CSE269 CS Design Laboratory – Project 3

DUE: Given on Part-by-Part Basis

Each team is to submit ONE assignment with the team name and the names of all team members. Clearly identify which team member did what portions of the submitted assignment.

Part I - Due on February 14, 2005, by 3pm

Analysis is an important part of any project. In the first part of this second assignment you are to analyze your written set of specifications against the software qualities (see Chapter 2 of Ghezzi or other Software Engineering textbook). Focus on the following qualities:

- Performance
- Portability
- Understandability
- Productivity
- Reliability
- User Friendliness
- Robustness
- Repairability
- Reusability
- Maintainability
- Interoperability
- Evolvability

Each team member must choose two of these qualities and discuss:

- Importance and relevance of the quality to your problem/domain.
- The attainment/incorporation of the quality in your specification and eventually in your design and implementation.

For each quality, limit yourself to 1/2 page of 12pt, word-processed, double spaced text.

Part II - Due on February 14, 2005, by 3pm

The second part of this assignment is to modify and expand your specification in light of the discussions of Part I and any comments that were provided on your handed in specification. Coincident with this part is Part III given below that begins to add the detail for your project solution. You can add up to three more pages of text (10 total) to your specification.

Part III - Due on February 21, 2005, by 3pm

The next step in completing the specifications is to more clearly define the control and data organization of the project. You must use a chief-programmer team approach, where a different team member serves in the role of chief-programmer for Tasks A through D given below. For each part, a second team member should provide backup and input. All team members are expected to be familiar with all four tasks and be able to converse about each task during the weekly meeting sessions. All teams must do Tasks A, B, C, and E. Task D is to be done only if deemed relevant by a consensus decision by the team. Tasks A through D are intended to evenly distribute the
work required by the team so that each team member can demonstrate an individual contribution as part of this assignment. Task E should be a joint team effort by all members.

The output of Part III should be the specification from Part II, expanded and refined to include the various modeling techniques from Tasks A through D.

Task A

Construct Entity-Relationship (ER) diagrams for the data showing the conceptual structure of the data and its interrelations (see Ghezzi, section 5.6.1, or another software engineering textbook). You must use the ER specification capabilities of Together CC in constructing your diagram. Utilize the documentation generation capabilities of Together CC to generate HTML documentation for your ER diagrams. Make sure that you describe the attributes of each entity including types and acceptable ranges. Create a twopage explanation of the selection of your entities and relationships. Include a discussion of any design alternatives and their evaluation. Integrate this documentation with the Together CC html documentation.

Task B

For Task B, you have the two following alternatives:

- Construct Data-Flow Diagrams (DFD) for the system (see Section 5.5.1, Ghezzi or another software engineering textbook). Your DFD diagrams should follow the graphical conventions in the book. DFDs may be both high-level and detailed, depending on your focus and approach, i.e., a hierarchy of DFDs. Create a one page explanation of the functions and data stores used for each DFD, and include a discussion of any design alternatives (different function and data store alternatives) and their evaluation.

- Construct UML Sequence Diagrams (SD) to show the sequential flow between your various system objects and components. Unlike DFDs, SDs concentrate more on the flow that occurs over time, whereas DFDs are more discrete in nature. See Chapter 18 of the UML textbook for a discussion and explanation of SDs.

Regardless of your choice, you must fully document your output, and integrate your documentation with the documentation produced by Together CC.

Task C

Construct Finite State Machines (FSM) to show the general control structure of the system (see Section 5.5.2, Ghezzi, or another software engineering textbook). Clearly identify each state and the transitions between states. You may want to create a hierarchy of FSMs to help manage the complexity. Each FSM should include an explanation of the states and the arcs between the states and include a discussion of any design alternatives and their evaluation. FSMs are supported in Together CC using statechart diagrams (state machines in UML, see Chapter 21 of UML textbook). You must use the statechart diagram capabilities of Together CC in constructing your FSMs. Utilize the documentation generation capabilities of Together CC to generate HTML documentation for your FSMs.
Task D

For Task D, you have the two following alternatives:

- Construct Petri Nets (PNs) to show timing related actions that compete for resources of the system (see Section 5.5.3, Ghezzi or another software engineering textbook). Clearly identify the places and transitions, with arrows indicating the flow. Each PN should include an explanation of its structure.

- Construct UML Activity Diagrams (ADs) to show structural components and flow between the various states. Like PNs, ADs support both forks and joins to indicate actions that can occur in parallel. See Chapter 19 of the UML textbook for a discussion and explanation of ADs.

Regardless of your choice, you must fully document your output, and integrate your documentation with the documentation produced by Together CC.

Task E

Create a three page document discussing and explaining the relationship between Tasks A through D. You should discuss how ERs, DFDs or SDs, FSM, and PNs or ADs, relate to one another and you should discuss the combined contribution of these four techniques in terms of correctness and robustness.

Walkthroughs - February 21 and February 28, 2005, 3pm

There are two walkthroughs scheduled. All team members must participate and present.