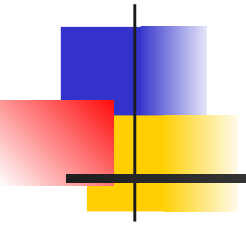


# Team Software Project

Getting started

*Let's Not Do Waterfall Projects*

# How to Get Started?

# You Already Have a [Partial] Plan

Your project proposal is a partial project plan:

- development approach
- initial time line with goals & deliverables
- technical approach
- some tools you will use

# Each Iteration

Must do this each iteration:

1. short iteration plan
    - goal
    - major tasks to perform
    - work product to produce
2. create a task board
    - tasks consistent with your plan
    - may have more tasks or breakdown big tasks
3. Retrospective & short summary
    - Retrospective is about process not the product
4. TA Meeting & Demo

# How to Find Things?

1. Project Hub (Information Hub)

- one place where someone can learn everything about your project.
- typically a wiki or project management site (like Assana or Trello)

2. Github Repository - for code, issues, build status

# Milestone

Milestone:

an indicator that shows tangible progress toward completing a project, along with objectively verifiable criteria that show the milestone has been achieved (or not).

A milestone usually shows you achieved some goal or major work of the project. The milestone criteria are how to verify that the work has been satisfactorily done.

# Define a Milestone

Each iteration, include at least one milestone in your Iteration Plan.

Milestone is usually related to the goal for the iteration.

The milestone is something that objectively shows you achieved a goal or made significant progress.

# Not Objectively Verifiable Criteria

These are bad milestones:

[ ] Study Django [*when is it "done"? how does it show progress toward finishing the project?*]

[ ] Write User Stories [*when are they done?*]

[ ] Team meeting [*so what? how did it help project?*]

[ ] Task Board [*this isn't a work product or activity*]

# Binary Milestones

A milestone is either done or not done.

There are no "90% done" milestones.

-- *Steve McConnell, Software Project Survival Guide*

# Milestones According to Agile

*"Working software is the primary measure of progress."*

They are not fans of documentation as milestones.

# Create Your Own Milestones

*Create milestones that are specific to your project plan & work.*
*Not a copy of examples.*

*Each milestone should have objectively verifiable ("done" or "not done") criteria.*

# Project Initiation Milestone

[ ] Vision statement reviewed, approved, and published

[ ] Initial timeline with iteration goals & features reviewed and published

[ ] Technical approach agreed on and published

[ ] Initial set of mock-ups of UI and screen-flow created, reviewed, and published

[ ] Project home page on Github and wiki created.

... *what is significant work <u>your project</u> needs to do?*

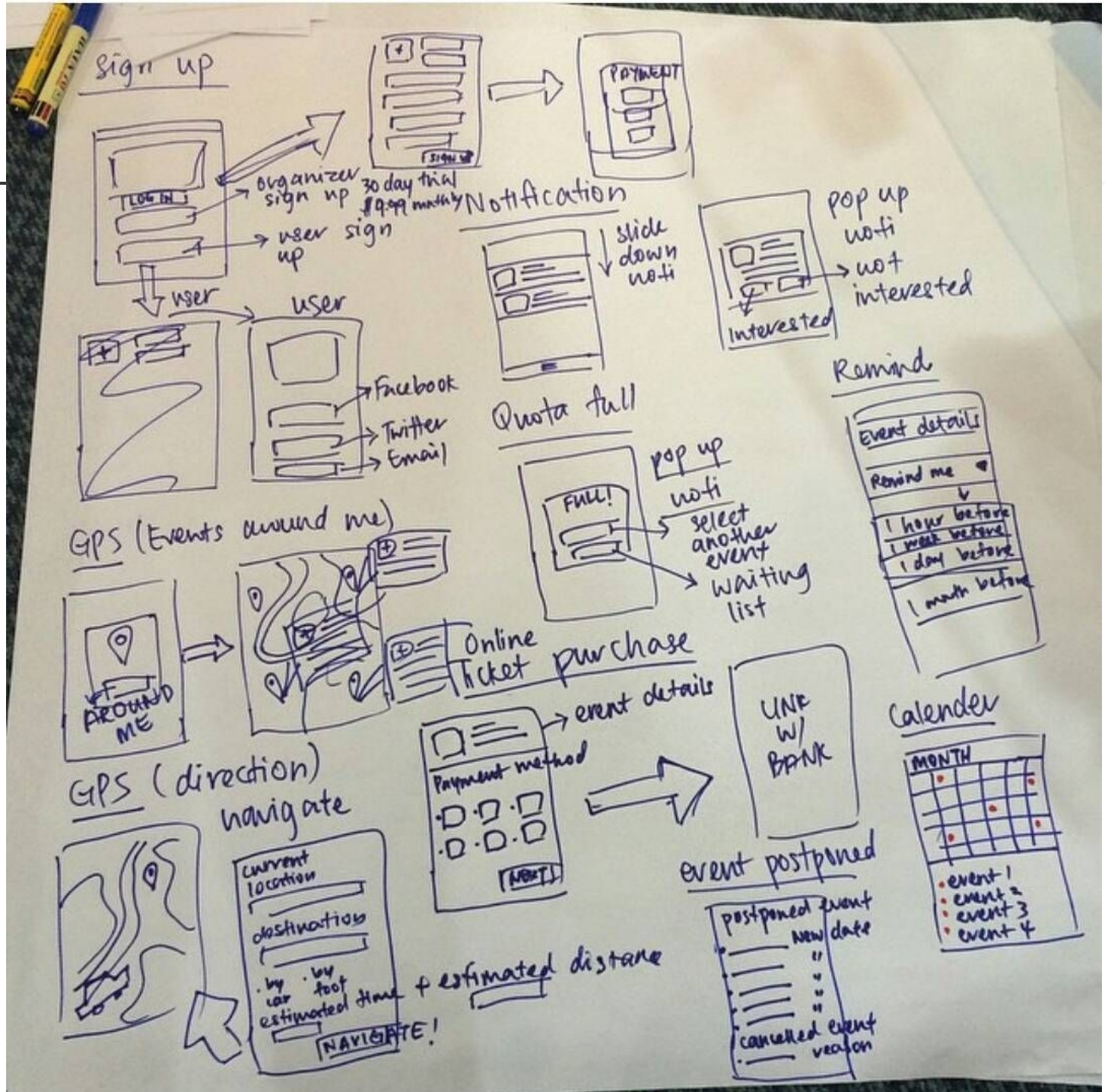
*published = published on wiki or Google Drive, with link on project home. approved = team and TA/instructor all agree on it.*

# Page Flow or Screen Flow

A mock-up of each page & how the application flows from one page to the next.

Can be:

1. one screen per piece of paper
2. a large diagram (next slide)


Sign up
LOG IN
organizer sign up
30 day trial
$9.99 monthly
user sign up
PAYMENT
user
user
Facebook
Twitter
Email
Notification
slide down noti
pop up noti
not interested
interested
Remind
Event details
Remind me
1 hour before
1 week before
1 day before
1 month before
Quota full
FULL!
pop up noti
select another event
waiting list
GPS (Events around me)
AROUND ME
Online Ticket purchase
event details
Payment method
NEXT
UNI W/ BANK
Calender
MONTH
event 1
event 2
event 3
event 4
GPS (direction) navigate
current location
destination
by car
by foot
estimated time + estimated distance
NAVIGATE!
event postponed
postponed event
New date
cancelled event
vocation

# Ideas for Iteration 1

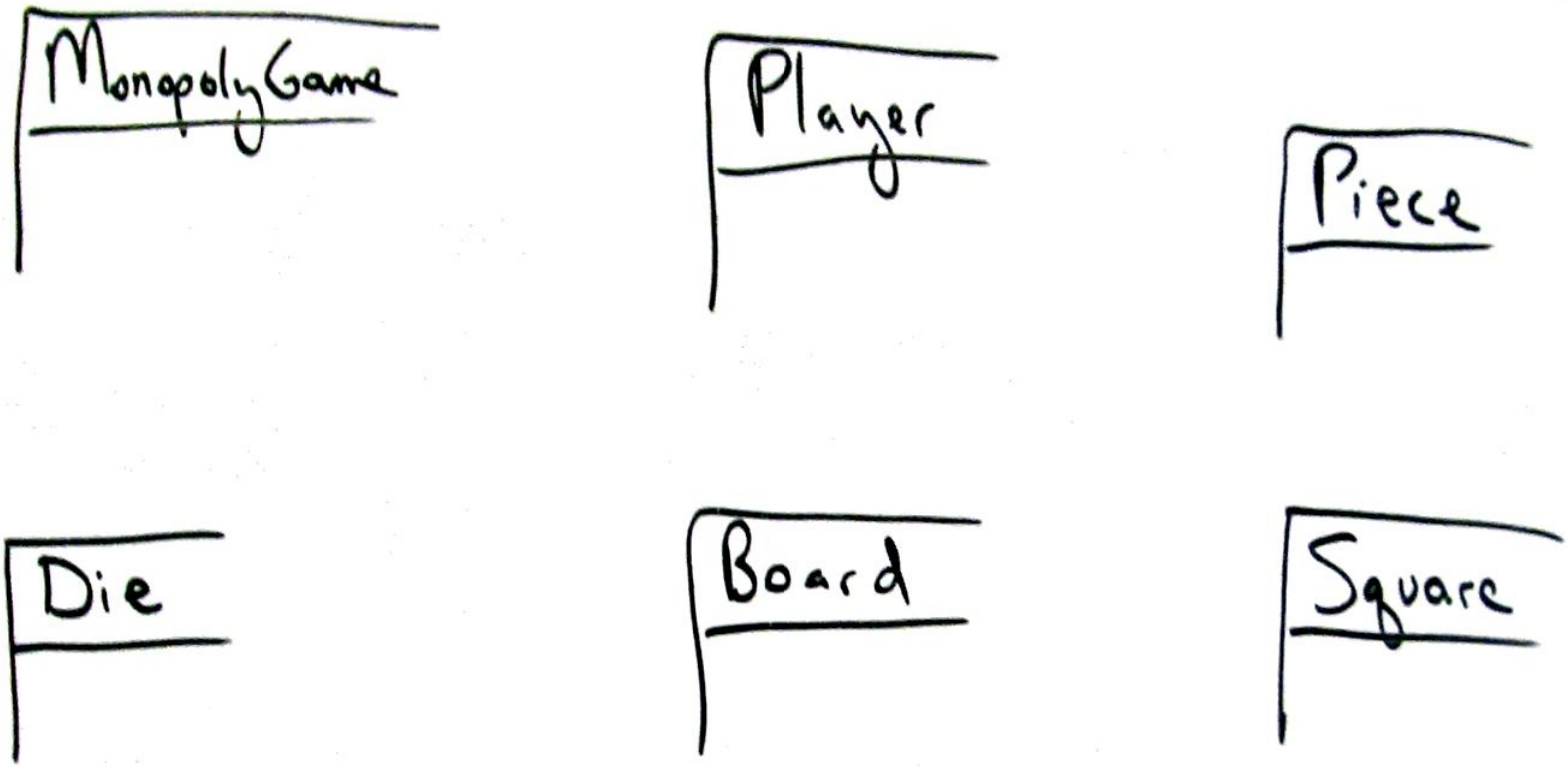
*Create a Domain Model*

Create a "Domain Class Diagram" showing...

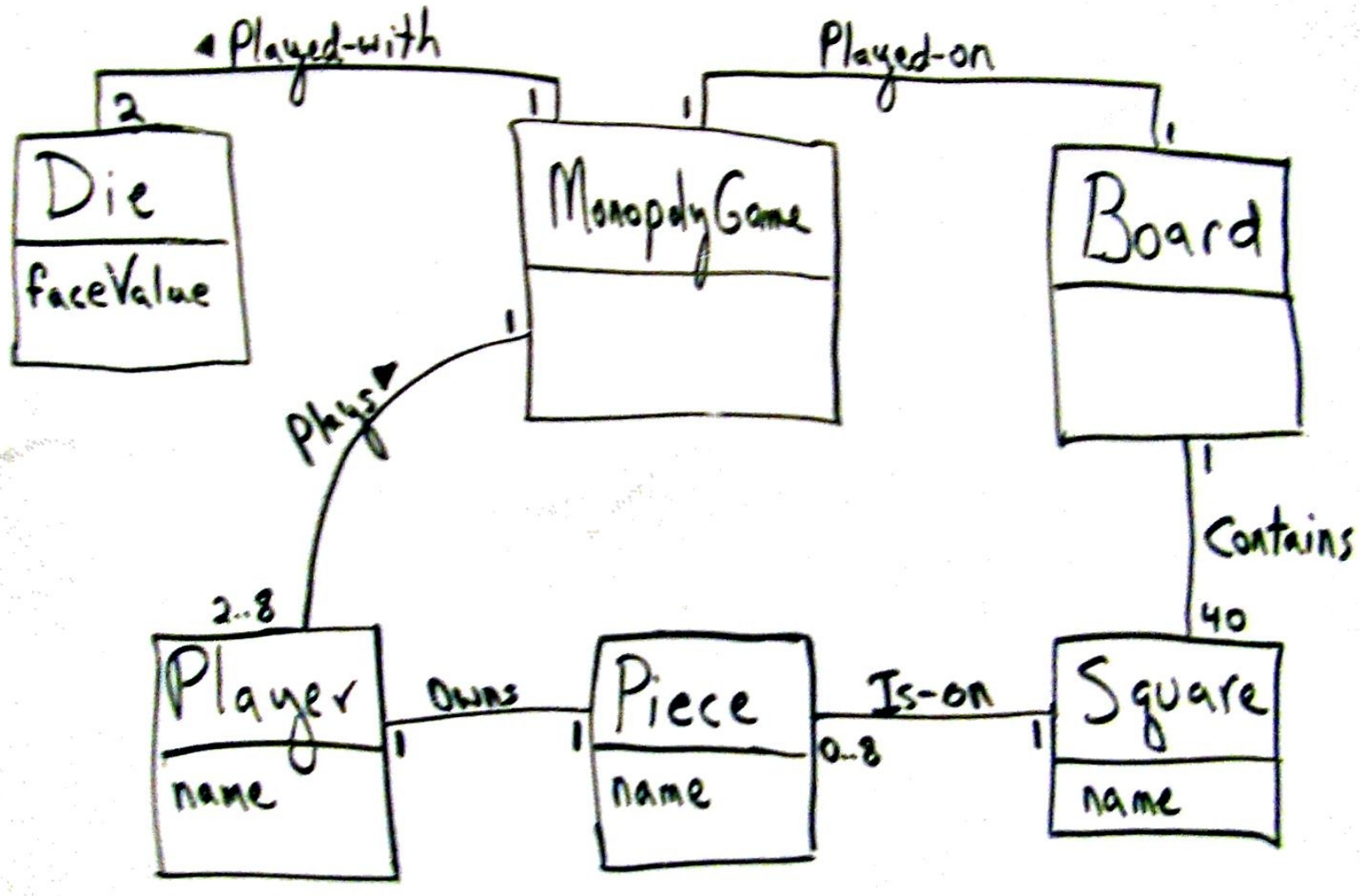
- important classes in the domain
- what they know (attributes)
- major responsibilities

Another tool for this is CRC cards.

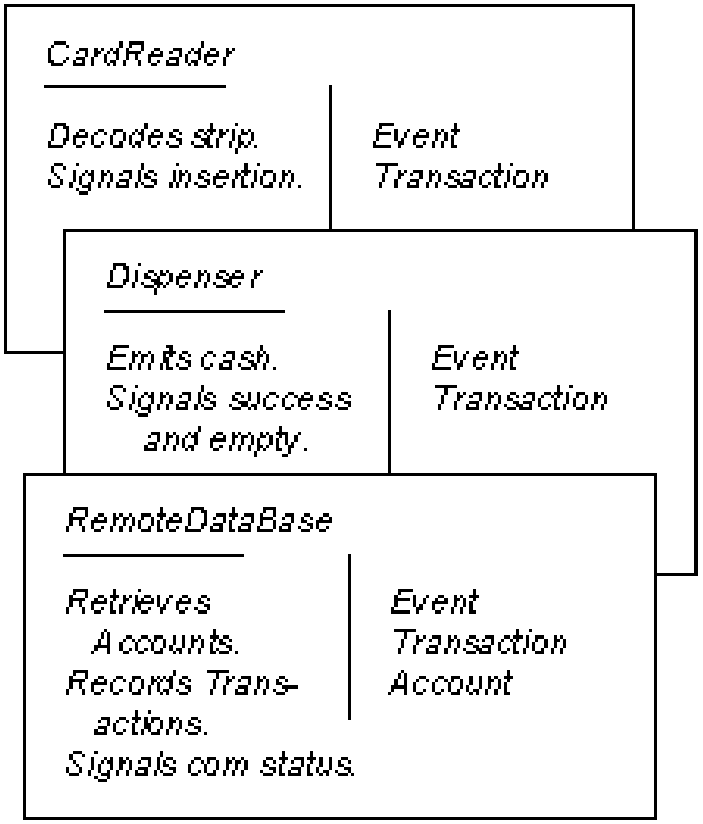
# Model: Identify classes



# Identify relations and key attributes



# Class-Responsibility-Collaborator Cards




Back Side: notes, important things to validate

# Concentrate on dynamic aspects

*Beginners spend too much time on static structure,*
*not enough time on dynamic (behavior)*

*-- Craig Larman*

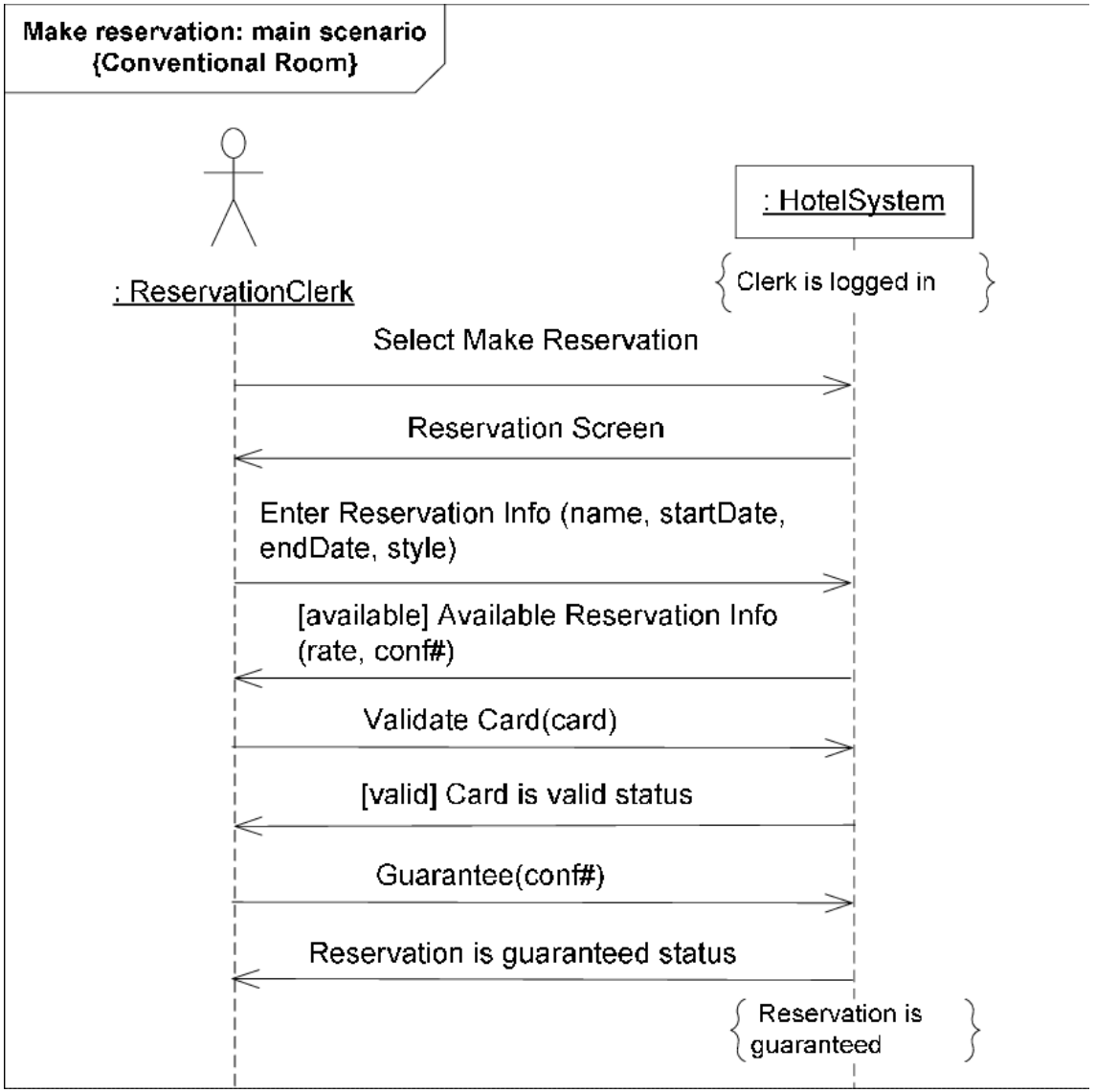
You should draw sequence diagrams, too.

# System Sequence Diagram

*Shows interaction between "user" & system for one usage scenario.*

*Use SSD to discover what system must do.*

*May show external services.*



# *Identify what you do **Not** know*

Things you don't know could be **risks**.

Identify important things you don't know.

Domain or business related "don't knows":

*what are the important terms and conditions of an apartment rental?*
*(so we can include them in the rental app UI)*

- learn more about app domain, what users want
- ask people who do know

# *Technical **Not** Knowns*

Examples:

- *How to send and receive JSON requests in Django?*
- *How to integrate Google Calendar or existing Calendar framework into our app?* (It's not useful to write your own calendar -- no one wants another calendar!)

Tasks:

- Study possible solutions
- Work through a tutorial or sample app
- Create your own proof-of-concept app

# Work should be VISIBLE

*For any group project it is important that progress is visible.*

Lack of visibility => miscommunication, "surprises", defective or unusable work products.

- Your work should produce work products.
- Work products should be shared with team.
- Make it visible and online.

Agile Practice: "*Develop in plain sight*".

# Let's Not Do Waterfall Projects

Typical course project:

- Project presentation during last week.

- Project code & docs submitted during final.

- Instructor finds problems after semester ends.

*Missed opportunity to learn.*

*Let's not do this.*

# Iteration Review & Demo

After each iteration:

- review progress & plan with TA.
- what did you do?
- any changes to the product or project plan?

Every iteration after the 1st one:

- Demo running software
- It should do more than the previous demo

# Iteration Review & Demo

## Who and where to demo?

- Each iteration a different team member must present
- Can be in-person or online