IC470 – Software Engineering

Fair warning: This is a fairly time intensive milestone, so get started early

A sample Mile 3 Part I (Formal IC480 Capstone Proposal) is available from the course's Resources page.

Milestone 3

Part I. Formal IC480 Capstone Proposal, and

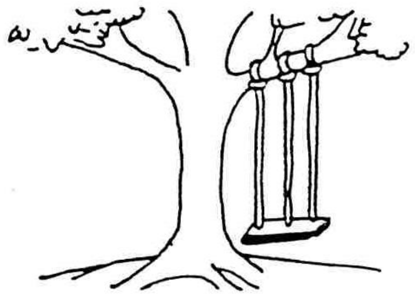
Part II. Targeted Design

Due date: Both Part I (no presentation, for Part I, just a binder is turned in) and Part II (includes a presentation and copies of slides turned in) are due at the start of class as per the syllabus. Also, note the Peer Review date given on the syllabus - the current draft of your Part I will be reviewed by your peers on that date, and feedback given with time for you to incorporate any changes into your Part I. The closer you are to a finished proposal by the peer review date, the better the feedback you will get from your classmates. Unlike Part I (which is required for students taking IC480 in the upcoming Spring semester), Part II is required for all students currently enrolled in IC470.

Errata/Updates. Any errata or updates to this document will be dated and shown in red both as a summary below as well as in the contents of this milestone. Each team is responsible for checking (and delivering their milestone in compliance with) any changes indicated in this Errata/Updates section that are dated prior to the delivery.

Summary of Errata/Updates:

None (yet)



As specified in the project request

Part I – Formal IC480 Capstone Proposal

(Part I is required for students taking IC480 in the upcoming Spring semester – note that Part I expands on portions of your earlier milestones and also includes some entirely new sections.)

Notes:

1. Your capstone proposal will be evaluated by Computer Science Department faculty members that have never seen your proposal or earlier milestones, so make sure the contents of this, your formal proposal, are complete and through enough that someone not familiar with your proposed capstone can understand the complexities of your project.
2. See the Capstone Proposal Review and Evaluation Rubric from the course's Resources page to ensure that you are familiar with how your capstone proposal will be evaluated.
3. **Growth:** Proposals require teams to explicitly identify an area of 'growth' for each team member, where 'growth' is defined as concepts that will require the student to learn how to independently find resources, apply and integrate their respective program knowledge, and overcome concepts/hurdles that go beyond those found in courses already taken in the program. Proposals must clearly include a significant amount of software development that goes beyond writing device drivers, integrating open source code or things that would make for little more than an interesting lab in a course.

Deliverables: Your team's Formal Capstone Proposal must include the following sections. Give a number and descriptive title for each figure, table, UI, etc, that you include, and discuss each figure in the body of your document. For example, use something like, "Figure 2 shows the relationship between …" to begin your discussion of Figure 2 in the body of your document.

a. **Electronic copy:** Each team leader is to email your IC470 instructor your team's Formal Capstone Proposal as a Word file attachment.
   1. **File naming convention:** Use the following naming convention for the file you email: GroupX_AY83_CapstoneProposal_InitialDraft.doc where X is your assigned team ID number (see the team compositions link on the course web page), and 83 is replaced with the last two digits of your class year.
   2. **Email Subj line:** Use "GroupX, Capstone Proposal - Initial Draft" as the subject line of the email via which you transmit your electronic copy to your instructor.

b. **Paper copy.** One paper copy of your Formal Capstone Proposal turned in to your IC470 instructor, in a 3-ringer binder with a clear plastic sleeve on the front cover and a clear plastic sleeve on the inside of the front cover. Insert a sheet in the clear plastic sleeve on the front cover of the binder giving your:
   1. Capstone Proposal's Title, Customer and Technical Advisor,
   2. Team number and Team Nickname, and Team Logo (see c.2.ii below)
   3. Team members and their majors (CS, IT, CS/IT (dual major), CE/CS,

etc), also, clearly indicate the team leader,

4. The words "**Capstone Proposal – Initial Draft**"
5. The paper copy (to include all figures) must be readable. This is not the time to save on ink.

c. **First 3 pages inside the binder.** Place the following as the first 3 pages inside your binder:

1. **Page 1**: **Evaluation Rubric**. A copy of the "Capstone Proposal Review and Evaluation Rubric" available from the course's Resources page. Fill in the indicated portion with your Capstone Title, Team Number, Team Leader's Name, and Customer's Name.
2. **Page 2: Title Page** including:
    - i. **Capstone Proposal's Title** (100 character limit – including spaces), **Customer and Technical Advisor information** (name, phone#, email, USNA affiliation).
    - ii. **Team number, Team nickname**, and **Team Logo.** Your team logo should be graphical, fun (but professional!) and have a tie-in with your project. If you need some help with this, jot some ideas down and head over to the Graphics Technology Lab on the first deck of Nimitz Library, Room 105 (x35856, mscgraphics@usna.edu) and contact the Graphics staff for assistance.
    - iii. **Team Composition**: Give the names of the team members and their majors (CS, IT, CS/IT (dual major), CE/CS, etc), also, clearly indicate the team leader,
    - iv. **Milestone Lead**. Identify the milestone lead (same requirements for Milestone Lead as Milestone 0).
    - v. **Customer Concurrence** statement. Customer Involvement Alert: At the bottom of your Title Page, place the following concurrence statement(s) and have your Customer and Technical Adviser initial the following concurrence statement(s) after reviewing your proposal (and prior to turning in your binders).

        a. The Customer and Technical Adviser initials must both be attained before a team may turn in their Formal Capstone Proposal. It is the team's responsibility to meet with their Customer and Technical Advisor with enough lead time in order to get the required initials and meet published delivery deadlines.

        b. **48 working hours rule**. Note that your customer or technical advisor not being available in the 48 working hours (ie., 2 work days) prior to a delivery deadline is not considered a valid excuse for lateness, and no delivery delays will be approved in these types of circumstances. This will instead be viewed as a planning shortfall on the part of the team and will result in a loss of points.

c. Teams with off-Yard Customers may attach an email acknowledgement from their Customer in lieu of initials.

d. Concurrence Statement(s) - Place these on the bottom of your cover page and have your Customer and Technical Advisor initial them:

Customer's initials: __________ **Date initialed:** __________ "I have reviewed the Teams Capstone Proposal and am satisfied with its contents including, in particular, the team's Acceptance-Testing-Focused Functional Requirements Trace Table. I understand that the team will be following the Agile Software Development approach and that I may (and am encouraged to) ask for the addition, removal, or modification of any functional or non-functional requirements and/or acceptance test cases at any time as the Capstone Project progresses."

Technical Advisor's initials: ___________ **Date initialed:** __________ Required only for teams whose Customer is not a faculty member in the Computer Science Department.

3. **Page 3: Table of Contents page**: include page numbers for each section as well as page numbers for each figure, table, and chart.

Organize the remainder of your project binder into sections with the following section titles and content. Note that you must give a number and descriptive title for each figure, table, UI, etc, that you include, and you must discuss each figure in the body of your document. For example, use something like, "Figure 2 shows the relationship between …" to begin your discussion of Figure 2. Do not put your pages into plastic sleeves, this slows the review process as some reviewers will prefer to write comments directly on the pages.

Page 4 and beyond:

d. **Abstract.** Give a 500 character or less (including spaces) abstract of your project. The severe character limit means you must concisely convey the essence of your project.

   i. You may find it helpful to take the mindset that your proposal is one of many being read by tech-savvy business investors. Your abstract serves as a pitch of your project idea in the hopes of it getting selected for funding (here, academic credit).

   ii. Your abstract must be clear enough that someone who was not at any customer interviews can get a clear, big-picture understanding of what your proposed project is about, and why it is important to undertake.

e. **Project Introduction**. Provide an in-depth introduction to your project. Include the following subsections (to include the titles):

   1. **Motivation.** Start with a persuasive paragraph (or two) whose purpose is to spark the reader's interest in your capstone project. Include your motivation for undertaking this project and why it is interesting to you (and by extension, should be interesting to the reviewers!) and is important to undertake. Keep it technical, and capture the reader's curiosity.
      1. Hint: Think about what you would tell the CEO of a tech company if you bumped into them in an elevator and they ask what you are doing for your capstone project.
      2. For example, "Sea-keeping in littoral environments is a difficulty faced by mariners throughout the world. Large tide variations typical of littoral environments present challenges to navigational aids dependent on precise positioning. GPS-enabled self-positioning buoys eliminate the need for physical connections, such as anchors, but require embedded position-aware software control systems capable of …"
   2. **Overview**. In this section, convey the goal, scope and purpose of your project. In a few paragraphs, expand on what the project entails as a preview of what it must do. While the precise requirements for your project will be identified in detail in the Functional Requirements Trace Table section, your objective here is to give the reader a discussion of the details of your project. The overview must expand on your motivation paragraph(s) and subtly lead the reader to a point where they are interested in learning more about the nitty-gritty specifics of your project.
   3. **High-Level Diagram**. Include at least one diagram (see Figure 1) giving a graphically appealing high-level view of your system to include major software/hardware components of what your project will entail and what other systems your project will interact with. This can be a semi-cartoonish image but must use persuasive graphics to represent systems (ie., Use a labeled cylinder to represent a database instead of just a box with the label "Database" on it). Discuss (and refer to) this figure in your overview paragraphs above.
   4. **Glossary**. Include a glossary that defines words (and their meaning in the context of your project) used in your proposal that may be unclear to a reviewer who may not be familiar with the nuances of your project area. Don't define things like "computer network" but do define things like SAT Solver, or MOOW.

   5. You must <u>refer to</u> and <u>discuss</u> all figures used throughout your proposal, for example, "Figure 1 shows a high-level view of the OCTOPUS Sea-Keeping System. Control flow for station keeping begins with the Wave Radar Buffer receiving a weather signal from the Radar Antenna. In response, the …" If you need help with the graphical aspects of this, head

over to the Graphics Technology Lab on the first deck of Nimitz Library, Room 105, call x35856 or email mscgraphics@usna.edu to contact the Graphics staff for assistance.

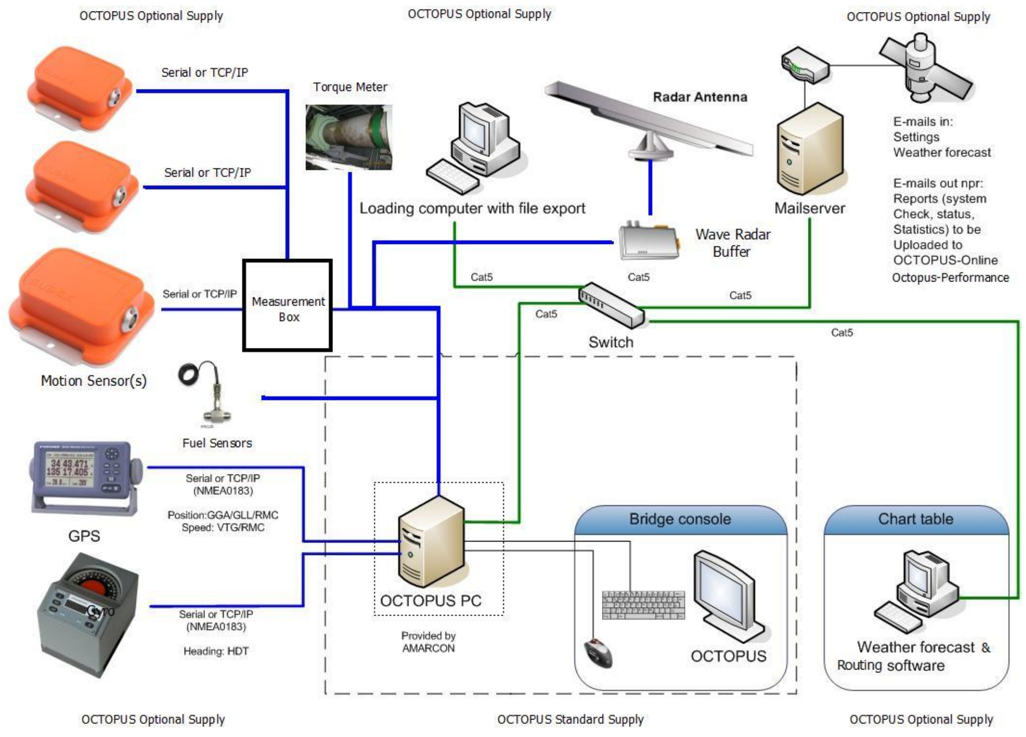


Figure 1. High Level View of the OCTOPUS Sea-Keeping System

e. **Justification.** Same requirement as Milestone 0.

f. **Customer's Current Process.** Same requirement as Milestone 0.

g. **Topical Areas**. Same requirement as Milestone 0.

h. **Existing software to be used and significant software to be developed by the team**. Include a paragraph or so discussion as well as a table (see Table A) indicating any existing software and algorithms that you expect to make use of in the left column, and in the right column give a high-level description of the software and algorithms that you expect to substantively develop yourselves. Note that people sometimes use the word "algorithm" to mean, "We don't know exactly what we'll do here, but it will probably be hard…" That's not helpful for planning! Instead, with a few paragraphs immediately below Table A, provide a detailed discussion of any algorithms you plan to use or create to include:

1. Existing Algorithms. If you plan to use, mimic, or build-on existing algorithms, name them and give an as-detailed-as-possible discussion of how you will use/modify them.
2. From-scratch Algorithms. If you plan to create algorithms from scratch, create a name for each and give an as-detailed-as-possible discussion of what the algorithms must do.
3. Algorithms is also a theoretical discipline within the computing domains. Include a discussion of whether your algorithm development or modifications will be focusing on asymptotic run-times/space, or, if other aspects are more important in your development/modifications, provide a discussion of these aspects.

Existing software the capstone team intends to use with little or no modification	**Software the capstone team intends to develop** from scratch or significantly modify (explain)
DeepBlues's Open Source Network Traffic Routing Algorithm (OSNTRA) modified to support BlueTooth.	An NLP algorithm of our own design, which we will call TrustKnot, which will improve its accuracy over time based on data classifications that are either approved or rejected by the user. In particular, this algorithm will …
…	A device driver to connect our software system with a Nomad motion sensor via a USB connection.
…	…

Table A. Existing software, and software to be developed

i. **Required Resources.** Same requirement as Milestone 0.

j. **Functional Requirements Trace Table.** Give a Functional Requirements Trace Table similar to Figure 2 that lists all the functional requirements for your entire system.

1) **Number of functional requirements.** In your Functional Requirements Trace Table, identify at least 2 (and no more than 4) functional requirements per team member. For example, a team with three members

must have at least 6 functional requirements, while a team of five would have at least 10 but no more than 20.

   a. Having too few functional requirements may mean that the proposal is not meaty enough for a capstone project given the team size - or that the granularity of the functional requirements needs to be further refined and perhaps broken down into smaller units.
   b. Having too many functional requirements may mean the proposal is too complex an undertaking for a semester-and-a-half capstone project - or that the functional requirements identified are too simplistic.
   c. See your instructor if you are having trouble complying with the indicated number of functional requirements for your team size.

2) Customer Involvement Alert Be sure that your Customer and Technical Advisor (if required) understand that we will be using the Agile method, and as such they are permitted (encouraged!) to add, modify or remove Functional Requirements at any time as the Capstone Project progresses and their needs are further refined. Note that the Course Instructor also reserves the right to add, dis-allow removal, modify or remove Functional Requirements as deemed necessary.

3) **Functional Requirements Trace Table Organization.** In the left column of your Functional Requirements Trace Table list each functional requirement (one per row). Organize your table as follows:

   a. In the center column of each row, enumerate all of the *objectively validate-able* Acceptance Test Plan test case scenarios (annotate each as either normal and abnormal and include the expected result) that in total can be used to conclusively show whether the final system fully meets all aspects of the indicated Functional Requirement.
   b. In the right column, indicate during which build your team anticipates completing this functional requirement. Assume five builds, and that at least 30% of all acceptance test cases will be completed by Build 1, 50% by Build 2, 65% by Build 3, 85% by Build 4, and 100% by Build 5.
   c. For planning purposes, you may assume that Build 1 will be completed during the IC470 semester, and Builds 2-5 will be completed during the IC480 semester. See the Gantt chart discussion in milestone 1 for more details.
   d. Customer Involvement Alert: Include your Customer in the decision of which acceptance test cases should be completed early on, and which can occur later.

4) **Necessary preliminary steps.** Include research or preliminary work you will need to do before you can start designing/implementing your capstone project as Functional Requirements (with a note that says "Preliminary Step).
   a. For example, foundational or exploratory activities, which we will term "preliminary steps," such as first demonstrating that you can render a simple line, then rendering a rectangle (which must both be accomplished as necessary prior steps leading to an eventual acceptance test case of rendering of a complex 3D shape of Bill the Goat) <u>should be included</u> as Functional Requirements or Acceptance Test Plan test cases, even if these preliminary steps won't directly end up in your final product.
   b. Identify the lead and backup midshipmen for each such requirement (see 6 below), however, <u>do not</u> include preliminary steps in your count of 2 (and no more than 4) functional requirements per teammate.
   c. For example, as part of a "Database" functional requirement, break out the subtasks that go into the database portion of your project, such as: developing the database schema, creation of the tables and user accounts, testing the database functionality under both normal and abnormal conditions, etc.
   d. Preliminary steps that are clearly test cases that contribute to an already existing functional requirement are to be added as Acceptance Test Plan test cases to the respective functional requirement. Add "Preliminary Step" as a descriptor to distinguish such test cases from the remainder of your test cases.
   e. Preliminary steps that don't directly contribute to an existing functional requirement, but that nonetheless need to be accomplished before you can proceed with some portion of your project, such as becoming familiar with how to use the OCTOPUS Application Programming Interface (API) are to be handled by adding a functional requirement, such as: "Learn how to use the OCTOPUS API (Preliminary Step)."
   f. Include Acceptance Test Plan test cases that prove that the preliminary step functional requirement has been met (such as a demo of your system interacting with the OCTOPUS API).

5) **Normal and Abnormal test cases**: For all your functional requirements, include (and identify as such) both normal and abnormal test cases for your system in your set of Acceptance Test Plan test cases. Recall that:

   a. **Normal** test cases are used to demonstrate that the system meets all or part of the indicated function requirement. These are the expected uses of the system. *For example, a user with the correct* *user name and correct password being allowed to log in by the system.*

b. **Abnormal** test cases are used to demonstrate the system's response to an unexpected, but possible, state (such as invalid user input) that the properly functioning system *could* find itself in. *For example, a user with the correct user name but an invalid password not being allowed to log in.*

c. **Not Useful** test cases involve states that a *properly* functioning system should not be able to be placed in, and will be considered "not useful" as test cases. This is because a correctly functioning system cannot demonstrate passing such test cases. *For example, a user with an incorrect user name and incorrect password being nonetheless allowed to log in by the system.* Do not include any "not useful" test cases in your Functional Requirements Trace Table.

6) **Lead developer.** As part of the table, identify which team member will primarily take the lead on the development of each functional requirement, and also identify a backup team member that supports the lead and assists as necessary.

7) **Mapping of Functional Requirements and their Acceptance Test Cases.** Number each Functional Requirement, and give a corresponding sub-number to each Acceptance Test Plan test case that contributes to demonstrating that the Functional Requirement is met.
   a. All possible uses of the system (both normal and abnormal) must be addressed by these test cases. For each test case, identify the expected result that allows the test case to be objectively validated.
   b. For example, Figure 2 gives the functional requirement for a Login/Password GUI that oversees users logging into the system, and three Acceptance Test Plan test cases (1.1, 1.2, and 1.3) that cover normal and abnormal scenarios that together demonstrate that the functional requirement has been completely met. Note that additions to the table from previous milestones is shown in **green**.
   c. Include the build number in which you plan to meet each functional requirement (by passing all acceptance test cases).

Functional Requirement	**Acceptance Test Plan test cases** (Set of scenarios that, in total, show that the Functional Requirement has been met.	

	Include all normal and abnormal uses of the system. Clearly distinguish between normal and abnormal test cases by identifying them as shown below.)	**Build**
1. Login/Password GUI: Each user must have a unique login/password pair that allows access to the system and sets their User Role within the system. ***Primary*: MIDN J. Gish** ***Backup*: MIDN W.T. Door**	1.1 User with correct login/password attempts to login. Expected result -> User is able to login, the correct User Role is associated with the login/password pair. (*normal*) 1.2 User attempts login with the wrong password. Expected result -> User is prevented from logging in. (*abnormal*) 1.3 User indicates they forgot their password. Expected result -> System emails a temporary password which user is required to change upon logging in. (*abnormal*) 1.4 …	**Build 1**
2. …	2.1 … 2.2 …	

Figure 2. Functional Requirements Trace Table

k. **Critical Component**: In this section, you will give your design for what you consider to be the most complex subset of your project. By complex, we mean the most difficult portion of your project for you to implement. Fair Warning: your reviewers will use this section to help gauge whether your project is challenging enough to be a capstone project, and also whether you have a good grasp of the real challenges of the project. Note that neither a High-Level Diagram such as that used earlier in this proposal nor UML Use Case diagrams are sufficient for your system design as these do not shed sufficient light on *how* you expect to implement your project.

1. **Critical Component Overview**. Give a short, written description of your Critical Component to help give context for the following subsections. Indicate why you consider this to be the most complex part of your capstone project.
2. **Critical Component's Requirements.** Identify the Functional Requirement(s) (by name and number) and give the acceptance test cases (by description and number) that correspond to the most complex subset of your project.

   a. You might find it helpful to copy just the applicable subset of your Functional Requirements Trace Table here.
   b. These functional requirements and test cases indicate *what* your Critical Component must do.

3. **Critical Component's Design Artifact(s)**. Give at least one design artifact (see "Design Artifacts" available from the course's Resources page) that represents your Critical Component's Design. The purpose of this design artifact is to convey to the reviewers how well you understand the most complex subset of your project.
   a. You may use any type of design artifact so long as the type of artifact is identified, the artifact is used correctly, and is detailed enough to convey *how* the component being designed will eventually be implemented.
   b. While you must have at least one, you are free to provide as many design artifacts as are needed to convey the complexities of your Critical Component.
   c. **Design Artifact Title.** Include a title for each design artifact that also identifies the type of design artifact used. Example: "Figure 1. Dataflow Diagram for the FuzzySmart Search Algorithm."
4. **Critical Component Discussion.** Refer to your design artifacts, and give a description of how you intend to implement your Critical Component. For example, "As shown in Figure 1, our FuzzySmart Search Algorithm first determines the Levenshtein distance by …"
5. If you have trouble determining which is the 'most' challenging part of your project (ie., you have multiple complex challenges) you are free to repeat parts 1..4 of this section as needed so that your project's complexities and corresponding design artifacts are properly conveyed.

l. **Risk Management:** Build on your Risk Analysis from Milestone 2, and, as specifically as possible, identify at least two real risks per sub-system that your team faces in the development of this project, and consider ways for mitigating these risks that can be objectively validated. Avoid trivial risks such as "procrastination on the part of the team" and instead focus on the real risks involved in developing your project. One approach to uncovering your risks is to consider your project from the perspective of, "If your performance (grade, pay, etc) is based only on measurable success, what are you *least* comfortable promising given your knowledge, ability to implement, time, etc."

Prepare a **Risk Management Plan** for your capstone project that includes:

1. **Risk Identification:** Develop a prioritized list of your risks ordered by your analysis of the risk's chance of occurring and the potential consequences of each risk. Risks that pose the greatest danger to your project must be dealt with first.

Use 1 to indicate the highest priority risk, 2 for the second highest priority risk, etc.

a) Consider risks both from your perspective (example, your technical knowledge) and well as the customer's perspective (example, will the critical GUIs be user-friendly).
b) Include both a Probability assessment (High, Medium, Low) of the likelihood that the risk being identified will actually occur as well as a Severity assessment (High, Medium, Low) of how catastrophic it will be to your system's development if the risk actually occurs.
c) A risk with a high probability of occurrence and a high level of severity should have a higher priority than a risk with a high probability of occurrence but a low level of severity.

2. **Risk Management Techniques**: Develop a plan for resolving each risk and indicate the current status of your attempts to mitigate the risk.

3. Figure 3 gives a sample **Risk Management Plan** related to a Functional Requirement that passwords be encrypted.

Priority	Risk	Risk Management Technique	Status
1 **Probability**: Low **Severity:** High	Unable to encrypt passwords that are embedded in Java non-serializable objects	Examine encryption techniques and either build our own encryption tool or use an open source solution such as JASYPT (JAva Simplified EncrYPTion)	2 person team (Smith and Jones) preparing a prototype for demonstration at Milestone 4.

Figure 3. Risk Management Plan

m. **Project Plan** & **Gantt Chart for your capstone project.** The above parts of your capstone proposal have laid out *what* you intend to do for your capstone project in a somewhat disjoint fashion. In this section you will bring it all together and clearly describe your *plan* for *how* you will accomplish your capstone project**.** Even though we are using Agile, we still need an initial plan!

1. **Project Plan.** In this section, give a written discussion of your plan for meeting the project requirements to include the following labeled subsections. Note that your Gantt Chart (see below) and Functional Requirements Trace Table are to be tightly connected to, and in agreement with, your project plan discussion.

a. **Major Tasks**. Identify by name each major task on your Gantt Chart, and give at least a one paragraph discussion for each task describing what each major task entails.

b. **Sequential and Concurrent development**. Include a discussion of which tasks must be worked on sequentially and which can be accomplished concurrently. Include the following areas in your discussion:

   i. **Sequential development**: Identify which parts of your system development *must* be done in a set order. For example your plan might include things like: *Preliminary Step 1: We will first work on getting just a basic shape (like a rectangle) to display in the Oculus environment. Preliminary Step 2: We will then develop 2D simulations of the game pieces, such as a rectangle that rotates based on user input. Once these preliminary steps are accomplished we will develop 3D renderings of each of the game pieces to include ...*

   ii. **Concurrent development:** Identify which parts of your system development (functional requirements/test cases) can be worked on concurrently with other parts of your system development. Your plan might include things like: *The User Interface will be developed concurrently with our team's design of our SmartMove Algorithm for selecting the next movement of game pieces when playing in "learning mode…"*

c. **Risk mitigation**. Include a discussion of your timeline for mitigating the risks identified in your risk management plan and why you will resolve some risks before others.

   i. For example your plan might include things like: *We have to develop an algorithm to encrypt passwords embedded in Java non-serializable objects before we can send the embedded data stream to the display field because…*

2. **Gantt Chart.** Prepare a new Gantt Chart for this milestone, as you likely have a much better understanding of your project proposal at this point. Note that the parts of your project that you have identified as being able to be developed concurrently must be shown as being worked on concurrently on your Gantt Chart. See Milestone 1 for the requirements for your Gantt Chart.

n. **Quality Assurance.** Oftentimes, teams will give one team member the task of doing a section of this proposal in isolation, and other team members the task of doing other sections perhaps in small groups. Breaking down the work in this way is to be expected on a team project.

   1. However, your team must ensure that your submitted proposal is well written, flows smoothly, uses consistent terminology, is grammatically correct, conveys the essence of your project, contains all the required sections and,

importantly, does not read like it was haphazardly assembled or prepared by different people.

2. While the Milestone Lead is responsible for setting internal timelines and making sure everyone knows what they are supposed to be working on, it will greatly help the quality assurance aspects if nearly complete rough drafts are ready for review by your quality assurance team member well in advance of the due date. See Milestone 0 for a discussion of the Milestone Lead's responsibilities.
3. **Quality Assurance team member:** On the last page of your capstone proposal, identify the Quality Assurance team member.
    a. Give the name of the team member assigned to quality assurance and who was tasked with reviewing the entire completed document.
    b. Ensure that any sections originally written by the quality assurance team member are also reviewed by a different team member (and give that reviewer's name as well).

o. **Customer Acknowledgement**: Same requirement as Milestone 0, you may include a previously signed enclosure (1).

Enclosure (1) from COMPSCIDEPTINST 1520.1E

Computer Science Department

Capstone Customer Acknowledgement

Required for teams whose customer is not a member of the USNA Computer Science Department

By signing below, the customer acknowledges that the project developed as part of this capstone coursework becomes the property of the DoD, and that the CS Department does not assume any responsibility for maintaining the software produced for the client. The client may use the software within the context of their USNA affiliation, and may not distribute it without approval from the USNA legal office.

Capstone Team Leader Name____

Capstone Project Title ____

Customer Name (printed) ____

Customer Contact Info (email/phone) ____

Customer Affiliation ____

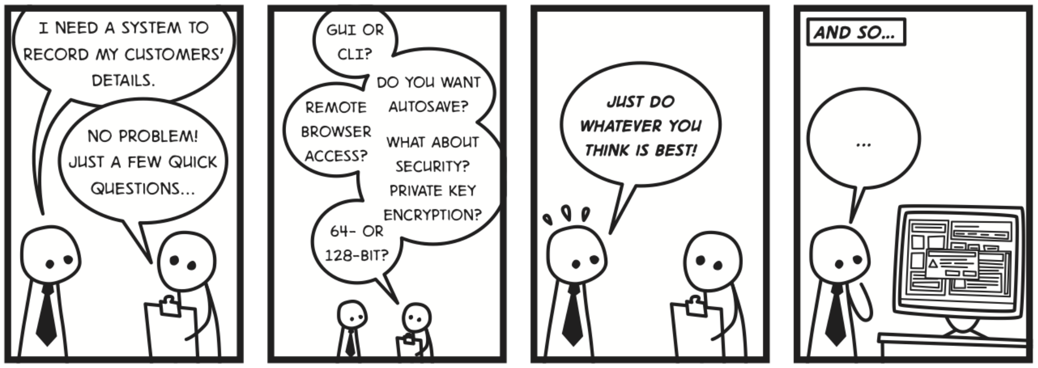
Customer Signature ____

Date___________________

-------------------------------- end of Enclosure (1) ---------------------------------

==

Part II. Targeted Design


I NEED A SYSTEM TO RECORD MY CUSTOMERS' DETAILS.
NO PROBLEM! JUST A FEW QUICK QUESTIONS...
GUI OR CLI?
DO YOU WANT AUTOSAVE?
REMOTE BROWSER ACCESS?
WHAT ABOUT SECURITY? PRIVATE KEY ENCRYPTION?
64- OR 128-BIT?
JUST DO WHATEVER YOU THINK IS BEST!
AND SO...
...

Due date: At the start of class as per the syllabus

Errata/Updates. Any errata or updates to this document will be dated and shown in red both as a summary below as well as in the contents of this milestone. Each team is responsible for checking (and delivering their milestone in compliance with) any changes indicated in this Errata/Updates section that are dated prior to the delivery.

Summary of Errata/Updates:

(none-yet).

See the course web page to determine during which period your Milestone will be presented (all team members in that section will participate in the presentation). Team members that are not in the same section as the scheduled milestone presentation will earn the same grade as the one earned during the presentation period, and are expected to contribute their fair share to the development of the presentation's materials.

Milestone _______ Team _______
(the above filled in by the team)

Deliverables checklist (see below for more info on each of these items). Be prepared to present the below, in order, during your milestone delivery.

- Checklist (a copy of this deliverables checklist sheet)
- Customer's Evaluation Cover Sheet completed by Customer (and Tech Advisor), or, copies of emails showing your attempts to contact them at least two days prior
- Concise Project Overview
- Customer meetings summaries and action items with lead mid for each item id'd
- 4 Targeted Acceptance Test Cases for this milestone and justifications
- Design Artifact(s) for the 4 Targeted Acceptance Test Cases
- Note: There is no system demo required for this milestone.
- Paper copy of presentation ready to turn in at start of period
- Email presentation slides to instructor with subj: Group X, Milestone Z as the subject line

Milestone Deliverables (paper copies turned in to your instructor *prior* to beginning your oral presentation)**.** An oral presentation supported with PowerPoint slides is required for this portion of the milestone to be delivered as per the table on the course web page. Unlike Part I, Part II is required for all students currently enrolled in IC470. Turn in paper copies of your slides to your instructor *prior* to beginning your oral presentation.

1. **Arrange to meet with your Customer**. Same requirements as Milestone 1

2. **Admin.** Same requirements as Milestone 1.

3. **Proj Mgnt** (Project Management)

    a. **Concise Project Overview**. Same requirements as Milestone 1.

    b. **Customer Meeting Summaries.** Same requirements as Milestone 1

    c. **4 Targeted Acceptance Test Cases**.
        a. Customer involvement alert => Meet with your Customer and determine which 4 acceptance test cases are "appropriate" for you to focus on for the design aspects of the milestone (heads-up: these will be the same test cases that you will later implement for Milestone 4).
            i. By "appropriate" we mean functionality that your team anticipates will be useful to undertake at this early point in your capstone implementation in order to gain understanding of a complex-to-design and implement portion of your project, but, at the same time, comprise functionality that you will be able to accomplish by the delivery time of Milestone 4 (see the course webpage).
            ii. Note that the 4 targeted acceptance test cases will likely not be the same as those identified in the "Design of Most Complex Subset of Project" portion of your formal capstone proposal (unless you are prepared to commit to implementing that subset by Milestone 4).
            iii. The test cases for this milestone do not have to all come from the same functional requirement.

        b. **Present your 4 targeted acceptance test cases** (include their corresponding Functional Requirement(s)).
            i. **Justification.** Present a justification as to why you selected the particular acceptance test plan test cases.
                1. Teams picking easy to implement test cases, such as not being able to login due to the wrong password, will be docked for missing the intent of this portion of the milestone.

4. **Modeling.**

a. **Present your design artifact(s)** that together give a complete design for your 4 Targeted Acceptance Test Cases.
    1. You may use any of the artifacts described in "Design Artifacts" available from the course's Resources page.
    2. Include a title for each design artifact that also identifies the type of design artifact used. Example: "Figure 1. Pseudocode for the FuzzySmart Search Algorithm"

b. Keep in mind that you do not need to implement these acceptance test cases just yet. The focus of this milestone (Milestone 3 – Part II) is your design for your targeted acceptance test cases. The implementation of these 4 test cases will be deliverable by Milestone 4.

5. **Coding** (Implementation). N/A for this milestone.

6. **Testing.** N/A for this milestone.

7. **Training.** N/A for this milestone.

Notes:

a. Each team is to be fully ready to go at the beginning of the presentation period to include handing in a paper copy of all slides, source code, and GUI screen shots used in the presentation/software demonstration as well as the documentation. Also, each team is to turn in a copy of the oral-presentation grading sheet (available from the course web page), with your team members' names filled in, at the *start* of the period *prior* to beginning your oral presentation.

b. Any team not ready to hand in their paper copies of the above, or to deliver their presentation/demonstration when called upon, will have 10 points deducted from their presentation grade and will go to the end of the presentation cycle for that day. Presentations not delivered during class on the due date will earn a grade of zero, but will still have to be completed and turned in to receive a passing grade for the course.

c. Each team member must participate in all portions of the term project, including *each* oral presentation.