Recitation #2

18-649 Embedded System Engineering

Friday 5 Sept 2014

Electrical & Computer ENGINEERING

Note: Course slides shamelessly stolen from lecture

Carnegie Mellon

Announcements and Administrative Stuff

- **Project 2 posted**

- **TA office hours**
  - http://www.ece.cmu.edu/~ece649/admin.html#info
    - Monday: 5-6 PM (Sajjan Gundapuneedi)
    - Tuesday: 6-7 PM (Felix Hutchison)
    - Wednesday: 6-7 PM (Patrick Jang)
    - Thursday: 5-6 PM (Jeff Lau)
    - Friday: 6-7 PM (Felix Hutchison)
  - This week, undergrad lounge

- **Course announcements will be made mostly via blackboard**
  - Check blackboard prior to emailing staff
  - You are responsible for reading these (check it once a day or so)
  - Only critical notices sent by e-mail

Weekly Progress & Individual Contribution

- **Fill these in status reports every week and submit with your project**
- **Weekly progress updates due every week with your project submission**
- **First one due this project:**
  - Report hours
    - From the beginning of the semester till Project 1 submission
    - From handing in Project 1 to handing in Project 2
  - Report Contributions for each team member
- **Each project handin should have an Progress/Contribution report**
  - Projects without a report are not considered turned in.
  - You **WILL** receive the late penalty every day until it is resubmitted
  - We will try to remind you, but ultimately it is your responsibility to have it in your portfolio.

*

Weekly Progress & Individual Contribution

- **Lying on the Individual Contribution is a violation of Academic Integrity**
  - In industry this is called ***FRAUD***
- **Both the person who claims to have done the work, and the person whose work it is are in violation of Academic Integrity policy.**
  - Both students will ***FAIL*** the course.
- **The academic integrity policy can be found at http://www.ece.cmu.edu/~ece649/admin.html#cheating**

A Note on Late Submissions

- **If you submit late, e-mail staff**

- **If there is an on-time submission, the TA will grade that**
  - Unless you tell us otherwise (e.g., "please grade my late submission instead")
  - Either place a readme in the on-time submission folder or e-mail staff

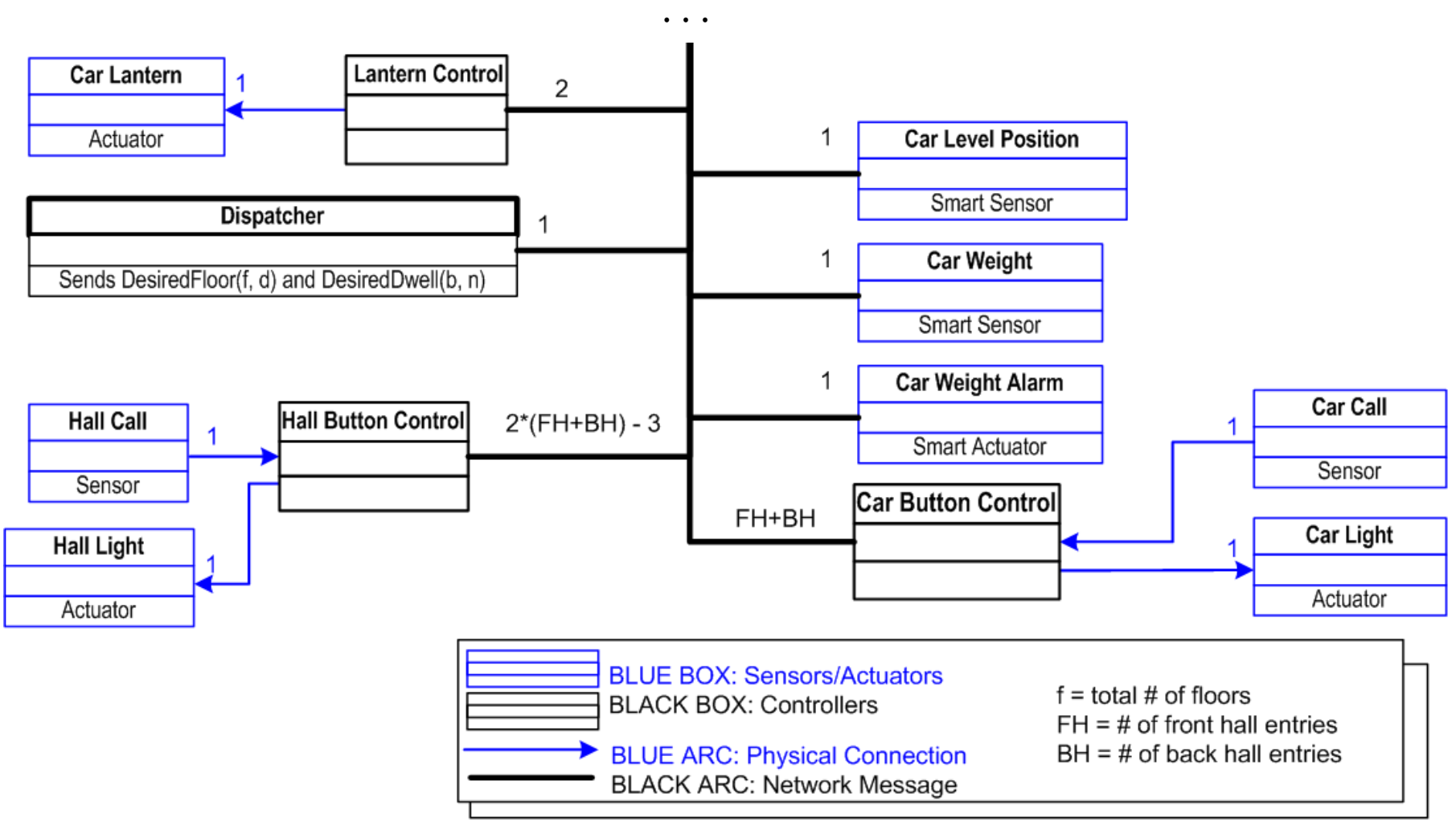
- **TAs are obligated to grade only one copy of your project**
  - Specifically, we aren't going to grade two versions and give you the higher of the two grades

Project 2 - Overview

- **Lecture Mon & Wed will go over all the material for Project 2**
  - We suggest you look through project 2 *now* so you can tie lecture to the project
- **Download the portfolio**
  - You will be responsible for submitting an updated portfolio every week
  - Refer to the portfolio matrix to see which parts you're updating each week
- **READ the architecture diagram**
  - Understand how modules are connected and communicate
  - Understand network vs physical messages
  - Architecture is fixed
- **READ the elevator behavioral requirements**
  - Detailed description of each part of the system
  - Defines inputs and outputs
  - Communication requirements are fixed
- **We give you a set of use cases**
  - Create scenarios for each use case
  - Create sequence diagrams based on each scenario

Architecture Diagram

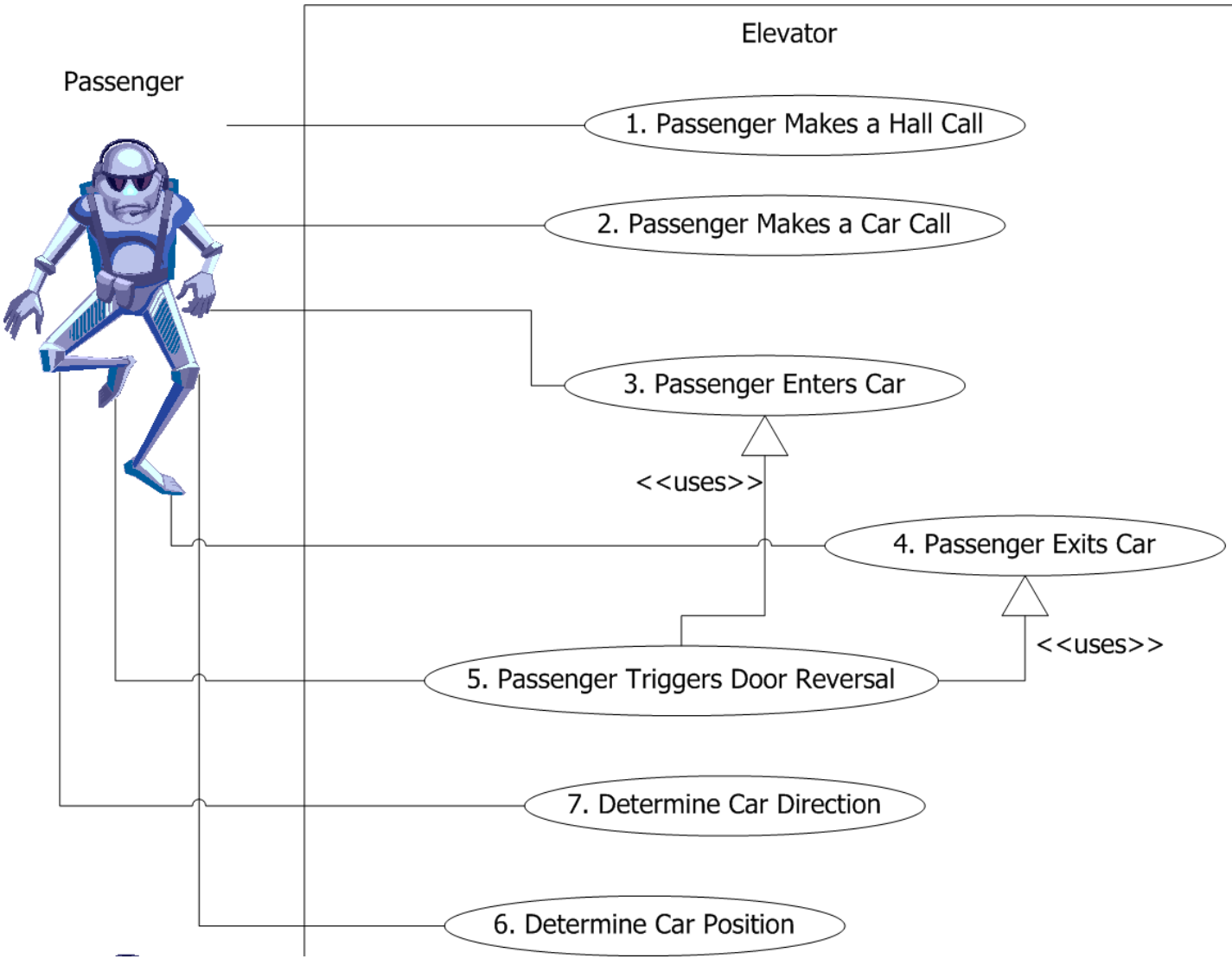
- **We provide a fixed architecture**
  - Your sequence diagrams will show communication between these components
  - Blue arc = physical commands, black arc = network message



*

Use Cases

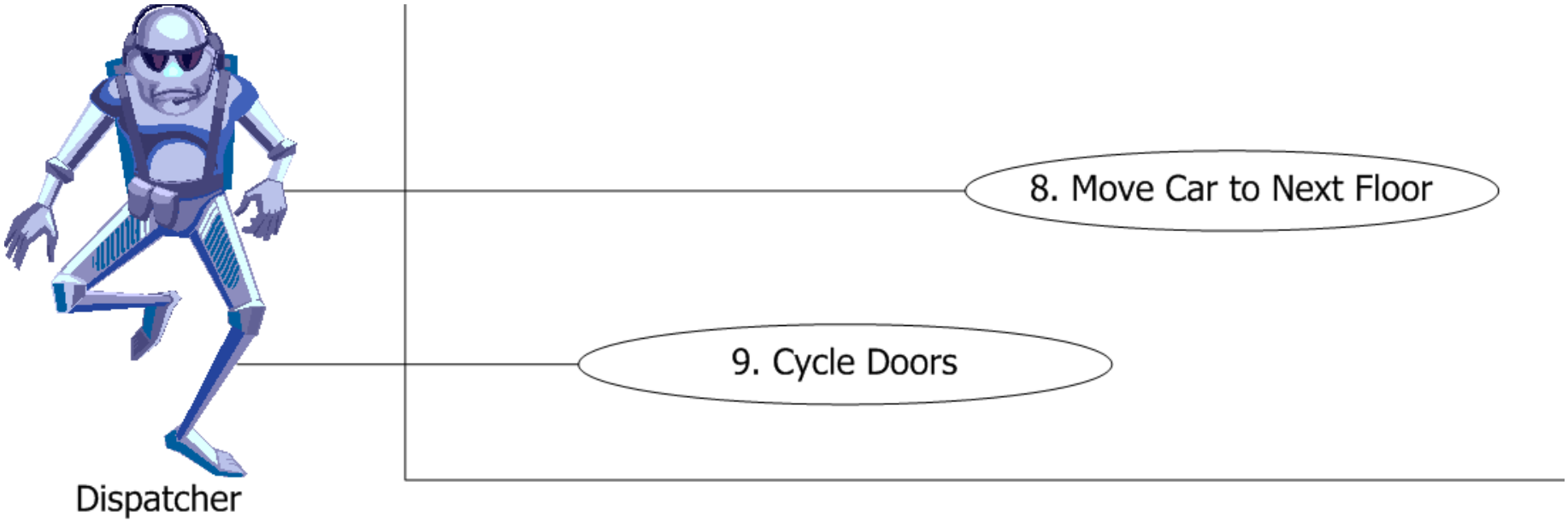
- **How can an outside entity interact with our elevator?**
  - Use cases define actions a passenger can perform with respect to the system



*

Use Cases (cont.)

- **The passenger is not the only entity that interacts with the elevator**
  - Dispatcher - Determines which floor to service and when to cycle the doors
  - In our system this is another controller (not a person)
    - But there are still elevators that use a person in this role



Behavioral Requirements

- **Read these**
  - Plan to spend some time getting familiar with this
- **Detailed description of what each part does**
  - Replication, instantiation, and assumptions
  - Behavioral requirements and constraints
  - Inputs and outputs
- **You'll notice some look very "empty"**
  - The baseline elevator is very simple (and inefficient)
  - Over the semester you'll be improving most of these components
- **Message library**
  - The communications requirements are fixed
  - You can ONLY use messages that we have defined for you
  - Use the defined inputs and outputs for each component

Scenarios

- **There are one or more scenarios for each use case**
  - 12 total, each team member does *at least* 3
  - Over the semester, you'll likely have to define more for a complete elevator
  - You're encouraged to think of more scenarios, but 12 is the minimum

- **We give you a set of preconditions**
  - You specify a scenario (similar to what you did in project 1)
    - Scenario describes what happens from preconditions to postconditions
  - Remember you're describing all system components, not just one
  - You specify the post conditions for the scenario
    - The state of the system after the scenario completes
  - Create a sequence diagram showing component interaction for the scenario

- **How long are scenarios supposed to be?**
  - Rule of thumb: Stop when you begin to describe another use case

Example Scenario 1B

- **Passenger arrives at a hallway when elevator is already there and the car is traveling in the same direction as desired by passenger.**
- **Preconditions:**
  - Car is at same floor as passenger.
  - Car is traveling in same direction d as desired by passenger.
  - At least one door[b,r] is open.
  - Hall button light [f,b,d] for passenger's desired direction is off.

Example Scenario 1B

- **Passenger arrives at a hallway when elevator is already there and the car is traveling in the same direction as desired by passenger.**
- **Preconditions:**
  - Car is at same floor as passenger.
  - Car is traveling in same direction d as desired by passenger.
  - At least one door[b,r] is open.
  - Hall button light [f,b,d] for passenger's desired direction is off.
- **Scenario:**
  - Door[b,r] starts closing.
  - Passenger arrives at a hallway[f, b] to use the elevator, with intent to travel in direction d.
  - Passenger presses hall call button before doors are fully closed, but after doors are too fully closed for passenger to enter.
  - Hall button lights up.
  - Doors complete closing.
- **Postconditions:**
  - Elevator is at the passenger's floor.
  - Door is closed.
  - Hall button light for passenger's desired direction is on.

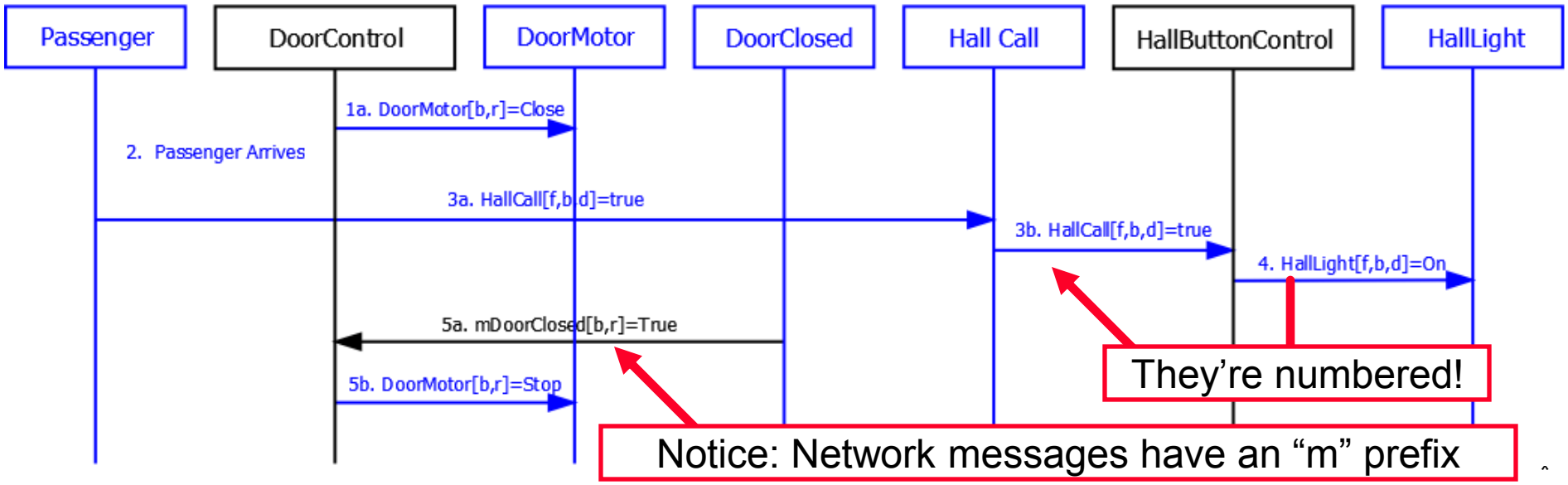
Sequence Diagrams

- **Sequence diagrams illustrate the flow of messages between objects**
  - Each component is shown as a box
  - Time progresses downward from the components
  - Arcs go between objects to show messages and physical commands
    - You must number all arcs!
  - Use the following color coding:

    Blue box = Sensor/actuator Blue arc = Physical command

    Black box = Controller Black arc = Network message

Sequence Diagram 1B:



Audit

- **Once you finish your (at least) three sequence diagrams**

- **Have one of your partners review them according to audit procedure**
  - Name of the person conducting the audit?
  - Are the events in the sequence diagram consistent with the steps in the scenario?
  - Are all arcs labeled with a valid command in the sequence diagram?
  - Are all arcs correctly colored (blue for physical message, black for network messages)?
    Are all boxed items correct parts of the elevator architecture?
  - Does each message arc in the sequence diagram originate from the correct object (according to the interfaces defined in the Requirements I and II documents)?

- **Include a copy of every audit as directed on project web page**

*

Individual Contribution

- **Track your own contributions to each project**
  - Each task for that week belongs to at least one team member
  - Comprehensive weekly list of who was in charge of completing what
  - We want to know how you tend to divvy the work

- **This will add detail to your progress report**
  - Minimum list of project requirements given, add any additional work you did as well

- **For this week**
  - Who worked on a Sequence Diagram? the Team Portfolio? a peer review?
  - This is all divided in a straightforward way this week, in future weeks the division of work will not be so neatly split

*

Portfolio

- **Update the appropriate parts of the portfolio template**
  - Portfolio Table of Contents
  - Scenarios and Sequence Diagrams
  - Improvements Log

- **Double-check**
  - The required parts of the portfolio are up to date (check the portfolio matrix)
  - All documents have group number and member names (including code)
  - All documents have uniform appearance
    - They don’t look like four people slapped them together at the last minute
    - Hint: Agree ahead of time what tools to use

What Do We Expect?

- **Grading criteria is focused on process**
  - The project is designed to teach you the importance of process and how to do it!
  - Don’t just go through the motions
  - Traceability, audits, logs, -- These things matter in industry!

- **We expect a design that passes acceptance test criteria**
  - You must have a working elevator to complete the course

- **Much of the project is open-ended**
  - Requires you to think about what is reasonable
  - We have fixed the architecture and message requirements to guide you

- **What do we NOT expect?**
  - A perfect elevator (how would you know?)

Project Tips

- **Leave time for traceability, audits and other process steps**
  - These can take longer than you expect
- **Put REAL effort into the early design phases**
  - Each project builds on the previous one
  - **Every** single group in years past recommended this by the end of the semester
  - If your design barely scrapes by, you will eventually have to fix it
  - Traceability requirements mean that your fixes will have to be propagated all the way through the design
- **Don't blow off bug tracking**
  - When you fix a bug, log it, and propagate the fix through the whole design
  - We'll do diffs on your submissions to check bug tracking / propagating fixes
- **Look at tools for automating repetitive tasks (e.g. makefiles)**
  - The simulator might have the feature you're about to implement
- **Resist the urge to put lots of time into automating one-off tasks**

Teamwork and Administration Tips

- **Deal with these issues early (first or second week)**

- **Establish meeting times and team roles**
  - Plan to meet with your group the same time every week
  - Exchange contact info too!

- **Decide what tools to use**
  - Pick tools everyone can use (especially for diagrams)
  - Look at tools for collaboration (e.g. version control)
    - Or at least set up a location to share the current design files
  - Use a version control tool (e.g., SVN, GIT)
    - Note: GitHub can be tricky for spreadsheet merges

- **Read over the project before Recitation**
  - Come ready with questions, it's the best time to ask

Team Meeting Tips

- **Meet early for planning**
  - Review the assignment together
    - Make sure everyone in your group understands the assignment
  - Get started on the assignment together
  - Record things you don't understand or need clarifications on
    - Assign a member to visit office hours with your questions
  - Distribute the remaining work before you end the meeting

- **Establish a checkpoint mid-week (even if just by e-mail)**
  - Make sure everyone's on track prior to the deadline

- **Keep meeting minutes**
  - Decisions made, tasks assigned, etc.
  - Make sure everyone gets a copy so they know what they're responsible for!

Suggestions and Reminders

- **For project 2, spend some time reviewing the documentation**
  - There's a lot of documentation, it takes time to review
  - Do it prior to your team meeting

- **Start early with assignments**

- **Reminder: Assignments are due Thursday evening by 10:00 PM**

*

Questions?