Project #13

18-649 Embedded System Engineering

Electrical & Computer ENGINEERING

Carnegie Mellon

Note: Course slides shamelessly stolen from lecture

All course notes © Copyright 2006-2013, Philip Koopman, All Rights Reserved

New Simulator

- **Mostly cosmetic fixes**
  - **Real Time Rate**
  - **OSX GUI**

- **Substantive Change**
  - **Passenger behavior – Fixes a bug where “overweightExitAction at non start floor” exception occurs**
  - **Non exception behavior – slight variation because of the fix**

What does this mean for groups?

- **If the old simulator 4.8 works for you, stick with it.**
  - **If not you can use 4.8.1**
- **All conflicts between sims resolve in your favor!**
- **If we see an “overweightExitAction at non start floor” exception we discard it!**
  - Rerun with a new seed.
  - If that fails, run with new simulator
- **Applies to Demos and Project 13**

Project 13 Grading

	Points	Breakdown	Awar
Runtime Monitoring			
AUTOMATED: Execute the grading monitor on the acceptance tests listed in the project 13 writeup. If they pass with no warnings, give full credit. If fewer than 4 warnings are issued, give half credit	15	B	
Testing			

- **Make sure your project passes *YOUR* runtime monitoring.**

Project 13 Grading

pass with no warnings, give full credit. If fewer than 4 warnings are issued, give half credit	15	B	
Testing			
AUTOMATED: To check testing, copy the group's code from their portfolio into a clean copy of the simulator framework and make sure the code will compile.	0	N/A	
Is the Unit Test Log complete and up to date (all controller modules listed, all tests passing, input and output files properly linked.	7.5	B	
AUTOMATED: Execute the unit tests using the simulator assembled in the design portfolio grading. (Note that this step requires a valid unit_tests.txt summary file). All tests must pass (0 failed assertions), and all tests listed in the unit test log must be listed in the unit_tests.txt file. If the simulator will not compile, award no credit.	7.5	B	
Is the Integration Test Log complete and up to date? "Complete" means all sequence diagrams are tested (up to a total of 20) and include all the original sequence diagrams (1A, 1B, 1C, 2A, 2B, 3A, 4A, 5A, 5B, 6, 7A, 7B, 7C, 8A, 9A). "Up to date" means all tests passing, input and output files properly linked.	7.5	B	
AUTOMATED: Execute the integration tests using the simulator assembled in the design portfolio grading. (Note that this step requires a valid integration_tests.txt summary file). All tests must pass (0 failed assertions), and all tests listed in the integration test log must be listed in the integration_tests.txt file. If the simulator will not compile, award no credit.	7.5	B	
Is the Acceptance Test Log complete and up to date? All acceptance test files listed in Project 13 writeup must be passing. Each entry must be complete (all fields filled out and input and output files properly linked). Any test that does not pass must be documented to describe the problem that causes the test to fail.	5	B	
AUTOMATED: Execute run all acceptance tests from the project writeup and the undisclosed acceptance tests using three arbitrary random seeds. The test must deliver all passengers.	25	B	
Complete and Consistent Portfolio			

- **Make sure your project passes all unit and integration test**
- **Make sure your project delivers all the passengers for the acceptance test that you have**
- **The undisclosed tests are the Demo tests**

Project 13 Grading

tests using three arbitrary random seeds. The test must deliver all passengers.	25	B	
Complete and Consistent Portfolio			
This value is computed from the average in the "End-to-End" sheet.	90	B	
Improvements Log			

Project 13 End-to-End Traceability and Complete Design Por		
Blue cells are to be graded by the grading TA and double-checked by the head TA. Each item is given a linear ranking from 0 to 4, with 0 being "completely ignored the requirement" and 4 being "executed the requirement perfectly".		
Portfolio	**Score**	**Notes**
The portfolio conforms to the guidelines provided in the portfolio layout page on the course website.		

- **Consistent Portfolio**

Project 13 Grading

requirement perfectly".		
Portfolio	**Score**	**No**
The portfolio conforms to the guidelines provided in the portfolio layout page on the course website, namely: the portfolio is composed of vanilla HTML documents (except where other formats are specifically required), all hyperlinks point to the correct document, and all inline images are present and readable.		
A random sampling of all portfolio files (including test inputs and outputs and code files) contain proper headers listing the group number, course and semester, and all group members' names and andrew IDs.		
All the required design project artifacts are present (Architecture, use cases, scenarios, sequence diagrams, requirements, statecharts, code modules, unit, integration, and acceptancet test files and logs)		
For the remaining items, choose one module (e.g. DoorControl) and perform an end-to-end check on the following list of c		
Module(s) checked:		

- **http://www.ece.cmu.edu/~ece649/project/portfolio/portfolio_layout.html**
- **Make sure you've been running your tests with –head**

Project 13 Grading

Module(s) checked:	
Sequence Diagrams - choose two SD and verify that the following items are correct. The quick reference document has a list of messages with correct/network framework status and replication.	
All network messages are black arrows using the mMessage notation with correct replication.	
All framework messages are blue arrows with framework notation (no 'm') and correct replication	
Sequence Diagrams to Requirements Traceability	

- **Has to be consistent with your network schedule**
  - No messages in SDs that you removed
- **And the fixed replication of the elevator,**
  - e.g. no atFloor(2, FRONT)

Project 13 Grading

Sequence Diagrams to Requirements Traceability		
Use the SD-to-Reqs traceability table to identify the sequence diagram arcs that are traced to the requirements. Is each SD arc relevant to the requirement it is traced to? A relevant arc is one that pertains to either the trigger conditions or values set in the requirement.		
Every requirement traces to at least one sequence diagram arc.		
Requirements to Contstraints Traceability - Check the following criteria for each requirement		
Every constraint and requirement is listed in the table.		
The entries with X's substantially address the constraint.		
No entry with a ~ directly contradicts the constraint, or directly meets the constraint (then it should have an X instead of a ~)		
Requirements - Check the following criteria for each requirement		

- **Come on, you guys. You know traceability.**

Project 13 Grading

X instead of a ~)	
Requirements - Check the following criteria for each requirement	
Requirement has the form IF <trigger condition> THEN <value SHALL/SHOULD be set>.	
Each requirement is numbered, and requirements that set multiple values have a unique number for each SHALL/SHOULD verb	
All trigger conditions and values set conform to the message / framework notation (see the quick reference document for a list of messages)	
All trigger conditions are based on defined state variables or on messages/framework values in the input interface of the controller.	
All values set in the reqs are defined state variables or messages/framework values in the output interface of the controller.	
Statecharts	
The guard conditions for each state are mutually exclusive.	

- **Make sure these are up to date.**
  - New messages in and out
  - Numbers are up to date

Project 13 Grading

of the controller.	
Statecharts	
The guard conditions for each state are mutually exclusive.	
There are only Time-Triggered behaviors in statecharts (every action performed every time, no actions on arcs, no entry actions)	
Every output listed in the output interface is set in every state.	
Every state variable mentioned in the statechart is defined in the requirements document.	
If AND substates are present, every output or state variable is only set in one ANDed (concurrent) set of substates.	
If OR substates are present, no transition crosses the superstate boundary	
The top level state machine and every ANDed or Ored set of substates contains exactly one intialization arc	
Every arc (except intialization arcs) is labeled with a unique number	

- **Guards need to mutually exclusive, do not need to be exhaustive**

- **Super/substates**
  - I hope you didn't do these but if you did…
  - Don't assign values to the same output from states that happen at the same time
  - Don't jump weirdly in and out of nested states.
  - Initialize nested state machines too.

Project 13 Grading

This value is computed from the average in the "End-to-End" sheet.	90	B	
Improvements Log			
Is there an entry for project 13 in the improvements log and minimum requirements sheet?	5	B	
is there an "Overall Project Comments" entry in the improvemenst log?	5	B	
Deductions	**Points lost**	**Point Breakdown**	**Dedu Awa**
Check the previous project grade sheet. Were the issues noted in that project addressed?	-19	B	
	Possible	**Points**	

- **Usual stuff**
- **Go through all your previous project and make sure you've addressed stuff.**
  - Most of it is stuff that's on this grade sheet anyway, so if you don't fix it, you'll be losing points twice.

Questions?