Fall 2020 Syllabus

CPS 391: Junior Seminar I

CPS 491: Software Engineering Project I

These courses meet together, by design, but have separate Canvas sites.

Mon. 4:10-5:40pm, MAC 211 (in MacDonald Hall)

(This the live version. Watch the schedule section for routine updates.)

Professor

Russ Tuck, Ph.D.
russ.tuck@gordon.edu, KOSC 243, 978-867-3754 (O), 408-335-0890 (M)
Office hours: MWF 9:15-10:15am, TR 2-3:30pm and by appointment at https://russtuck.youcanbook.me/**.** Or drop in when my door is open. To visit my regular office hours by Zoom without an appointment, use the link in the "schedule" section of https://360.gordon.edu/profile/Russ.Tuck (click "Show the Schedule). If I'm not there, text my mobile number above. (I might be in an appointment meeting.)

CPS 391: Junior Seminar I (2 credits)

Catalog Course Description: Explores principles and practices of computer science in various professional contexts, including related theological and ethical issues. Students read papers and sections of textbooks, present them in class, and lead related discussions. BS students' readings focus on the software development process. Also follows the progress of projects in CPS 491. **Prerequisites**: none (but junior-level experience in computer science is assumed).

CPS 491: Software Engineering Project I (2 credits)

Catalog Course Description: Students work in teams to develop requirements, specifications, high-level design and prototype code for a computerized solution to an actual problem. The project is described in oral reports and written documentation. Readings and class discussion related to process. **Prerequisites**: CPS 391 (may be taken concurrently)

Course Objectives

This pair of courses is intentionally interwoven, to better achieve their goals. Students in CPS 391 read about and lead discussions about the software development process, and observe students in CPS 491 practicing it. Students in CPS 491 practice software development, and reflect on the process in discussion with students in CPS 391. Both classes also prepare and refine their resumes, in order to apply successfully for an internship or professional job.

Primary goals of CPS 391 include:

1. Gain familiarity with the process of software development through reading, oral presentations, and class sessions, and through observing the progress of students in CPS 491.
2. Understand how these practices reflect a biblical understanding of human nature.
3. Gain familiarity with some professional roles of computer scientists through reading, oral presentations, and class sessions.
4. Practice communicating technical ideas clearly and effectively, in both written and oral forms, and practice facilitating technical discussions.

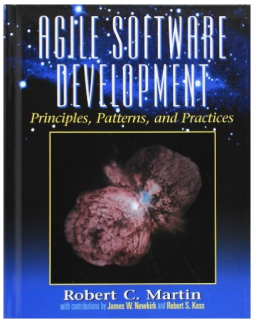
Primary goals of CPS 491 include:

1. Practice Agile software development by beginning a two-semester senior project to solve a problem for a real customer.
2. Practice designing, documenting, and describing a software system clearly and effectively, through both oral presentations and written documents.
3. Make the transition from small-scale class projects to the kind of thinking and methodology needed for medium and large scale projects which are more typical in industry. Upon completion of this course and part 2, CPS 492, you should have a general familiarity with the best principles and practice of Software Engineering.

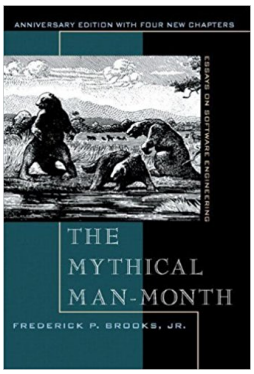
A shared goal of these courses is to become independent, life-long learners. In order to practice this and develop supporting skills, these courses are structured quite differently from a traditional lecture course. Most of the content will come from your own study and project work, and from class sessions led by class members. The professor will often be a mentor, reviewer, resource, and backstop.

Resources

Required Texts (These will also be used for CPS 492.)



There is one copy of each in the workstation lab, KOSC 244. **For your safety, sanitize your hands BEFORE and AFTER touching these books.** Do not remove the lab copy from the lab.

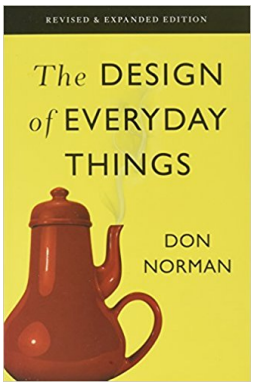


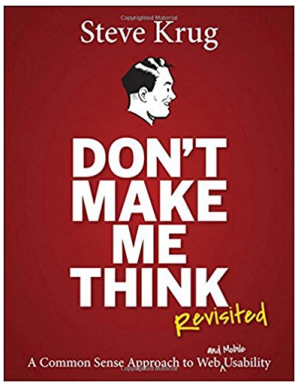
- Martin, Robert C. ***Agile Software Development: Principles, Patterns, and Practices*** (Upper Saddle River, NJ: Pearson Education/Prentice Hall, 2003). ISBN-13: 978-0135974445, ISBN-10: 0135974445.
- Brooks, Frederick P. Jr. ***The Mythical Man-Month. Anniversary edition*** (Reading, MA: Addison-Wesley, 1995). ISBN-13: 978-0201835953, ISBN-10: 0201835959.

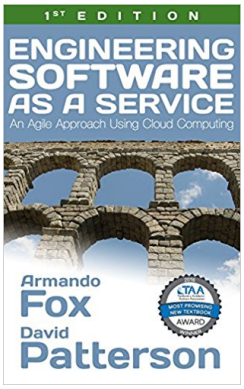
Recommended Texts:

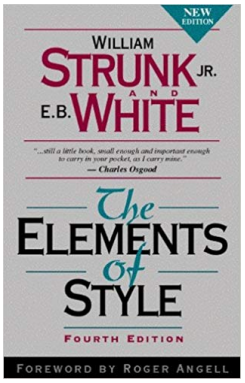
There is one copy of each in the workstation lab, KOSC 244. **For your safety, sanitize your hands BEFORE and AFTER touching these books.** Do not remove the lab copy from the lab.

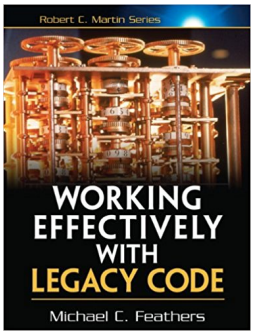
- Norman, Don. ***The Design of Everyday Things (Revised and Expanded Edition)*** (Basic Books, 2013), ISBN 978-0-465-05065-9.
- Krug, Steve. ***Don't Make Me Think, Revisited*** (New Riders, 2014), ISBN 978-0-321-96551-6.
- Fox, Armando and David Patterson. ***Engineering Software as a Service: An Agile Approach Using Cloud Computing*** (Strawberry Canyon LLC, 2016). ISBN-13: 978-0984881246, ISBN-10: 0984881247.
- Strunk, William Jr and E.B. White, **The Elements of Style, Fourth Edition** (Pearson Education, 2000, 1979), ISBN-13: 978-0205309023, ISBN-10: 9780205309023.
- Feathers, MIchael C. ***Working Effectively With Legacy Code*** (Prentice Hall, 2004), ISBN-13: 978-0131177055, ISBN-10: 0131177052.
- Beyer, Betsy and Chris Jones, Jennifer Petoff, and Niall Richard Murphy, ***Site Reliability Engineering: How Google Runs Production Systems*** (O'Reilly, 2016), ISBN-13: 978-1491929124, ISBN-10: 149192912X.
- Hicks, Marie and William Aspray, **Programmed Inequality, how Britain discarded women technologists and lost its edge in computing** (MIT Press, 2017), ISBN-13: 978-0262535182, ISBN-10: 0262535181.
- DeMarco, Tom and Timothy Lister, **Waltzing with Bears** (Dorset House Publishing, 2003), ISBN-13: 978-0932633606, ISBN-10: 0-923633-60-9.

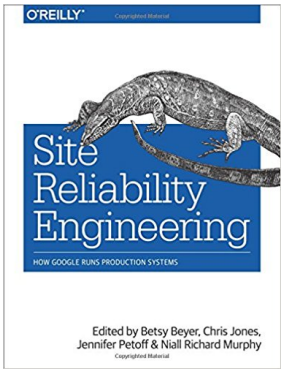


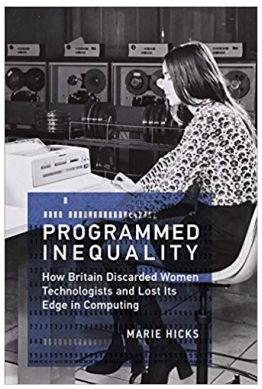


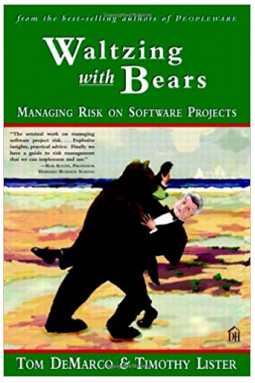












Online:

- Cockburn, Alistair, ***People and Methodologies in Software Development***, Ph.D. dissertation, University of Oslo, Feb. 25, 2003. Downloaded from https://www.researchgate.net/publication/253582591_People_and_Methodologies_in_Software_Development on 8/25/18.

Course Requirements and Evaluation

For each semester hour of credit, students should expect to spend a minimum of 2-3 hours per week outside of class in engaged academic time. This time includes reading, writing, studying, completing assignments and group projects, among other activities.

If you are taking both CPS 391 and CPS 491 now, plan to spend AT LEAST 8-12 HOURS PER WEEK, PLUS CLASS TIME, on this course. Plan your other commitments accordingly, and limit them if necessary.

Read the rest of this section carefully. It is the instructions and the assignment for most of the course. **Note that homework is due BEFORE the first class meeting.**

Reading and Discussion

Each class will have required readings. For classes marked with RR, for Reading Response, CPS 391 students will write a summary and analysis of the required reading. The purpose is to convince the professors that you read, understood, and deeply thought about the readings. Since this is the biggest piece of work for the course, it should take 3-5 hours to do it well.

The RR should have a section for each reading, and one set of questions you would like to see discussed in class. Use the reading names as section headings, and "Questions" as the final section heading. Each section should give a brief summary of the big ideas of the reading and, if possible, describe how and why the reading applies (or doesn't apply) to projects the CPS 491 students are working on. If you can't answer that, at least summarize the most important points.

The RR should be 1-2 pages overall, single spaced, including the questions. Practice insightful, clear, correct, and concise writing. Impress me with your deep understanding and your interesting questions. Communicating clearly, often in writing, is important in practicing computer science. Please add a note at the end, required but ungraded, saying how long it took you to do the reading and writing. (You can use this to see if the writing gets easier with practice, and I can use it to measure and potentially adjust the reading workload.)

The RR must be submitted in Canvas before midnight on Thursday before the Monday class. Students leading the discussion the subsequent Monday will use your questions in preparing the discussion, and attempt to find answers to them in the readings.

Grading: the professor will sometimes give detailed feedback, and sometimes very brief or even just pass/fail. The goal is to use finite grading time for maximum benefit, knowing that it's not possible to give detailed feedback on all this writing. The following rubric will be used when possible. The rows will be summed, and the score is out of 10 (so exceptional work in one row can balance less good work in another for full credit, and better than 100% is possible). If there is no reading response for one or more readings, the score will be pro-rated based on the number of readings covered. (That is, someone gets 4s in all rows based on what they wrote, but they skipped one of four readings, they will get a 3 overall -- because they did 3/4 readings but got full points for what they did.)

Content	Missing or seriously incomplete 0	Reasonably complete and somewhat insightful 4	Thorough and insightful, with good application to project 6
Writing	Several spelling and grammar errors, or very wordy or terse or unclear. 0	Mostly clear and correct writing. Reasonably concise. 2	Consistently concise, clear, and correct. 3
Questions	Missing or seriously incomplete 0	3 pertinent and somewhat insightful questions 2	3+ questions showing excellent understanding and insight. 3

Students who are only in CPS 491: you are responsible for reviewing the readings and your RR from the previous year, in order to participate actively in the discussion. But you are not required to submit a RR. Instead, review the submitted questions and think about your project, and try to come prepared to be an "expert participant" in the discussion.

Leading Class Session(s)

Several times during the semester, each CPS 391 student will be responsible for leading part of a class session on one of the topics. There will be an opportunity to sign up for specific topics at the first regular class meeting. You will be designing a learning experience for your peers, so give serious thought to how to help them learn. They will have read the material carefully, so don't just repeat it. Focus on helping them apply the material. It's fine to emphasize some of the most important points, and it's good to explain major implications that you see beyond the reading itself. But your primary goal is to lead a productive discussion. Use the submitted questions well, because they are clear indications of interest and imperfect understanding. See if you can find good answers in the readings, and consider how these questions can help you lead a great discussion. Basic Tips for Leading a Discussion by Dr. Kent Seibert is a helpful guide.

You may want to meet with the professor during his office hours the week before to describe your plan and get advice.

Students and the professor will evaluate each student-led session, and the professor's score will carry extra weight in the grade.

Software Project (CPS 491 only)

Choosing a Senior Project, and Potential Senior Project Ideas give instructions for forming teams and choosing a customer and a project.

Status Reports

Status reports (and individual reports, before teams are approved) are due in a shared Google Doc (linked from Canvas) before classes marked with SR (for Status Report) on the schedule. Status reports should follow the instructions at the bottom of that document. Be sure to include:

- Accomplishments
- Issues or Problems
- Result compared to goal
- Goal for next week

Presentations

Each member of a team must participate in each presentation in a roughly equal fashion. Students and the professor will evaluate each presentations, and the professor's score will carry extra weight in the grade.

Documents Requiring Customer and Department Signatures

Each of the following documents must by approved by your customer before it is submitted. Therefore, you'll need to **give a draft to your customer several days before the due date**, so they can review it, and so you can resolve any issues and concerns they raise before it's due.

- Initial problem statement (class 3)
- User stories in Github (class 4)
- MVP Description (class 10)

The customer should indicate their approval by email, cc'd to the professor. This email should be copied and pasted into the approved document.

Each of these documents must also be approved by your assigned department representative. If they are not satisfied, you will need to revise it until they approve.

Code

All code will be managed, reviewed, and submitted with Git and GitHub, in the Gordon CS organization (https://github.com/gordon-cs/). Each CPS 491 student is expected to do their fair share of design, documentation, and coding. **This must be reflected in commits which are merged into the develop and main (aka master) branches.**

All code must be reviewed by another team member via a pull request before being integrated into the team's develop branch. After further testing, the develop branch should be merged into the main (aka master) branch.

If two students pair-program, they may use a shared branch, but they must alternate who types, and whose computer and account they use, so that experience and commits are done roughly equally. Commit messages for pair-programmed code should always name the other student in the pair. Pair-programmed code is considered reviewed by the process of pair programming, so the pull request should note this and may be integrated into develop right away.

Documentation (section added 9/21/2020)

All documentation for the project must satisfy these logistical requirements:

- Clearly linked from github repo's Readme.md
- Written with a system which shows revision history and authorship -- so reviewers can see changes and make comments, and I can see who did what. (Github markdown files and Google Docs are examples of systems meeting these requirements.)

In some cases, links to the documentation should be submitted in Canvas before it is due.

Grading

CPS 391	Weight
Reading Responses	60%
Leading Class	15%
Resume and Cover Letter	15%
Feedback on CPS 491 MVP Presentations	10%

CPS 491	Weight
Resume and Cover Letter	10%
Status Reports (both the report, and the progress reported)	15%
Proposal and documentation: (30% total) Initial Problem Statement (class 3) User stories in Github (class 4) End to End Hello World and Tool proposal (class 6) Mid-term Presentation with end-to-end "Hello World" demo (class 7) MVP Description approved by customer (class 10) GitHub use: clear commit and pull request descriptions, good code reviews	 2% 6% 2% 10% 5% 5%
Lo-fi UI Usability Tests with Customer (at least 2)	10%

MVP: Demo, Usability Test Report, Functionality, and Quality	35%

While most of the work in CPS 491 is team work, and grades will commonly be given to the team as a whole, each team member must contribute substantially, and differences in contribution will be reflected in the grades as appropriate. See the section on Honesty, below, for important requirements.

The following are minimum guaranteed grades for the scores indicated:

98% - 100%: A+	93% - 97.9%: A	90% - 92.9%: A-
87% - 89.9%: B+	83% - 86.9%: B	80% - 82.9%: B-
77% - 79.9%: C+	73% - 76.9%: C	70% - 72.9%: C-
67% - 69.9%: D+	63% - 66.9%: D	

Honesty

Never lie by presenting another's work as your own - whether by copying code, ideas, test answers, homework, or otherwise. It is always better to give credit where it's due, and to confess confusion, ignorance, or difficulty and get help. The Computer Science Project Guidelines elaborate on how to give proper credit. **They also make it clear that substantial differences in contribution should be clearly indicated in the project documentation.** (It is common for students to contribute somewhat differently to different aspects of the project, reflecting their skills and interests as well as the needs of the team. Documenting the notable efforts of each team member is good, and helps meet the Project Guidelines requirements.)

Some practical advice for this course, since we will be using Git and GitHub heavily.

- When you pair-program, note that in the commit message. Include "(with Name)" on the first line. Never rob your partner of credit. And take turns "driving", so some of the commits come from each of your accounts.
- When someone contributes suggestions, ideas, debugging help, etc, give them credit in a comment or in the commit message.

Note that Gordon College's Academic Dishonesty policy (in the Student Handbook) requires that plagiarism result in an F for at least the assignment, and in some cases the entire course. Further consequences can include suspension from the College.

For completeness, the standard statement also applies: Academic dishonesty is regarded as a major violation of both the academic and spiritual principles of this community and may result in a failing grade or suspension. Academic dishonesty includes plagiarism, (see Plagiarism in Student Handbook), cheating (whether in or out of the classroom), and abuse or misuse of library materials when such abuse or misuse can be related to course requirements

Extensions and Incompletes

Due to the nature of the course, extensions and incompletes will be considered only in the most dire of circumstances.

If you are in CPS 491 and in the General Concentration and an incomplete becomes necessary, it must be made up by the start of classes for spring semester in order to continue in CPS492. If you get behind early in the year, it will be very difficult for you to finish your project on time. You MUST keep up!

Attendance Policy

Attendance at all sessions is mandatory. Each unexcused absence will result in a reduction of 1/3 of a letter grade in the term grade.

Accommodations for Students with Disabilities

Our academic community is committed to providing access to a Gordon education for students with disabilities. A student with a disability who intends to request academic accommodations should follow this procedure:

1. Meet with a staff person from the Academic Success Center (ASC) and provide them with current documentation of the disability.
2. Obtain a Faculty Notification Form from the Academic Success Center, listing appropriate accommodations.
3. Submit this form to professors and discuss those accommodations with them, ideally within the first two weeks of classes.

Some accommodations need more time to arrange so communicating early in the semester is important. For more information consult the Academic Success Center's web page (https://www.gordon.edu/asc) or email asc@gordon.edu.

Course Schedule (subject to change)

Abbreviations: F/P = Fox & Patterson; p. = page(s); ch. = chapter(s)
RR = 391 only: Reading Response (**due Thurs midnight** before class);
SR = 491 only: Status Report (due: **individual Sat midnight, team Mon 9am**)
SL = Student-led session;
MVP = Minimum Viable Product;
Team meetings = each team meets privately with prof;

#	Date	Topic	Reading Due	Work Due
1	M 8/24	Course intro; Discuss teams, customers, and projects.		

		(slides)		
2	M 8/31	Intro to Software Engineering and Agile Development Discuss proposed teams, plus customers and projects. (slides)	Syllabus; **Choosing a Senior Project**, and **Potential Senior Project Ideas** Brooks ch.1 (p.3-9); Fox ch.1-1.5 (p. 1-21); Martin ch.1-2 (p. 1-17); *Optional, ungraded (not in RR):* Martin appendix C (p. 507-516), Cockburn "Abstract" (pdf p. 4) and section 3.4 (p. 49-53 of its numbers, pdf p.52-56)	RR; All: GitHub ID and Google account name 391: SL topic preferences (but tell me immediately if you'd like to lead class 3) 491: Proposed Team; If possible, do Project Problem Statement now, even though it's not due until next week.
3	M 9/7	Present project proposal to class and give each other feedback (SL) Requirements and Human-Centered Design: Figuring out what to build Writing well (slides)	Fox ch. 7-7.4 (p. 218-229); Norman ch.6 through p.247 (p. 217-247 and 2 lines of 248); Strunk & White, part II (p. 15-33)	RR; SR; 491: Customer Contact and GitHub Info; Project Problem Statement and Slides to present problem statement, linked from repo's README.md;
4	M 9/14	(SL:) Process: Usability Testing Writing Well (slides)	Krug ch. 9 (p. 110-141); Strunk & White, p. 70-74	RR; SR; 491: User stories in github project in your repo (or in Pivotal Tracker, with a link from your repo's Readme.md), arranged in a logical order, and other requirements in a doc.

				491: Schedule meeting this week with me and customer to discuss user stories.
5	M 9/21	(SL: unit, sys/int, load/perf) Process: Testing (unit testing, system/integration testing, load/performance testing) -- each presenter should demonstrate an example, using 360.gordon.edu (and its tests, where it has good examples), and help classmates work together in class to write another one of each (SL:) Process: Basics of Source Configuration Management What did you learn from 1st usability test? (slides)	Feathers, "Test-Driven Development", p. 88-94; Beyer, p. 185-191; Git - Book chapters 1-3 (sections 1.1-3.7) (extra credit: https://martinfowler.com/articles/testing-culture.html - ***also applies to security***) Optional resources: Architecture (major concepts): https://aosabook.org/en/git.html, How it works, bottom-up: Becoming a Git pro. Part 1: internal Git architecture, Overview: The Architecture and History of Git: A Distributed Version Control System	RR (note things you don't understand or disagree with); SR; 491: Report of 1st Lo-fi UI usability test with customer. Submit in Canvas, where there are detailed instructions.

6	M 9/28	Job search and grad school application strategies, resume writing, and preparing for a technical interview. Guest presenters: (not this year) Discuss tool proposals and hello-world definitions. (slides)		Draft resume (submit in Canvas and be prepared to share with speaker) SR; 491: End to End Hello World and Tool Proposal (part. 1) - meeting with me while preparing this is a good idea
7	M 10/5	(SL:) Process: Security Best Practices Present recent work: ● give Lo-fi demo of core UI ● report on usability tests Team meetings as needed (slides)	CMU SEI Top 10 Secure Coding Practices; OWASP Secure Coding Practices Quick Reference Guide; Correction: https://owasp.org/www-pdf-archive/OWASP_SCP_Quick_Reference_Guide_v2.pdf And https://owasp.org/www-project-top-ten/	~~"Perfect" resume and cover letter (no mistakes from this list);~~ (delayed by my slow feedback) SR; 491: Report of 2nd Lo-fi UI usability test with customer and other potential users.
8	M 10/12	Present recent work: ● show end-to-end hello world demo and system diagram, with clear evidence it's all working ● describe 1st story to implement (SL:) Process: Changing Legacy Code with Refactoring (slides)	Feathers, p. 18-28 ("The Legacy Code Change Algorithm" and "Ch. 3 Sensing and Separation");	RR; SR; "Perfect" resume and cover letter (no mistakes from this list); 491: Demo of end-to-end Hello World, with how-to documentation (keep updating it so it becomes documentation for your project). Identify 1st story to implement.

9	M 10/19	Standup report on first story sprint; (slides) Reading details: ● **Some** readings are assigned by last name: ○ A-L read “A” ○ M-Z read “Z”. ● Presenters read both for their topic. 1. (SL:) Process: Finishing -- Replacing and retiring the old system 2. (SL:) Process: Large Projects 3. (SL:) Process: Code Reviews Team meetings as needed	1. A: Replacing Legacy Systems (part 1); Z: Delivering Success when Replacing a Software System; A&Z: Google Dumps MapReduce; 2.Brooks ch. 8 (Calling the Shot); 3. Software Engineering at Google just sections 2.2-2.4; A: Lessons From Google: How Code Reviews Help Build a Better Company Culture, Z: Code Health: Too Many Comments on Your Code Reviews?	RR; SR: 491: 1st story working (include convincing screen shots in your status report) and **all related code committed, reviewed, and merged**. 491: Stories for next sprint identified in project (in “in progress” or “do next” column)
10	M 10/26	Demo of 2nd sprint Discuss upcoming milestones (SL:) Process: Site Reliability Engineering (SRE) -- release, reliability, monitoring (SL:) Process: Internationalization (I18n) & Localization (L10n) (slides)	SRE: Beyer: Preface (p. xv-xviii), Part I and intro to Part II (p. 1-24) Facebook motto i18n/l10n: Comparative definition; Kwintessential blog; Background, just skim, don’t include in RR: Wiki overview; Chrome.i18n example	RR; SR; 491: Demo of 2nd sprint
11	M 11/2	(SL:) Role: UX - User Experience Designer (SL:) Role: Product Manager	UX Designer: Norman ch2 p.37-73 Psychology of	RR; SR;

		~~Prof. led~~ ~~● GMail launch process & schedule~~ (slides) ~~5:30pm: Role: Data Scientist -- Special Guest: TBD~~	Everyday Actions Product Manager: HBR: SW Redefines PM, *and* Forbes: Great PM	491: Document describing MVP approved by customer (see syllabus for process), with total story points and estimated velocity showing it will be working this semester; all stories required by MVP show story points and have "mvp" label in github project board. (If achieving MVP is too much work, explain why, give a reasonable date for the MVP, and also document an intermediate milestone to demo this semester.) 491: Schedule meeting this week with me and customer to discuss your plan.
12	M 11/9	How hard was it to get to a feasible MVP definition? What made it hard? (slides) Other roles - panel of CTS professionals: ● Database Administrator ● Systems Adminstrator ● Systems Integration ● Network Administrator ● Technical Support ~~● Technical Writer~~ ~~● Technical Sales~~ Role: Data Scientist -- Special Guest: Joy Kimmel '15 ~~(SL: SN) Role: Maker~~ (SL: SM) Role: People Manager	Maker: Maker's Schedule, Manager's Schedule People Manager: Peter Kelly, Juan Rivera, Kate Matsudaira's Nine Things and Views from the Top Letter to a Young IT Worker	RR; SR
13	M 11/16	Demo of progress toward MVP	DeMarco *Waltzing with Bears*, prolog &	RR (due Wed 11/27); SR;

		(SL:) Role: Maker (SL:) Role: Project and Risk Manager (just the DeMarco reading) Prof. led: • GMail launch process & schedule (earlier) • Human nature and agile development (slides)	chapters 1-7 (p. 3-50) & 16-17 (p.128-142) Computer Science: Creating in a Fallen World Optional (no RR required): Hope is a Strategy (Well, Sort Of), *Waltzing with Bears* ch. 8-9 (p.53-72)	Demo of progress toward MVP
14	M 11/23	Brief Status Report show & tell. Finish discussion of human nature and agile development Intellectual Property (SL: EM) Roles: Inventor, Patent Agent, Attorney, & Examiner (but the readings address these indirectly, so **talk with the professor about what to do**) Equity (SL:) Programmed inequality : how Britain discarded women technologists and lost its edge in computing (slides) Please **read the special instructions in Canvas about how to write your RR.**	Copyright vs. Trademark vs. Patent vs. License: Everything to Know, Patent protection for software-implemented inventions, Which License Should I Use? MIT vs. Apache vs. GPL, MIT License, US Patent 8,533,274 (left column down through "Summary", and right column down through claim 22); Hicks *Programmed Inequality*, Introduction (p. 1-18)	RR (**special instructions in Canvas**); SR 491: major progress toward MVP
	11/30-12/4	**Final Demo** 491: 12/1-12/3 Schedule 30 minutes with professor over Zoom: 15 min. Demo (recorded) + 15 min private feedback 391: by 12/4 watch and give feedback on all 491 presentations (no late penalty if done by 12/8)		491: MVP demo 391: feedback on demos

* For each chapter in Brooks, read the corresponding section in ch. 19, where he reflects on his original writing 20 years later.

Preview of CPS 392

While CPS 391 (2 credits) includes a lot of reading, writing, and discussion, CPS 392 is simply observing the second half of all the projects continuing in CPS 492. CPS 391 students will observe major presentations and give some feedback. Observing how projects progress is important for everyone, and doubly so for students who will be in CPS 491-492 next year.

Preview of CPS 492

Since CPS 491 (2 credits) and 492 (4 credits) are focused on a single year-long project, it's helpful to understand the whole project schedule. So here's an outline of next semester.

Week #	**Major Milestone**
1	Project Demo: MVP plus
2	User Study report; Project backlog defines complete product
3	
4	Design Document
5	
6	Handoff Document, part 1; revised Design Document
7	Project Demo: Complete Product
8	User Study report; Project backlog defines final product
9	Handoff Document, part 2; revised Design Document
10	
11	Documentation complete
12	Documentation accepted by customer and professor; Updated User Study report (test that improvements worked)
13	Project Demo: Final Product
14	Reflection and Celebration