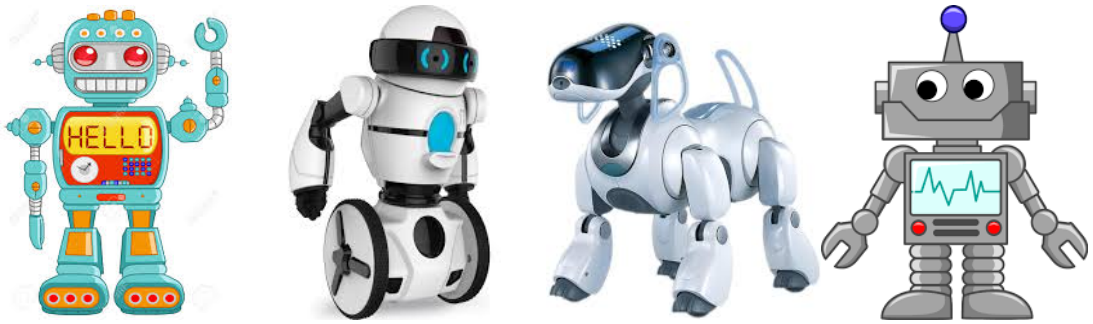
CPSC 362
Foundations of Software Engineering
Homework Outline
Spring 2017
[Version: 1/10/2017]

HW1 Outline

(Tentative – this HW outline can be changed later. Please check any updates.)

The Target System

Your organization is required to build a robot system (titled "*Fullerton_Robot (fRobot)*"). The *fRobot* project is to build a machine (both hardware and software) that automatically behaves as pre-programmed. *fRobot* can be programmed (1) to speak a few words and a few statements (less than 10 words/statements) (and show its text/script/sign-language on the screen), (2) to listen to your order (by either voice recognition or programmable buttons), (3) to move around the designated places (by using censors, camera to detect surroundings), and (4) to hold/release objects by their hands (less than 10 pounds). Such programs (for what to speak, what to listen (or programmable buttons), where to move, what to hold) can be done by users (end-users, sellers, manufacturers) in any computers (with the given programming tools), and can be installed by USB or by wireless transmission. Your team is to build the basic/default application software (to install apps, program buttons, speak, move, and hold/release objects) running on the *fRobot*. Your team must collaborate with other teams who will build the *fRobot* programming tools/environment and the *fRobot* hardware.



Competitors [Source: Web]

What To Do

In HW1, You need to apply the principles (tasks/activities/models) you learned from the textbook (Ch.7 – Ch.11), and practice (perform and document) ***requirements analysis*** for *fRobot*.

Functional requirements must include the functions that are given in the target system description (above).
In addition, each team may extend functions to include their own ideas.

Understand, analyze, design, and code the system as realistic as possible, however, do not try to tackle too large set of requirements (make the problem simple and solvable in this class).

- **HW1** will perform the requirements analysis activities (and its documentation) using CASE tools.
- **HW2** will complete detailed design (its documentation) and perform some partial (not complete) coding and testing (or review/inspection) using CASE tools.
- HW2 is an extension (a continuing work) of HW1. HW2 contents will include all contents you build for HW1.
- Presentations will be required in the lab practice and HW discussions. In presentation the instructor will check your understanding of the associated course materials and appropriateness of your presentation.
- Reading Ch.7 - 11 will be helpful for HW1, and reading Ch.12 – 18 (and Ch.19 – 23) will be helpful for HW2. Document all activities in terms of models (and suggested practices) shown in the textbook.
- Appendix 1, Appendix 2 and references (listed in the syllabus) provide information for the modeling language, UML, Object-Oriented Concepts, and CASE tools (for modeling, coding, testing, configuration management, project management, and etc.).

Team Charter and Team Evaluation

Build your team to collaborate for doing both HW1 and HW2. (Each team = minimum 2 or maximum 6 members)

Construct and submit your **Team Charter** (using the given template, however, fill it in with your own words).

Also review the given template for **Team Evaluation** now, so that you can know how to evaluate each other at the end of each HW. You must evaluate using the Evaluation Criteria that you construct now in your Team Charter – later in Team Evaluation, do not use other personal/subjective matters (other than Evaluation Criteria that you build now) (find/fill-in the last section titled by "Evaluation Criteria" in your Team Charter).

Functional Requirements (FRs)

List the most important 10 FRs.

Start with 10 FRs in the analysis phase.

Prioritize FRs to focus on.

Non-Functional Requirements (NFRs)

List the most important 5 NFRs.

Start with at least 5 NFRs, and then finally prioritize/choose/focus on 2 NFRs later.

NFRs include both Quality Attributes and Design Constraints.

Minimum process requirements

- To make your work simple, use the (1-pass/customized) Waterfall Process (to learn a simple software development process, but to practice it completely/entirely).

- In the HW report, have a separate section that describes each phase of your (1-pass Waterfall) process (Figure 2.3). (See the format of HW Report below, and find why the HW2 report format is organized in that way.)

- **Requirements Analysis**: Perform and document requirements analysis. (This part must be completed in **HW1**.)
    - Perform all tasks shown in Ch.7-11.
        - See Ch.7-11, and Appendix 1 and 2.
    - Minimum requirements (minimum work-products):
        - Functional requirements (list at least 10 FRs, and focus on mainly 3-4 FRs)
        - Non-functional requirements (list at least 5 NFRs, prioritize NFRs, focus on 2 NFRs)
        - (Textual) Use cases (UCs): Construct at least 10 UCs total. Construct at least 1 UC for each FR (Ch.8, Ch.9)
        - At least 1 UC Diagram (UCD) (Figure 8.2, Figure 9.4, Ch.8-9)
        - At least 1 (Analysis) Class Diagram (ACD) (that show all 'candidate/potential' classes that can be extracted from FRs/UCs, and their relationships) (Ch.8-10, Figure 8.4, Figure 10.1, Figure 10.2, Figure 10.4-7).
        - At least 3 Class-Responsibility-Collaborator (CRC) cards (for 3 candidate classes; must show at least 3 responsibilities for each class; must have at least 1 collaborator for a class) (Figure 10.3)
        - At least 1 Activity Diagram (AD) (for the most appropriate FR/UC/scenario) (Figure 8.3, Figure 9.5, Figure 11.6)
        - At least 1 Swim-Lane Diagram (SLD) (for the most appropriate FR/UC/scenario) (Figure 9.6)
        - At least 1 State Diagram (SD) (for the most appropriate FR/UC/scenario) (Figure 11.1)
        - At least 1 (Analysis) Sequence Diagram (ASD) (for showing interactions between actors and the system (and external systems only if necessary) as a black-box; these interactions can be identified from FRs/UCs/scenarios) (Figure 11.2)
        - Do not follow both Figure 11.3 and 11.4 - they are not good examples for both sequence and class diagrams)

- Design: Perform and document design activities (only as much as you can at this time). This is not required for HW1, and will be completed in **HW2** (to be required later in HW2).
    - Perform all tasks defined in Ch.12-18.
    - Design (to be done in HW2) must be tightly related to the requirements analysis models (constructed in HW1).
    - Here we emphasize on translating analysis models into design models.

- Code: This part will be completed in **HW2** (to be required in HW2.)
  - Only prototype coding for most part of your design will be required (class names and method names only; no body implementation). However, coding must be tightly related to the design models. Here we emphasize on translating design into code (rather than coding to a specific system).
  - Complete a coding for a part of the system that you select. Testing must be performed for this part also.

- Test: This part will be completed in **HW2** (to be required in HW2.)
  - Perform both inspection and testing for the part of completed code.
  - Document what you have done for both inspection and testing (describe what/how you have done/learned; document with some screen-snapshots for testing).

- CASE Tools: This part can be completed in either **HW1** or **HW2** (or in both HWs)
  - For most part, hand-writing/drawing is fine. However, use of at least one CASE tool is required.
  - Use at least one CASE tool. Hand-writing/drawing is fine. However, use of one or more CASE tool is recommended.
  - Document what you have done to use CASE tools (describe what/how you have used/learned; document with some screen-snapshots).

- No other activities are required (unless it is 'explicitly required' in the HW Outlines or in class/labs).

Traceability

- Showing bi-directional traceability is required.
- Construct 'Traceability Matrix' showing relations between artifacts (within 1-2 pages).
  - Bi-directional traceability between requirements models.
    - (e.g.) A traceability matrix that shows relations between FRs and use cases.
  - Bi-directional traceability between requirements and design models
  - Bi-directional traceability between design models and (prototype) code.
- Construct 'Traceability Diagram' (1 page) that shows relations among all artifacts you construct. This helps you understand how all artifacts are related.
- See both Figure 12.1 and Figure 12.4 to learn relations among artifacts.

[**HW Report Format**] (Required Work Products and Document)

- File format: (Open-) office file format (DOC or any open-office file format), PDF, RTF or standard graphic file formats. Space: single/double, Fonts: Times/Arial, Font size: 11-12.
- Document all activities (and their work-products/artifacts) in terms of the phases of the Waterfall Process (Figure 4.1, Figure 12.1, Figure 12.4).

- **The contents/format of the HW report** must be (at minimum) like the following:
- A Cover Page
  - HW Title, team title, names of all (only participating) members (and email addresses, signatures), submission date, course title, instructor name, …
- Revision History (e.g., table format)
  - Who did what, when, …
- Table of Contents (chapter titles, page numbers)

- Communication (within 1 page) (What is this? See Figure 4.1)
  - Project initiation (5-10 lines)
    - A short description of the application to build (5-10 lines)
  - Requirements gathering (5-10 lines): what you did for this?
- Planning (within 1 page)
  - Estimating, scheduling, tracking (all in 1 page)

- Modeling: Requirements Analysis (See Ch.7-11, and Appendix 1 & 2)
  - Document activities. Produce work products related to this phase. Perform all activities described in the textbook. (Perform all activities to construct minimum work-products described above.)

- Modeling: Design (See Ch.12-18)
  - Document activities. Produce work products related to this phase.
  - Fill in this part in **HW2**. Now you only type "To be done in HW2".

- Construction: Code
  - Prototype code (only class names and method names – tightly related to design models)
  - Complete coding for a part you select.
  - Document activities. Produce work products related to this phase.
  - Fill in this part in **HW2**. Now you only type "To be done in HW2".

- Construction: Test
  - Document for both inspection and testing for the completed code.
  - Fill in this part in **HW2**. Now you only type "To be done in HW2".

- CASE Tools
  - Document activities for using CASE tools.
  - Fill in this part in either (both) **HW1** or (and) **HW2**.

- Deployment
  - Still need to have this chapter title and only say "No activity for deployment, deliver, support, and feedback is required in this HW."

- References (list references here, and cite them in appropriate places in the report)

- **Team Charter** (using the given template)

    - Construct and submit your team charter using the given template. Clearly discuss/specify how to evaluate team members (in the section of "evaluation criteria"). This helps you understand in advance how to evaluate members (and how you will be evaluated by members).

- **Team Evaluation** (using the given template)
    - Construct and submit your team evaluation using the given template. 'Fairly' evaluate members with objective evidence (by the "evaluation criteria" you discussed in your team charter).
    - (Note) To get a full credit for your Team Evaluation, you must put you best effort on HW. If your HW score is below B (80%), it will be prorated by your HW score. (e.g., 79% or below in HW will not get 100% credit in Team Evaluation.)
    - (Note) If you did not participate in HW properly, you will 'not' get full credits for both HW score and collaboration score 'as is'. Both scores can be 'zero' if you do not participate. No score for no/improper work.

[The End of the **HW Report Format**]

Submission

- By the due date (before the class hour), submit/upload a soft-copy of your HW report (in a DOC/PDF file that includes all work-products you produced).

Presentation

- Bring your files in a USB drive for presentation (or download from the Web).
- No separate presentation material is required. (Just project your report/source file 'as is', when presenting.)

No Plagiarism

- No copy or cut&paste from other sources is allowed for this HW report. Do not use other resources as the basis for your HW. Your HW report must be 100% unique (no copy of even a sentence or a diagram from other sources is allowed). All members will be responsible with plagiarism (even though only one member violates plagiarism). The worse score (F, or 0 depending on seriousness) will be given to the cases of cheating (or plagiarism). Start HW early and do it creatively. Prepare and ask questions in the discussion sessions (or Q&A, office hours) for HWs.

Grading of HW

Your homework will be evaluated based on the following factors: quality of report (rather than quantity), organization of report (required format), uniqueness and consistency of contents in the report, tight accordance with the subjects learned/required from this class (first you have to discuss it based on only what you learned from the class, and you may discuss more learned from outside only if you want – this is for fair grading with objective evidence), an overall team work, participation (attendance and/or discussions) and collaboration (self-evaluation), the logs and the meeting records

(submitted only if requested), the group/individual presentation, and the professional attitude toward team work.

The homework will be graded as group, but self-evaluation will be used to grade collaboration individually. If your contribution to the team is less than B (80%), or if your HW scores is less than B (80%), you will not get a full credit for the collaboration score.

A few exam questions will be asked to filter out understanding of homework's contents. Exam questions related to homework will individually measure understanding of each member. Students who put more effort into homework (who understand all parts of HW thoroughly) are usually more successful in those exam questions.

Grading Examples:
10/10: Almost perfect (or, with minor error) (A+ for this HW)
9/10: Good but need improvement to enhance quality (A)
8/10: acceptable but need critical improvement (B)
7/10: require serious improvement (C)
1-6/10: seriously deficient; need more serious work; unacceptable (D or F)

Be fair and objective: You must evaluate your team members fairly and objectively. Carefully build evaluation criteria in the Team Charter in advance, and use the evaluation criteria only in your evaluation. Do not involve any personal and subjective opinions in evaluation. Professionally behave.

Perform at your best (you must show proper effort): An unsuccessful team (below B) may not get a full credit for collaboration scores. If your HW score is below B (8/10), you may not get a full credit for the collaboration score (which is factored by self-evaluation score). For example, if your HW score is 6, and your self-evaluation is 100% (10), you may not get full 10 for the collaboration score and may get 6 (= [10 * 6/10]) for the collaboration score at maximum.

You may not get 100% (A) even though you work hard (working hard does not mean quality product). However, if you give up working hard (targeting B), you may not even pass (may get C or less if your target fails). Show your best effort to successfully pass (for A or B at least).

Be a good citizen in your team: An ill-collaborated member may not get a full credit for the HW score. For example, if you get 10% (1/10) in your evaluation score by team members, you may get 10% of your HW score at maximum. Make and keep your commitment. Let members fully understand your situation if anything happened / scheduled, and show your extra effort to make them up later or in advance. Put your best effort to make your stakeholders (members, customers, etc.) satisfy. You must accept the evaluation criteria, once it is defined in your Team Charter.

[The End of HW1 Outline]

[The HW2 Outline: see the next page]

HW2 Outline

(*Tentative – this outline can be changed later. Please check any updates.*)
HW2 continues software development work that you have done in HW1.
Especially, HW2 will complete (architectural and detailed) design activities, coding and testing, based on what you have done (requirements analysis) in HW1. Follow all tasks shown in the textbook, Ch.12-18. Reading Ch.12-18 will help you understand design techniques and work-products. Reading Ch.19 (Quality Concepts) is also helpful to understand non-functional requirements and qualities. See also Appendix 1 and 2 to learn UML and Object-Oriented Concepts.

What To Do

- In addition to the artifacts you produced (and submitted) in HW1, apply design techniques and produce work-products shown in Ch.12-18. Complete documentation (continuously fill in the report that you built for HW1):
  - Understand the Design Concepts (Ch.12)
  - Perform architectural design using techniques and architectural styles shown in Ch.13.
  - Perform some detailed design using techniques and work-products shown in Ch.14-18.
  - Perform coding and testing. No complete coding is required – Code only the signatures of classes and function names (no code for function bodies) that match with your design models – Fill in the code body only for the most important part (e.g., a main driver plus a few representative functions). Perform testing for the completed code.
- Traceability will be checked in grading. Build (one or more) *traceability matrix* that show(s) how analysis models are transformed to design models in your case (e.g., a traceability matrix that shows relations between one model to another model, and so on.). Build (one page) *traceability diagram* that shows all analysis models and design models, and their relationships.

Minimum Requirements (Minimum Work-Products)

- In addition to what you have in HW1, you must construct the following work-products. (You can refine/update contents for HW1 if you need – e.g., for traceability, errors found, etc.)
- For Architectural Design (Ch.13)
  - Discuss how to select the architecture (Ch.13.3, Figure 13.1-4)
    - Consider possible alternative architectural styles (Ch.13, Ch.13.3), discuss pros and cons of (2 or 3) candidate architectural styles, and select your architecture based on those architectural styles. Draw two levels of diagrams that specify your architecture you choose finally. The 1st level architecture diagram shows the architectural context diagram (Figure 13.5, 13.6) that shows the target system, external systems, input and outputs, at the high level. In the 2nd level architecture diagram, show components and interfaces of the system in detail. Give the keys that describe symbols/notations you use in the

diagram. Discuss also your rationale under each diagram (within 1 page).

- 1 Architectural Context Diagram (ACD) (Figure 13.5, 13.6) that shows the target system as a black-box in the middle and external systems (connecting to the system), input and output for the system.
- 1 Overall Architectural Structure (Figure 13.8, 13.9) that must show potential components (even though its component classes are not clear yet – that will be defined later) inside of the system and their interactions/communications (by which protocols/methods). Be careful, the textbook's examples do not specify their architectural styles well. But you must choose (an) architectural style(s) first, and show (instantiate from the chosen architectural style) your architecture that was drawn from architectural styles/patterns in the textbook (Ch.13).

- For Component-Level Design (Ch.14)
  - 1 Class Diagram that show all design classes (that you identified/updated from the analysis class diagram and requirements/use cases/CRCs), attributes and methods of classes, and relationships among classes (Figure 14.1, 14.3).
  - At least 2 Interaction Diagrams (Sequence Diagrams or Collaboration / Communication Diagrams) (Figure 14.6). Each ID/SD/CD must show some interactions (at least 3 message passing) among a few (at least 3) classes. Such interaction must be initiated by a system operation (service) identified from UCs. CRCs can be additionally referred, but mainly (scenario in) UCs must be used to identify a certain system operation (service) that is expected by/from an actor. Such system operation (service) must trigger (start) an ID (Ideally, each system operation/service shown in the Analysis Sequence Diagram must be a starting point in each ID – In HW2, choose at least two system operations/services shown in ASD, and draw at least 2 IDs).
  - 1 Refined Architecture Structure Diagram (from the above Overall Architecture Structure Diagram) that shows how components are arranged (based on the architecture that you defined in the architectural design), how components are interconnected/communicated (by which protocols/methods), how classes (you identified before) are assigned to the components.
  - At least 1 Activity Diagrams (AD) (Figure 14.8) (more specifically detail than analysis activity diagrams you built before)
  - At least 1 State Chart (State Transition Diagram) (Figure 14.9)) (more specifically detail than analysis activity diagrams you built before)
- For User Interface Design (Ch.15)
  - 1 Swim-Lane Diagram (Figure 15.2)
  - 1 Preliminary Screen Layout (Figure 15.3)
  - 1 Diagram for mapping user objectives into the interface structure (menu/navigation) (Figure 15.4)
- For Pattern-Based Design (Ch.16)
  - Pickup one design problem to solve, survey and find 2-3 candidate *design patterns*, select a design pattern among candidate patterns, and describe how to use such a pattern to solve the problem. Describe all within 1-2 pages.
  - 1 Pattern-organizing table (only) for the (2-3) patterns you used (Figure 16.2)

- For WebApp Design (Ch.17)
    - Draw a diagram, "A Design Pyramid for WebApps" or "A Design Pyramid for MobileApps" (similar to Figure 17.2). Explain what you did for each layer of the diagram for the user interface design (all within 1-2 pages).
    - You may also assume that some components of the mobile application (MobileApp) are for the Web running on mobile devices (WebApp).
    - Draw a diagram for the Content Architecture (like Figure 17.4 – 17.7). Explain which structure(s) you used (Linear, Grid, Hierarchical, Network, or combinations) (all within 1-2 pages).
    - Draw a diagram of the MVC Architecture of your system (when you assume that you use the MVC architecture) (Figure 17.8). Describe what M, V, and C mean with your system design (all within 1-2 pages).
    - Create a diagram for Navigation Semantics Units (NSUs) (like Figure 17.9). Show which use case (scenario) is used to draw it (all within 1-2 pages).
- For MobileApp Design (Ch.18) (all within 1-2 page(s))
    - Review 18.3 Mobile App Design – Best Practices, and discuss what you must consider for each item in the bulleted list (10 bullets total).
    - Review 18.4 Mobility Environments (and their INFO boxes also), and discuss what you can consider for the environment in general for your case.
- By reading Ch.19 (Quality Concepts), you may learn how to consider quality issues in design and test.
    - Think whether your non-functional requirements were identified properly, and used in architectural and detailed design. If necessary, refine associated work-products. (Just for your information; no specific writing for this part is required unless you refine artifacts; the idea you learn from this step is assumed to be incorporated in your analysis and design)
- For Construction: Code
    - Only prototype coding is required (class names and method names only; no body implementation). However, coding must be tightly related to the design models (such as design class diagrams, interaction / sequence / communication / collaboration diagrams, and so on). Consider traceability.
    - Complete coding at least a part (e.g., a main driver with a few functions; at least1 or more classes).

- For Construction: Test
    - Perform testing for the part you constructed (complete coding and testing are required even though it can be done for a partial system; for other parts, you can build only prototype coding).

The Contents/Format of the HW2 Report

- The overall contents/format
    - Use the same format with (the extended/updated contents of) the HW1 report.
    - Reference list (and give citations in appropriate places)
- Team Charter
- Team Evaluation (signed by all members).

- Even though a student did not participate (but once assigned to your team), he/she must be explicitly listed/graded/commented in the Team Evaluation.
- Even though you do not agree with evaluations of others, you must give your evaluation scores (for all members including you, based on the objective evaluation criteria shown in the Team Charter), you must sign, and you may leave your opinions (why you do not agree) in the comment box in the form.

[The End of HW2 Outline]