PROJECT MANAGEMENT

CSC404 Tutorial Slides

Context for Game Design

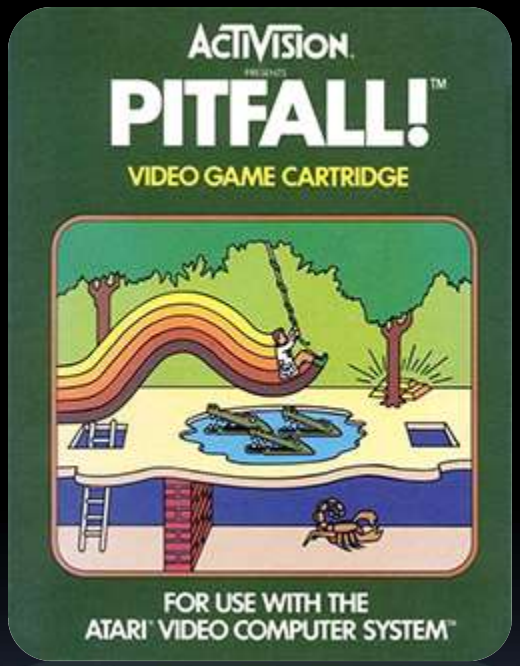
- Game development is an agile development process.
  - Incremental development
    - Demonstrable product
    - Product milestones
  - Small groups
  - Changing requirements
  - Unexpected developments

Recommended approach

- Constant communication
- Working build
- Shared workspace
- Team development
- Prioritize features
- Playtesting Playtesting Playtesting!

Biggest pitfalls

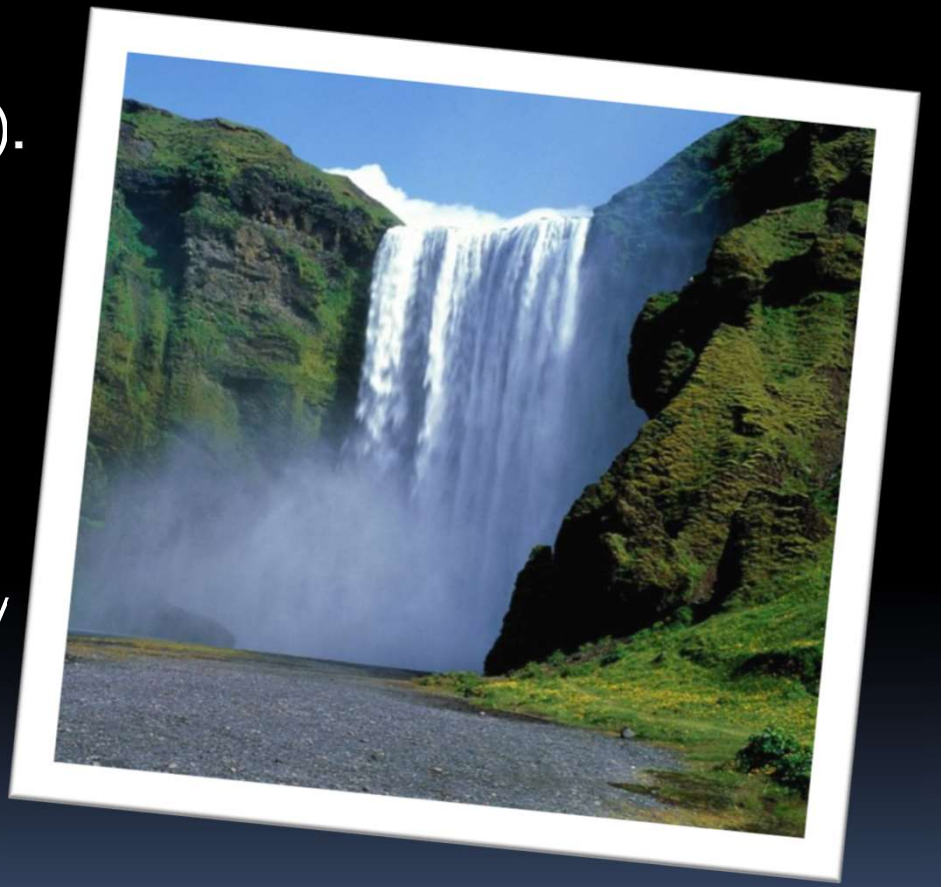
- Lack of communication
  - Between group members
  - With presentation audience
  - With player
- Last-minute assembly
- Navel gazing
- Time management
- Feature management



Key notes on scrum development

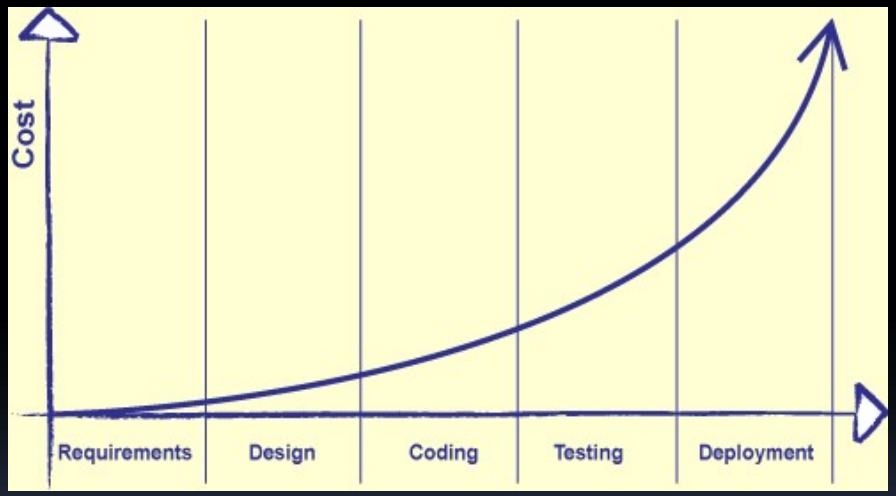
Sturdy Methodologies

- "Sturdy" (also known as "traditional" or "non-agile").
  - Underlying philosophy: Measure twice, cut once.
    - Or: The cheapest bug to fix is one that doesn't exist.
  - Inspired by traditional engineering methods, where planning and budgeting is key to winning contracts, and where late changes are expensive and unpopular.
- **Not for game design.**



Sturdy vs Agile

- Differences in practice are much less than differences in rhetoric.
- Example: Boehm's Curve.



- Need a methodology that responds to this issue.

Agile Methodologies

- "Agile" is a relative term.
  - Lots of small steps, with continuous testing and refactoring.
  - Underlying philosophy: Change is inevitable, so plan for it.
    - Or: "No battle plan ever survives contact with the enemy"
      - Helmuth von Moltke
  - Refers to the sturdy camp (unflatteringly) as BDUF ("Big Design Up Front")
  - Inspired by open source and 1990s web development.

Agile Motivation

- Agile home ground:
  - Low criticality
  - Senior developers
  - Requirements change often
  - Small number of developers
  - Culture that thrives on chaos

Sturdy vs Agile

- *"Reality is that which, when you stop believing in it, doesn't go away."* -- Philip K. Dick
- It's hard to get customers to sign off on "we'll make it up as we go along" and "trust me".
  - The waterfall method feels familiar and credible.
  - On the other hand, most scientific research is agile.
  - The second agile project is easier to sign off on than the first.
- "Adapting to change" is good, but "constant change" seems sloppy.
  - Is "continuous refactoring" the same thing as "better late than never", or a sign of "code first, ask questions later"?

Implications for you

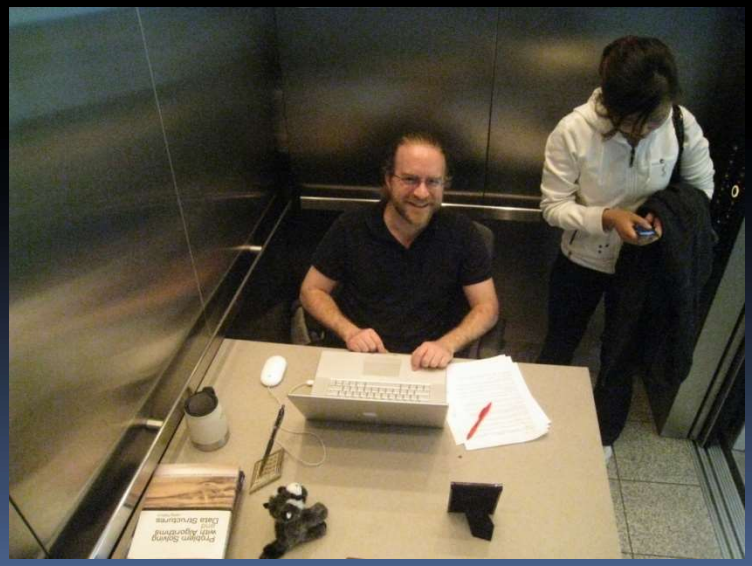
- So what does this mean for you (individual), you (group) and your project?
  - Your system is *always* buildable and releasable.
    - Easy for us to loop over your commits and check this ☺
  - Regular meetings *with minutes*.
    - Both with client and within group.
  - Issue tracking and prioritization.
  - Comprehensive testing.
  - Refactoring.
  - Collective code ownership.
    - We reserve the right to quiz you on *your team's code* as a part of any exercise.

Scrum Methodology

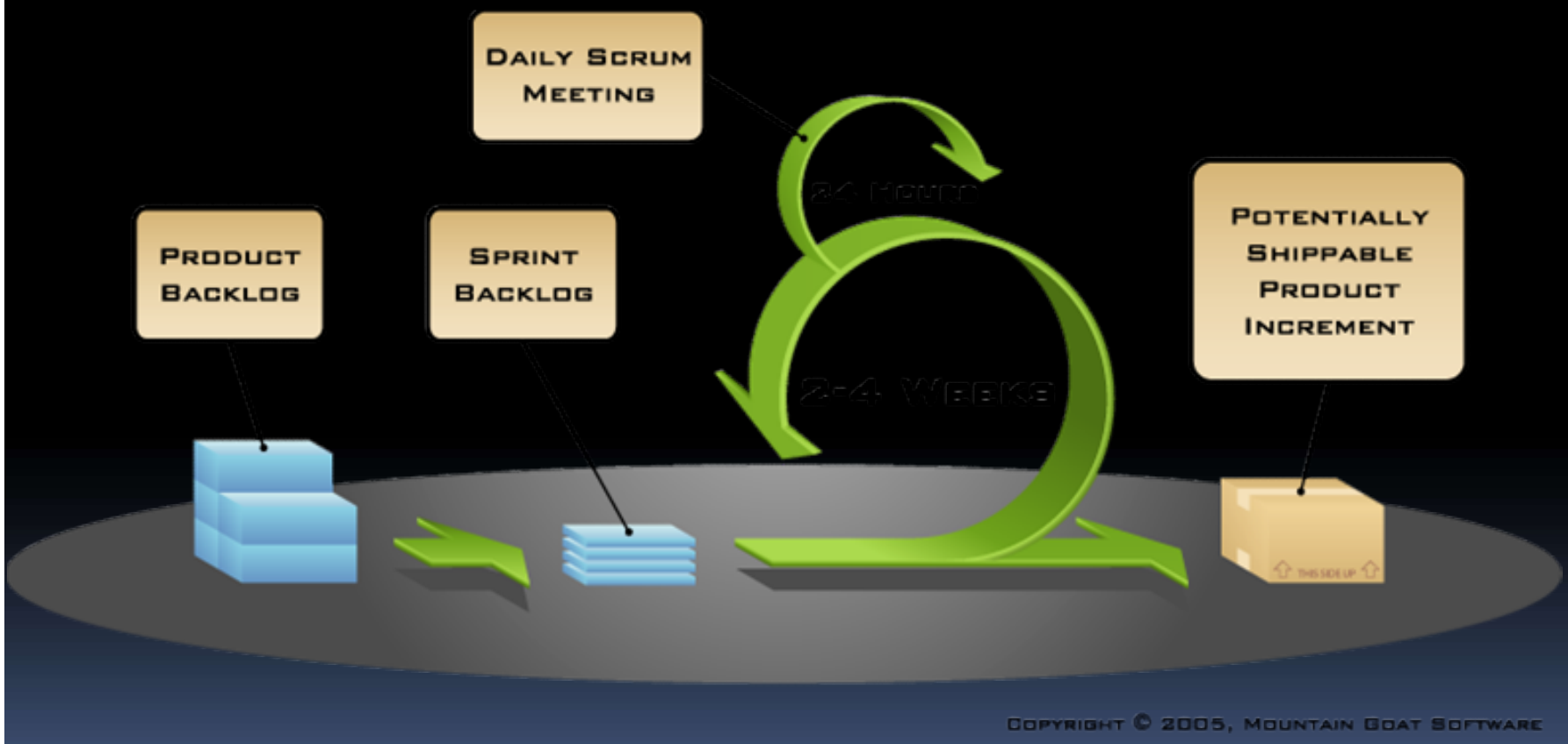
- All about *iterative*, *incremental* development.
- Roles in scrum:
  - Scrum Master
    - Maintains the scrum process
  - Product Owner
    - Who this program is for, or who's paying for it.
  - Team
    - A group of about 7 people who do the actual work.

Before You Can Scrum

- Selling your product (sometimes before it's been made).
  - The vision statement, aka the "elevator pitch".
  - Helps to keep everybody headed in the same direction.
  - A good way for project members to introduce themselves, or to define their (self-)importance at conferences.



Scrum at a Glance



Elements of Scrum

- Product Backlog:
  - A list of things the team hasn't yet implemented
  - Stays forever there until project is done
  - Each item has a priority
  - Visible by client
- Sprint Backlog:
  - Basically, a "to do" list
  - Smaller list of things needed to be done
  - Property of the team
  - Due by next sprint

More Elements of Scrum

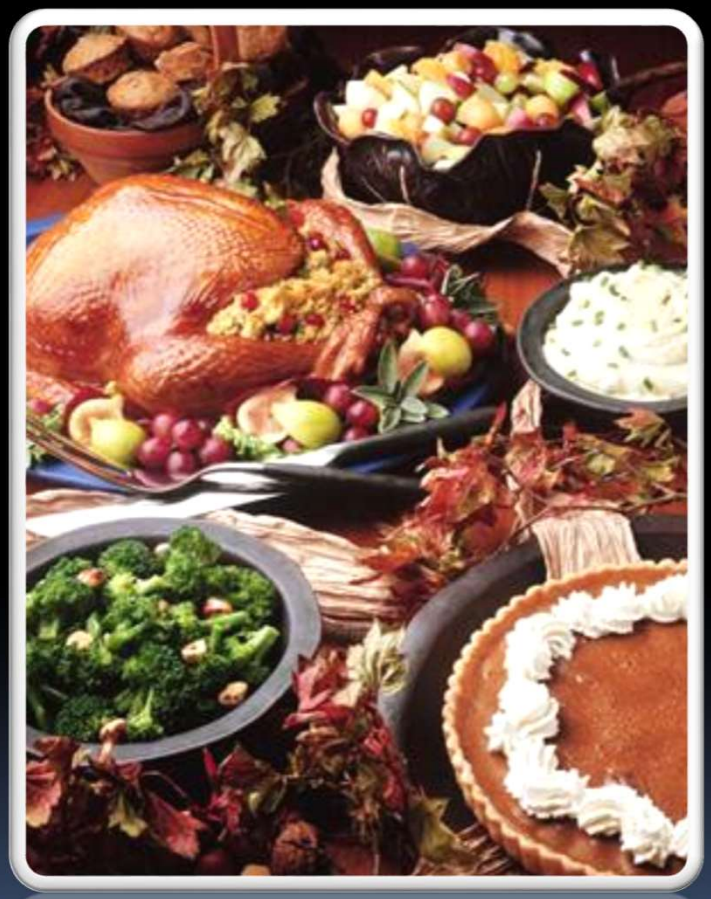
- Sprint:
  - Typically 2-4 weeks long (hence the 30 days)
  - During this time the team gets to finish the sprint backlog items
  - No one is allowed to change the sprint backlog during a sprint
  - Ideally, a team should every 24 hours discuss where they are at in the development

Justification for Scrum

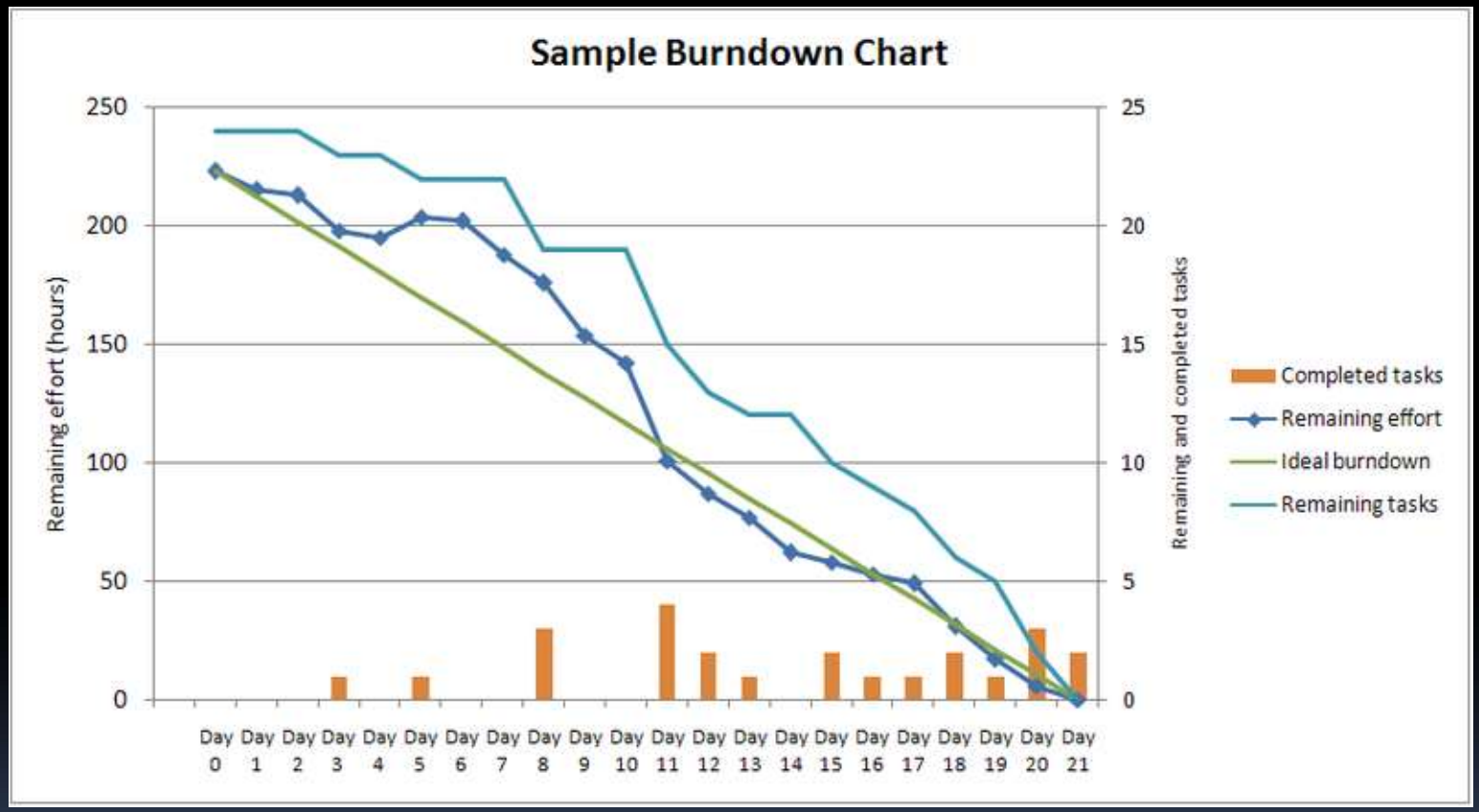
- Scrum takes into account:
  - Things will fall apart because humans are involved.
    - Accept that this problem can't be solved.
    - Work to instead get as much done as possible.
  - Tools
    - Does not discriminate (backlog can be in Excel)
  - Organization
    - Lots of communication is required (lots of meetings/updates)
    - Everybody needs to know how the project is doing.

Scrum Example: Thanksgiving Dinner

- Roles
- User stories
  - No technical details.
  - When complete, no further consultation with user necessary.
- Product backlog
  - User-side requirements.
  - User sets priority of each item.
- Sprint goals, backlog
  - Developer-side items.
  - Team sets priority of items (prioritization chart).
- Burn down chart
  - Chart of work remaining in sprint backlog.



Burn Down Chart



- If Steve was more diligent, this would be filled with dinner items. Oh well.