMMI by Tim Kasse Artech House © 2004](#)). Yet, RUP does not include such a separate quality process. Few process models separate the quality management functions. There have been some attempts to “improve” RUP, e.g., [Why isn't there a RUP workflow for software quality assurance?](#) by Karen Ulferts.

- Which is the best approach, quality as an inherent part of development or as a distinguished, managing process?
 - Does it depend on project characteristics, such as size, type, domain, *etc.*?

Debate quality improvement activity: requirements analysis vs. product testing

Read through any document on software quality, and you'll find that much of the text is devoted to code testing strategies. Yet, it is universally acknowledged that fixing requirements errors has a higher ROI than the downstream techniques. For example, if we could automatically derive programs from requirements (e.g., [Deriving Operational Software Specifications from System Goals](#)), and we had some assurance that the requirements were correct (e.g., [Reasoning about Partial Goal Satisfaction for Requirements and Design Engineering](#)), then there would be no need to test the code, right? [Note, you may simply want to scan to see what's in these very formal [KAOS papers](#). The details require formal training to understand.] Why is there no software quality book dedicated entirely to software requirements? The lack of such books suggest that discovered problems are best fixed in the requirements, but that it is too difficult to discover requirements problems?

- How should the development resources (time, people) be allocated over the life-cycle (requirements, design, coding, testing) for the maximum benefit of detecting and removing software defects? Should more effort be focused upstream on requirements analysis or downstream, on code testing?
 - Does it depend on project characteristics, such as size, type, domain, *etc.*?
 - Does the software artifact type (e.g., structure, formalism) and available methods and tools play a role?

Case analysis

Download the in-class case study document from our web site.

Given a software development problem description, you are to:

- **Deliver** answers to the case questions (distributed by the instructor)

Homework

Define a software quality process model

- **Deliver** the a single zip file via email containing:
 - Your Exported IBM Rational Method Composer Plugin
 - Use IBM Rational Method Composer to define a process plug-in. For you process, either:
 - Define (and/or improve) a process model based on your workplace development



- Define an improved process model for CIS student course work
- Ensure that the model includes a significant number of quality related activities.
- A 1-2 page executive summary document that justifies your process model, with special attention clarifying the ROI of the quality related activities.

Project quality assessment

- **Deliver** your Project Quality Assessment as an email attachment
 - The plan must be defined according to this PPQA assessment template (PPQA assessment.doc and PPQA assessment.xls).

Process assurance plan

- **Deliver** your Process Quality Assurance plan as an email attachment
 - Your updated IBM Rational Method Composer Plugin
 - Elaborate your Define a software quality process model with a Process Quality Assurance plan
 - A Process Quality Assurance plan for your software quality process model
 - Download the software quality process model document template (CIS.GSU-SQA_plan_template.doc) from our web site. Edit the document so that it describes your Process Quality Assurance plan
 - A 1-2 page executive summary document that justifies your process model, with special attention clarifying the ROI of the quality related activities.

Special topic

The student will describe a Quality Management *method* or *tool*

- a 5 page (single spaced) paper (figures & tables not included in page count)
 - Liberal use of quotations are allowed; however, the **sources must be referenced**. No more than 20 percent of the paper word count can be quotations
- a PowerPoint presentation summary (5 – 15 slides)

Select a topic

Example topics include any of the following, as well as **any other topic approached by the instructor**:

- Walkthroughs, Inspections, Reviews, Audits, Checklists, Interviews, Workshops
- Traceability analysis
- Quality Function Deployment
- Failure Mode and Effects Analysis
- Risk management (e.g., DDP)
- Requirements monitoring
- Model Checking (e.g., Spin from ATT) and Model Analysis (e.g., Alloy @ MIT)
- Goal-oriented requirements analysis: KAOS, i*
- ISO 9000 Quality Assurance
- Cost-Value approaches for prioritizing requirements, testing, *etc.*
- Use-case and test-case alignment



- Six Sigma methods (e.g., 5 Whys, Fishbone diagrams) and software
- Tools: Rational Method Composer, Requisite Pro, Doors, Holosofx (IBM WebSphere Business Modeler), Rational Robot (and other testing tools), *etc.*
- Eclipse Quality Assurance PlugIns, especially Jupiter (Document Review), Coverlipse, and the many metric calculation and visualization tools.
- Process simulation (AnyLogic, Holosofx, *etc.*)

Topic article

Your class peers are the intended audience for this article. They are among the best IT managers, engineers, and scientists. They will want to know the details of the method you describe, as well as “why do I care?” ☺

In describing your topic, please consider the following aspects:

- Answer questions of what, why, when, where, how, and who.
- Describe both theory and computer tool support.
- Describe implications for practitioners, i.e., does the method really help?
- Include at least **four** academic references (peer-reviewed articles) in your research.
- Do not summarize (substantially) the course materials; assume them as background and add new materials.
- Write the article using the wikipedia.org style. **Your article will be posted** (by the instructor) on our course web page as a Blog article—it will be visible on the WWW.

Include appropriate web links and article references using EndNote; See the exam review for a description. (To be updated a week before the exam.)

How to Scan *CIS Literature*.

- **Deliver** your article to the instructor as an email attachment
 - The article must be a Word document. Ensure that it is spell and grammar checked.

Topic presentation

- **Deliver** a PowerPoint presentation of your topic on the scheduled date
 - You will have 10 – 15 minutes, depending on class size
- **Deliver** your PowerPoint slides to the instructor as an email attachment

Examinations

Exam 1

See the exam review for a description. (To be updated a week before the exam.)

Exam 2

See the exam review for a description. (To be updated a week before the exam.)

How to Scan CIS Literature

Software

Install EndNote:

1. **Free** [EndNote @ GSU](#)



Literature Review

Search for peer reviewed articles using keywords:

2. Scan the web
 - a. www.google.com
3. Scan the web using scholar search engines
 - a. <http://scholar.google.com/>
 - i. Set the [Google Scholar Preferences](#) to
 1. Show library access links for Georgia State University
 2. Show links to import citations into EndNote
 - b. <http://academic.live.com/>
 - c. <http://citeseer.ist.psu.edu/>
4. Scan using library databases (@GSU)
 - a. <http://www.galileo.usg.edu>
 - b. In particular, the following databases
 - i. [ABI/INFORM Complete](#)
 - ii. [ACM Digital Library](#)
 - iii. [IEEE Xplore](#)

Workload Expectations

Students should plan for 2 - 3 hours of work outside of class each week for each course credit hour. Thus, a 3-credit course averages between 6 and 9 hours of student work outside of the classroom, *each week*. See GSU sites for Academic Success:

- <http://www2.gsu.edu/~wwwcam/incept/successtips.html>
- <http://www2.gsu.edu/~wwwctr/sac/StudySkills.htm>

Students must take responsibility for their learning. In contrast to high school, college has fewer opportunities for student teacher interactions. Consequently, students must prepare to gain the most from each interaction.

Self-Managed Teams: Teams will be allowed for some activities during the term. Please note that unless the activity is explicitly identified as a "team activity", I expect everyone to perform their own work (your hands on the keyboard). For team activities, you will be allowed to work with partners (of your choosing).

- Initial teams must be established by the second week of classes. Established teams may continue working together on subsequent team activities. Team membership may change during the term, if problems arise. However, team members must be designated within one week of the due date for the team activity. Exception: you may withdraw from a team at any time and submit an assignment individually.
- Teams will submit one assignment for all team members. In most cases, each member of the team will get the same score. However, an individual's score may be reduced at the discretion of the instructor.
- **Each team assignment must include the following:**
 - Tasks completed by each member.
 - Percentage of the total work completed by each member.
- Any individual with a low team contribution will be removed from their team.

Arbitration: There will be a one-week arbitration period after graded activities are returned. Within that one-week period, you are encouraged to discuss any assumptions and/or misinterpretations that you made on the activity that may have influenced your grade.



Attendance: If you are unable to attend a class session, it is your responsibility to acquire the class notes, assignments, announcements, etc. from a classmate. The instructor will not give private lectures for those that miss class.

Submission of Deliverables: Unless specific, prior approval is obtained, no deliverable will be accepted after the specified due date.

If you have a legitimate personal emergency (e.g., health problem) that may impair your ability to submit a deliverable on time, you must take the initiative to contact the instructor before the due date/time (or as soon after your emergency as possible) to communicate the situation.

Make-up exams will not be given: However, if a student has a planned absence, he or she may take the exam **earlier** with the permission of the instructor.

Student Behavior

Behavior in class should be professional at all times. People must treat each other with dignity and respect in order for scholarship to thrive. Behaviors that are disruptive to learning will not be tolerated and may be referred to the Office of the Dean of Students for disciplinary action.

Discrimination and Harassment

Discrimination and/or harassment will not be tolerated in the classroom. In most cases, discrimination and/or harassment violates Federal and State laws and/or University Policies and Regulations. Intentional discrimination and/or harassment will be referred to the Affirmative Action Office and dealt with in accordance with the appropriate rules and regulations.

Unintentional discrimination and/or harassment is just as damaging to the offended party. But, it usually results from people not understanding the impact of their remarks or actions on others, or insensitivity to the feelings of others. We must all strive to work together to create a positive learning environment. This means that each individual should be sensitive to the feelings of others, and tolerant of the remarks and actions of others. If you find the remarks and actions of another individual to be offensive, please bring it to their attention. If you believe those remarks and actions constitute intentional discrimination and/or harassment, please bring it to my attention.

Official CIS Department Class Policies

1. Prerequisites are strictly enforced. Students failing to complete any of the prerequisites with a grade of "C" or higher will be administratively withdrawn from this course with *loss of tuition fees*. **There are no exceptions, except as granted by the instructor with the approval of the department.**
2. Students are expected to attend all classes and group meetings, except when precluded by emergencies, religious holidays, or bona fide extenuating circumstances.
3. Students who, for non-academic reasons beyond their control, are unable to meet the full requirements of the course should notify the instructor, by email, as soon as this is known and prior to the class meeting. Incompletes may be given if a student has ONE AND ONLY ONE outstanding assignment.
4. A "W" grade will be assigned if a student withdraws before mid-semester if (and only if) he/she has maintained a passing grade up to the point of withdrawal. Withdrawals after the mid-semester date will result in a grade of "WF". See the GSU catalog or registrar's office for details.
5. Spirited class participation is encouraged and informed discussion in class is expected. This requires completing readings and assignments **before** class.



6. All exams and individual assignments are to be completed by the student alone with **no** help from any other person.
7. Collaboration within groups is encouraged for project work. However, collaboration between project groups will be considered cheating.
8. Copying work from the Internet without a proper reference is considered plagiarism and subject to disciplinary action as delineated in the GSU Student Handbook.
9. Any non-authorized collaboration will be considered cheating and the student(s) involved will have an Academic Dishonesty charge completed by the instructor and placed on file in the Dean's office and the CIS Department. All instructors regardless of the type of assignment will apply this Academic Dishonesty policy equally to all students. Abstracted from GSU's Student Handbook Student Code of Conduct "Policy on Academic Honesty and Procedures for Resolving Matters of Academic Honesty"
 - a. http://www2.gsu.edu/%7Ewwwdos/codeofconduct_conpol.html
 - b. <http://www2.gsu.edu/~wwwcam/>

As members of the academic community, students are expected to recognize and uphold standards of intellectual and academic integrity. The University assumes as a basic and minimum standard of conduct in academic matters that students be honest and that they submit for credit only the products of their own efforts. Both the ideals of scholarship and the need for fairness require that all dishonest work be rejected as a basis for academic credit. They also require that students refrain from any and all forms of dishonorable or unethical conduct related to their academic work.

Students are expected to discuss with faculty the expectations regarding course assignments and standards of conduct. Here are some examples and definitions that clarify the standards by which academic honesty and academically honorable conduct are judged at GSU.

Plagiarism. Plagiarism is presenting another person's work as one's own. Plagiarism includes any paraphrasing or summarizing of the works of another person without acknowledgment, including the submitting of another student's work as one's own. Plagiarism frequently involves a failure to acknowledge in the text, notes, or footnotes the quotation of the paragraphs, sentences, or even a few phrases written or spoken by someone else. The submission of research or completed papers or projects by someone else is plagiarism, as is the unacknowledged use of research sources gathered by someone else when that use is specifically forbidden by the faculty member. Failure to indicate the extent and nature of one's reliance on other sources is also a form of plagiarism. Any work, in whole or part, taken from the Internet or other computer based resource without properly referencing the source (for example, the URL) is considered plagiarism. A complete reference is required in order that all parties may locate and view the original source. Finally, there may be forms of plagiarism that are unique to an individual discipline or course, examples of which should be provided in advance by the faculty member. The student is responsible for understanding the legitimate use of sources, the appropriate ways of acknowledging academic, scholarly or creative indebtedness, and the consequences of violating this responsibility.

Cheating on Examinations. Cheating on examinations involves giving or receiving unauthorized help before, during, or after an examination. Examples of unauthorized help include the use of notes, texts, or "crib sheets" during an examination (unless specifically approved by the faculty member), or sharing information with another student during an examination (unless specifically approved by the faculty member). Other examples include intentionally allowing another student to view one's own examination and collaboration before or after an examination if such collaboration is specifically forbidden by the faculty member.

Unauthorized Collaboration. Submission for academic credit of a work product, or a part thereof, represented as its being one's own effort, which has been developed in substantial



collaboration with another person or source or with a computer-based resource is a violation of academic honesty. It is also a violation of academic honesty knowingly to provide such assistance. Collaborative work specifically authorized by a faculty member is allowed.

Falsification. It is a violation of academic honesty to misrepresent material or fabricate information in an academic exercise, assignment or proceeding (e.g., false or misleading citation of sources, the falsification of the results of experiments or of computer data, false or misleading information in an academic context in order to gain an unfair advantage).

Multiple Submissions. It is a violation of academic honesty to submit substantial portions of the same work for credit more than once without the explicit consent of the faculty member(s) to whom the material is submitted for additional credit. In cases in which there is a natural development of research or knowledge in a sequence of courses, use of prior work may be desirable, even required; however the student is responsible for indicating in writing, as a part of such use, that the current work submitted for credit is cumulative in nature.