Deliverables: Beta Release

- **Installation package**
- **Application sources and binaries**
  - One-step build for all sources
- **Latest specification & design documents**
  - Keep it short! Consider what is/isn't important for customers / devs.
- **Release notes**
  - Detailed instructions on how to run a (small) demo of your app
  - Known issues with prioritization, expressed in a bug tracking system
- **Up-to-date test plan**
- **Automated tests (unit, acceptance, etc.)**
- **Up-to-date schedule**
  - Including what has been done and what remains to be done

Questions to consider: Who is your audience – customers or developers? What do they expect from this release? What defines success for them?

Deliverables (tentative list): Final Release

- **Installation packages**
  - Including all of the items below
- **Application sources and binaries**
  - Separate distributions (installation packages) for customers and developers
  - One-step build – from compiling all sources to creating installation packages
- **User & technical documentation (separate)**
  - User doc: What does my mom need to know (and do) to run this product?
  - Technical doc: What does a support team need to know to work on version 2?
- **Release notes**
  - Known issues with associated severities & priorities
    - Include a link to your bug tracking system's tasks/tickets that reflect those issues
  - Specify where your current code repository is
  - Instructions on running the installer and your app are moved to the user doc.
- **Latest test plan**
- **Automated tests (unit and acceptance)**
  - Test coverage would be a very welcome addition.
- **Up-to-date schedule**
  - Things that have been accomplished (of those that were planned)
  - Features (of those initially planned) that are now pushed to version 2 or abandoned
    - How much would each such feature cost (in terms of dev effort)?

Questions to consider: Who is your audience – customers or developers? What do they expect from this release? What defines success for them?

Lecture 14: Risk Management

"If Las Vegas sounds too tame for you, software might be just the right gamble."
-- Steve McConnell

Outline

- The essence of risk and risk management
- Risk management themes: past and upcoming
- Risk exposure and prioritization
- Coping with risks
- Risk assessment in practice – exercises

Resources

- *Rapid Development*, by Steve McConnell
  - Ch. 5, 41;
  - Ch. 27 (optional)
- *Software Requirements*, by Karl Wiegers

Definitions of Risks

1. a condition that could cause loss or otherwise threaten the success of a project
2. a condition characterized by *lack of control*

Risk Management

- The goal
  - Successful project completion
- The job
  - Identify the risks
  - Address the risks with specific actions
  - Avoid or resolve the risks *before* they become real threats to the project
- Remember this:
  - Mistakes are made on *every* project.
    - "I feel so much better since I gave up hope."
  - The goal is to get to successful project completion *even though* mistakes were and will be made.

Levels of Risk Management

- **Crisis management**
  - Address risks only after they have become problems
- **Fix on failure**
  - Address risks only after they have manifested
- **Risk mitigation**
  - Plan for when risks will show, but no attempt to prevent
- **Prevention**
  - Identify and prevent risks from becoming problems
- **Elimination of root causes**
  - Identify and eliminate factors that make risks possible

It's **ALL** About Risk Management

- Themes we have discussed so far in the course:
  - Lifecycle models
  - Product proposal pitches
  - Requirements gathering techniques
  - Prototyping
  - Architectural design notations
  - Design principles
  - Usability design
  - Testing
  - Unit testing
  - Incremental releases
  - Project retrospectives
  - Team conversations
- What risk(s) does each of these practices help to manage / mitigate?

It's **ALL** About Risk Management: Still to Come

- What are some important risk areas that we have not yet covered in the course?

It's **ALL** About Risk Management: Still to Come

- Some important risk areas that we have not yet covered in the course… but will try:
  - Scheduling and estimation
  - Feature "creep"
  - "Code rot"
  - Version configuration chaos
  - Uncalibrated code and product quality
  - Inexperienced personnel
  - Interpersonal conflict
  - Miscommunication
  - Legal hurdles
  - Misalignment of incentives
  - Politics among stakeholders

The Multitude of Risks

- McConnell gives a list of 111 (!) schedule risks.
  - This does *not* even include risks beyond scheduling.
- How can one pay attention to all possible risks at once and proactively address them?
  - It's a full-time job
    - Managers who are good at it are sought after and get paid very well.
  - Not all potential risks apply to all situations.
    - There are patterns; past experience or data on similar projects/teams can show what to pay extra attention to.
  - Not all risks that apply are equally important or likely.
    - Calls for risk prioritization

Risk Exposure

- **Exposure = P(Loss) * |Loss|**
  - E.g.: a 15% chance of slipping a project schedule by 10 weeks => a slippage time of 1.5 weeks is to be expected.
- Allows a more intelligent estimate of the size of the "cushion" period you need for the project
- Don't take the estimation too far!
  - It's not precise, after all.

Student Submission

Risk Prioritization

- Compute the risk exposure for each risk.
- Sort all risks by their exposure: from high to low.
- Move large-loss risks up on the list.
  - To avoid unlikely but potentially catastrophic events
- Address the risks from top to bottom on the list.

Risk	P(Loss)	|Loss|	Pri
A	10%	10	
B	20%	5	
C	5%	25	
D	90%	1	
E	40%	2	
F	99%	1	

Approaches to Coping with Risks

- Avoid the risk
- Transfer risk off the critical path
- Buy information
  - Bring in outside help
  - Prototype
- Publicize risk
- Schedule to accommodate some risk
- Monitor risks as project progresses

Student Submission

Risk Management in Practice: Likelihood of Risks in Your Project

Choose (circle) the likelihood for each risk category:

Risk category	Risk likelihood
Changing requirements	High / Med / Low
Personnel issues (conflict, inexperience)	High / Med / Low
"Feature creep"	High / Med / Low
Is what you're building technically feasible?	High / Med / Low
Is what you're building compelling to customers?	High / Med / Low

Student Submission

Risk Management in Practice: In a Different Domain

- Risk is sometimes modeled as a random variable.
- E.g.: Professors A, B, C, and D assign grades at random according to known distributions. Give your preference for the profs (1 highest, 4 lowest):
  - Prof A: P(4.0) = ¾, P(0.0) = ¼
  - Prof B: P(3.0) = ½, P(2.0) = ½
  - Prof C: P(4.0) = ¼, P(3.0) = ¼, P(1.7) = ½
  - Prof D: P(2.4) = 1