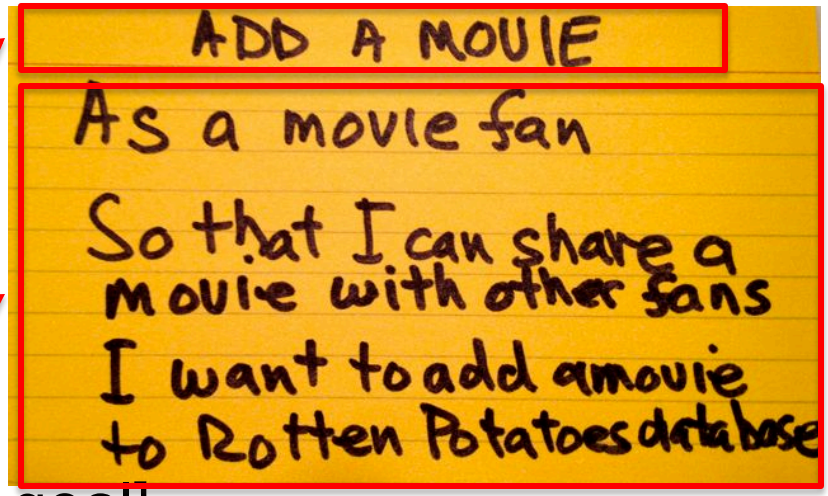
Behavior-Driven Design (BDD)

- BDD asks questions about behavior of app *before and during development* to reduce miscommunication
  - Validation vs. Verification
- Requirements written down as *user stories*
  - Lightweight descriptions of how app used
- BDD concentrates on *behavior* of app vs. *implementation* of app
  - Test Driven Development or TDD (future segments) tests implementation

User Stories

- 1-3 sentences in everyday language
  - Fits on 3" x 5" index card
  - Written by/with customer
- "Connextra" format:
  - Feature name
  - As a [kind of stakeholder],
    So that [I can achieve some goal],
    I want to [do some task]
  - 3 phrases must be there, can be in any order
- Idea: user story can be formulated as *acceptance test before* code is written



Why 3x5 Cards?

- (from User Interface community)
- Nonthreatening => all stakeholders participate in brainstorming
- Easy to rearrange => all stakeholders participate in prioritization
- Since stories must be short, easy to change during development
  - As often get new insights during development

Different stakeholders may describe behavior differently

- *See which of my friends are going to a show*
  - As a theatergoer
  - So that I can enjoy the show with my friends
  - I want to see which of my Facebook friends are attending a given show
- *Show patron's Facebook friends*
  - As a box office manager
  - So that I can induce a patron to buy a ticket
  - I want to show her which of her Facebook friends are going to a given show

Product Backlog

- Real systems have 100s of user stories
- *Backlog*: User Stories not yet completed
- Prioritize so most valuable items highest
- Organize so they match SW releases over time

Related: Spike

- Short investigation into technique or problem
  - E.g. spike on recommendation algorithms
  - Experiment, hack, do whatever works
- Bound the time allotted
- When done, *throw code away*
  - Now that know approach you want, write it right!

Measuring Productivity

- A measure of team productivity: calculate avg no. stories / week?

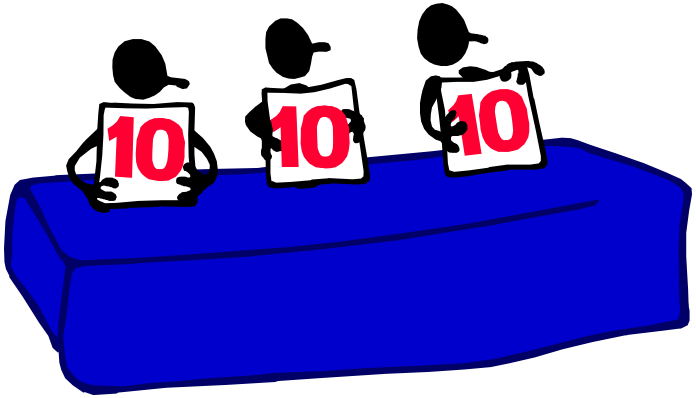
  - But some stories much harder than others
- Rate each user story in advance on a simple integer scale
  - 1 for straightforward stories, 2 for medium stories, 3 for very complex stories
- Velocity: avg number of *points* / week

More on Points

- Once get experience, Fibonnaci scale is commonly used: 1, 2, 3, 5, 8
  - (Each new number is sum of previous 2)
  - At Pivotal Labs, 8 is extremely rare
  - Advice: when starting, if ≥5 then **split up story!**
- Teams vote: hold up fingers, take average
  - If a big disagreement (2 and 5), discuss more

More on Points

- ≥5 => divide user story into simpler stories, group into *epics*
  - backlog not too demanding
- Doesn't matter if velocity is 5 or 10 points per iteration
  - As long as team consistent
- Idea is to improve self-evaluation and suggest number of iterations for feature set



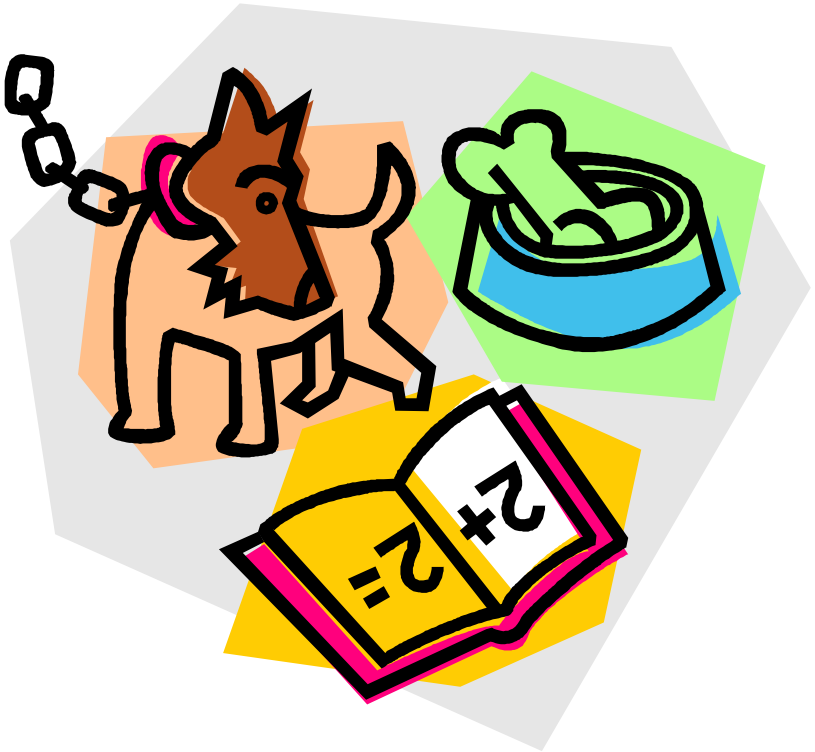
Team Cyberspace Whiteboard

- Tracker allows attaching documents to User stories (e.g., LoFi UI)
- Wiki with Github repository
- Google Documents: joint creation and viewing of drawings, presentations, spreadsheets, and text documents
- Campfire: web-based service for password-protected online chat rooms

Stories vs. Layers

- "Dividing work by stories helps all team members understand app & be more confident when changing it"
- "Tracker helped us prioritize features and estimate difficulty"
- "We divided by layers [front-end vs. back-end vs. JavaScript, etc.] and it was hard to coordinate getting features to work"
- "It was hard to estimate if work was divided fairly…not sure if our ability to estimate difficulty improved over time or not"

SMART User Stories



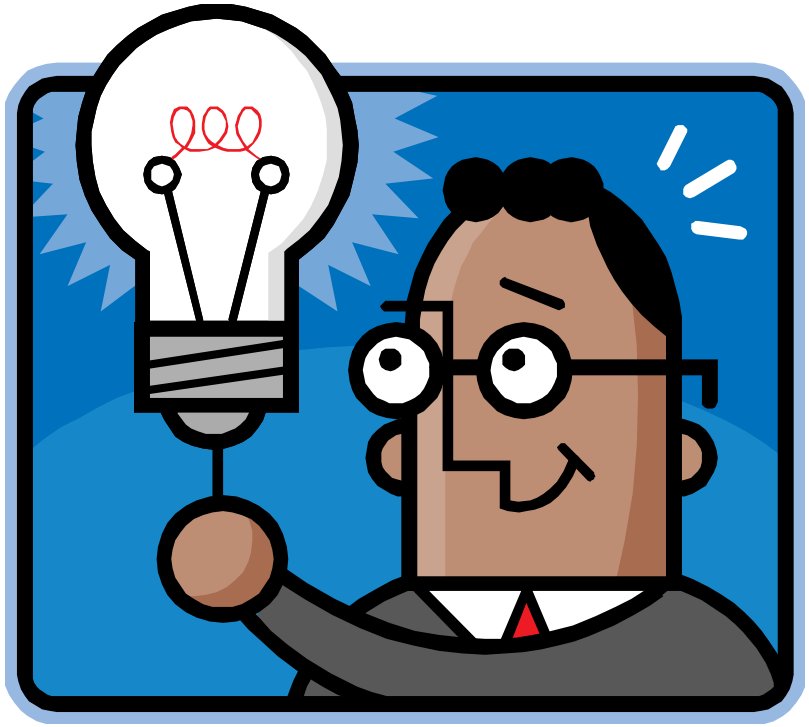
David Patterson

Creating User Stories

- How do you know if you have a good user story vs. bad user story?
  - Right size?
  - Not too hard?
  - Is worthwhile?

SMART stories

- **S**pecific
- **M**easurable
- **A**chievable
  (ideally, implement in
  1 iteration)
- **R**elevant
  ("the 5 why's")
- **T**imeboxed
  (know when to give up)



Specific & Measurable

- Each scenario testable
  - Implies known good input and expected results exist
- Anti-example: "UI should be user-friendly"
- Example: Given/When/Then.

1. *Given* some specific starting condition(s),
2. *When* I take specific action X,
3. *Then* one or more specific thing(s) should happen

Achievable

- Complete in 1 iteration
- If can't deliver feature in 1 iteration, deliver subset of stories
  - Always aim for working code @ end of iteration
- If <1 story per iteration, need to improve point estimation per story

Relevant: “business value”

- Discover business value, or kill the story:
  - Protect revenue
  - Increase revenue
  - Manage cost
  - Increase brand value
  - Making the product remarkable
- Can you include stories that don’t have obvious business value?

5 Whys to Find Relevance

- *Show patron's Facebook friends*

  As a box office manager

  So that I can induce a patron to buy a ticket

  I want to show her which Facebook friends are going to a given show

  1. Why?

  2. Why?

  3. Why?

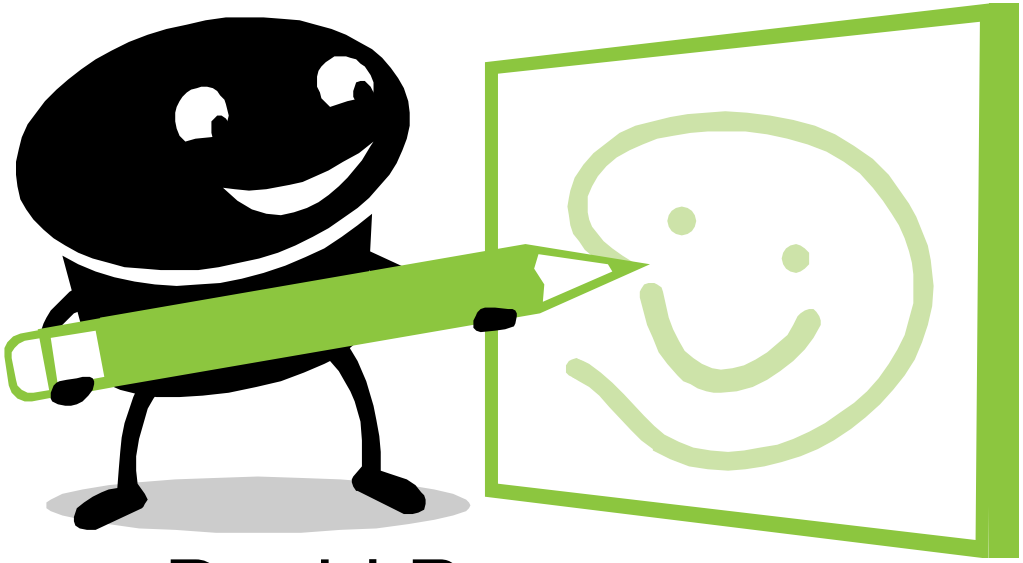
  4. Why?

  5. Why?

Timeboxed

- Stop story when exceed time budget
  - Give up or divide into smaller stories or reschedule what is left undone
- To avoid underestimating length of project
- Pivotal Tracker tracks velocity, which helps avoid underestimate

Lo-Fi UI Sketches and Storyboards



David Patterson

Building Successful UI

- SaaS apps often faces users

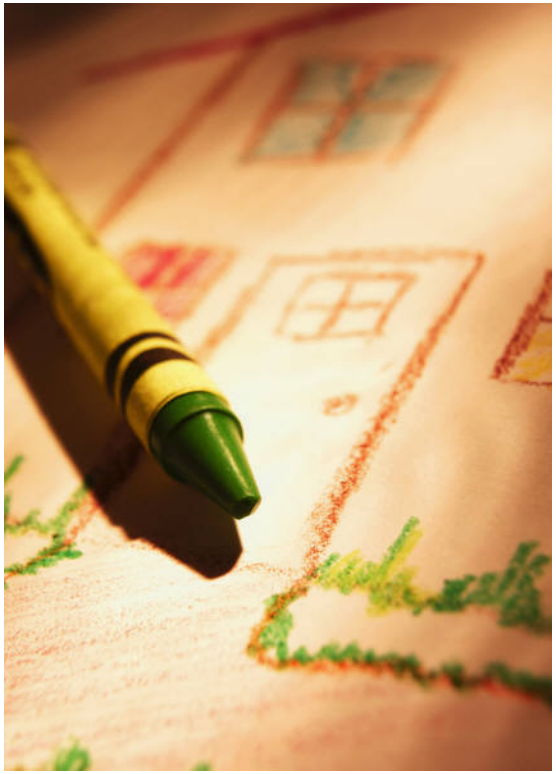
⇒User stories need User Interface (UI)

- How get customer to participate in UI design so is happy when complete?
  - Avoid WISBNWIW* UI?
  - UI version of 3x5 cards?
- How show interactivity without building prototype?

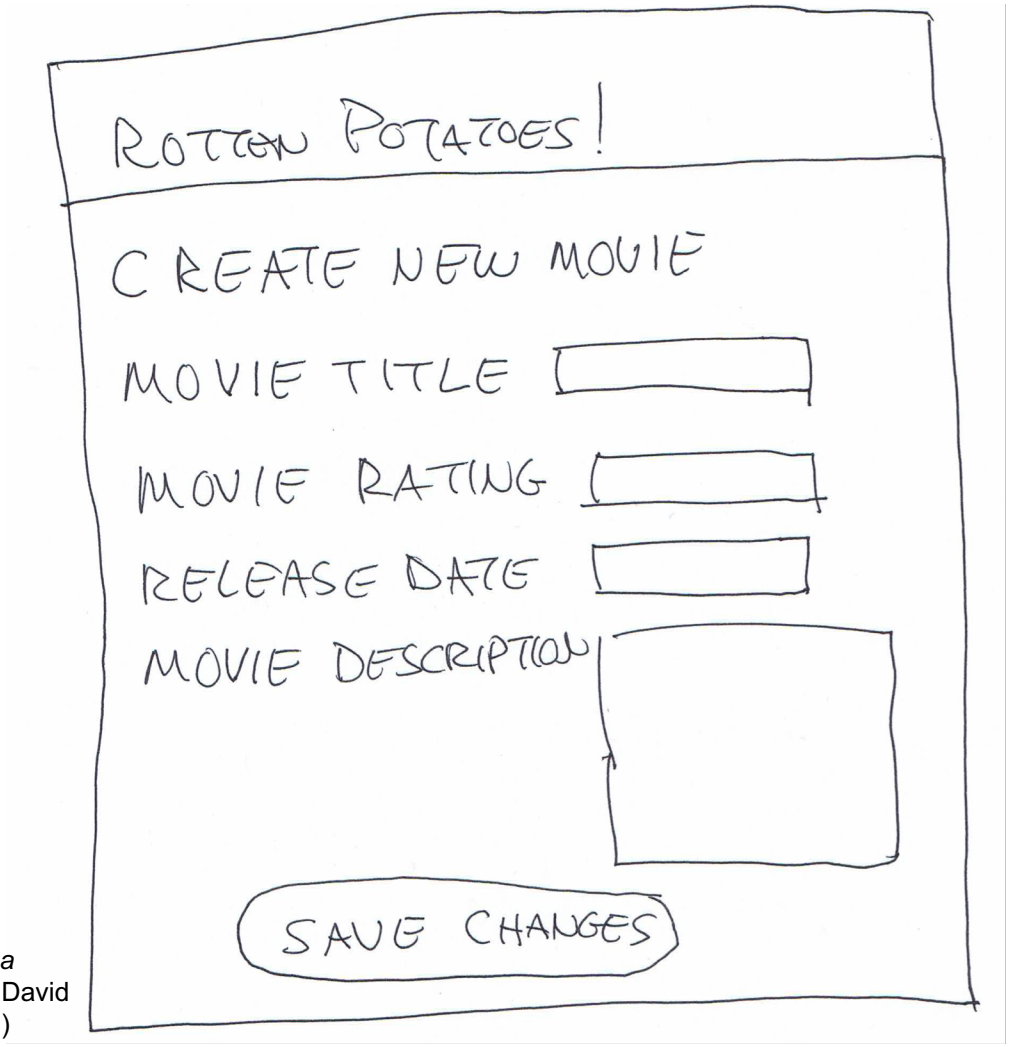
* What-I-Said-But-Not-What-I-Want

SaaS User Interface Design

- UI Sketches: pen and paper drawings or "Lo-Fi UI"



Lo-Fi UI Example

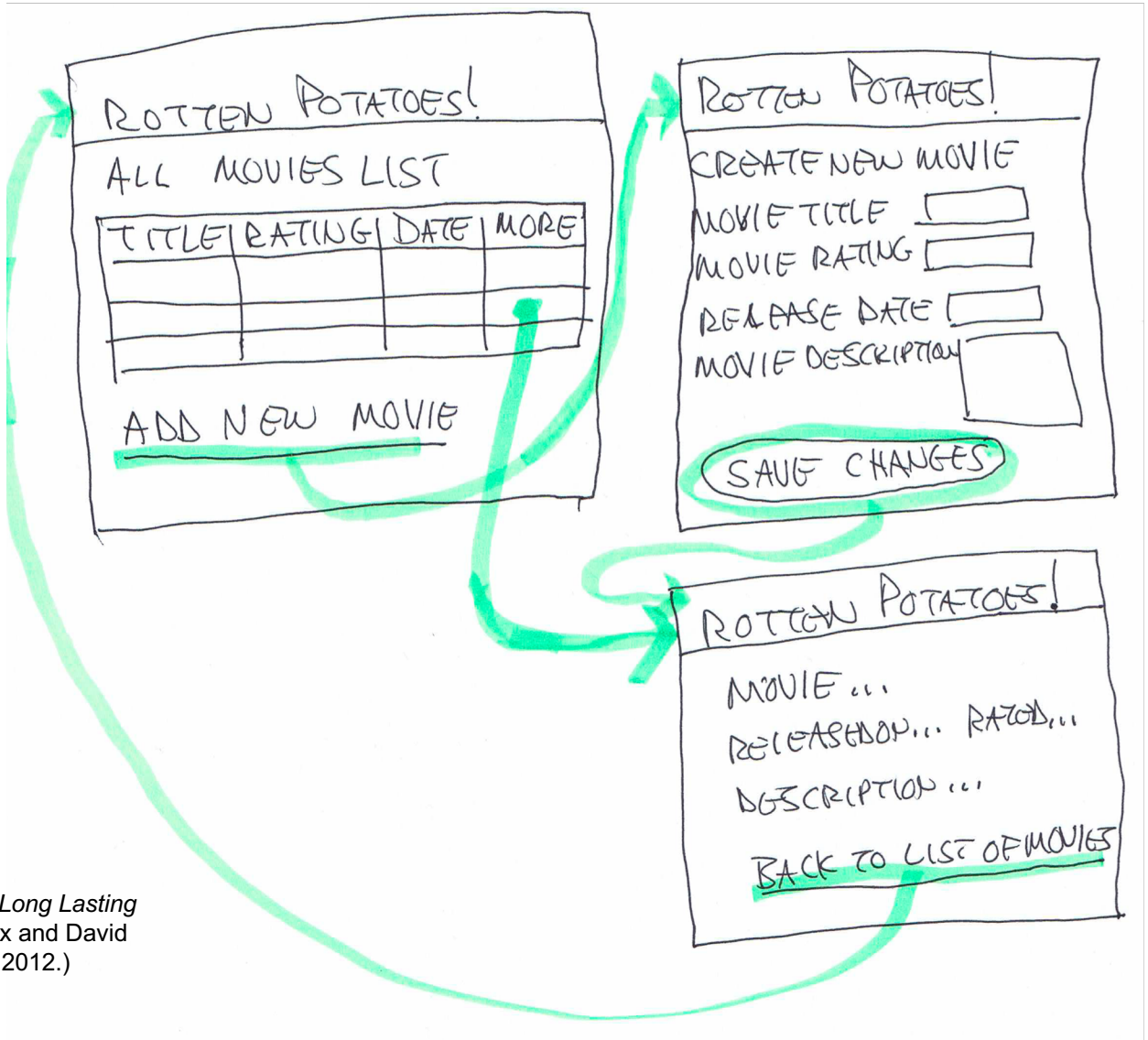


(From *Engineering Software as a Service* © by Armando Fox and David Patterson, used with permission)

Storyboards

- Need to show how UI changes based on user actions
- HCI => “storyboards”
- Like scenes in a movie, but not linear

Example Storyboard



(Figure 4.4, *Engineering Long Lasting Software* by Armando Fox and David Patterson, Alpha edition, 2012.)

Lo-Fi to HTML

- Tedious to do sketches and storyboards, but easier than producing HTML! And…
  - Less intimidating to nontechnical stakeholders
  - More likely to suggest changes to UI if not code behind it
  - More likely to focus on *interaction* rather than colors, fonts, …
- CSS (Cascading Style Sheets) can make it look nice later

Working with the customer: BDD & Lo-Fi Prototyping

- "Lo-fi and storyboards really helpful in working with customer"
- "Frequent customer feedback is essential"
- "What we thought would be cool is not what customer cared about"
- "We did hi-fi protoypes, and invested a lot of time only to realize customer didn't like it"
- "Never realized how challenging to get from customer description to technical plan"