CITS5501 Software Testing and Quality Assurance
Risk Management

Unit coordinator: Arran Stewart

April 24, 2018

Source

- Pressman, R. *Software Engineering: A Practitioner's Approach*. McGraw-Hill, 2005.

Motivation

Why look at risk?

- Because every project, even the simplest, involves risks
- And because people are usually not good at allowing for them

Risk

What is risk?

- Something that *might* happen, which would cause loss
  - If it will *definitely* happen, that's not a risk – it's usually called a "constraint"
  - If it doesn't result in some kind of unwanted consequence (i.e., a loss), it's also not a risk

What are some sorts of risk?

One categorization (Pressman):

- Project risks
    - Things that could affect the project plan or schedule
- Technical risks
    - Risks resulting from the problem being *harder to solve* than we thought
- Business risks
    - Things that threaten the viability, as a product, of the software to be built
    - (Could be commercial viability, but also applies to in-house or even open source software)

Sorts of risk – project risks

Project risks:

- Things that could affect the *project plan and/or schedule*, causing it to slip and costs to increase
- Examples: personnel move or resign, resources turn out to be more expensive/take longer to acquire than estimated, exchange rates shift

Sorts of risk – technical risks

Technical risks:

- Risks resulting from the problem being *harder to solve* than we thought
- They threaten the quality and timeliness of the software to be produced
- If they eventuate, implementation may become difficult or impossible
- Examples: design turns out to be infeasible, dependencies have bugs, language/framework turns out to be difficult to maintain

Sorts of risk – business risks

Business risks

- Things that threaten the viability, as a product, of the software to be built (even if there are no technical or project barriers to it being implemented on time and within budget)
- Examples: over-estimating the market for a product; being beaten to release by a competitor

What are some sorts of risk? (2)

Another categorization (due to Robert N. Charette, 1989):

- Known risks
  - Those risks that can be uncovered after careful evaluation of the project plan, the business and technical environment in which the project is being developed, and other reliable information sources (e.g., unrealistic delivery date)
- Predictable risks
  - Those risks that are extrapolated from past project experience (e.g., past turnover)
- Unpredictable risks
  - Those risks that can and do occur, but are extremely difficult to identify in advance

Reactive vs. proactive risk strategies

- *Reactive* risk strategies
  - “Don’t worry, I’ll think of something”
  - The majority of software teams and managers rely on this approach
  - Nothing is done about risks until something goes wrong
    - The team then flies into action in an attempt to correct the problem rapidly (“fire fighting”)
- *Proactive* risk strategies
  - Steps for risk management are followed
  - Primary objective is to *reduce* risk and to have a contingency plan in place to handle unavoidable risks in a controlled and effective manner

Steps for risk management

1. Identify possible risks; recognize what can go wrong
2. Analyze each risk to estimate the probability that it will occur and the impact (i.e., damage) that it will do if it does occur
3. Rank the risks by probability and impact
    - Impact may be negligible, marginal, critical, and catastrophic
4. Develop a plan to manage those risks
    - If probability is low and impact is negligible, you may simply to elect to ignore some risks.

Risk identification

- A systematic attempt to identify risks to the project
- Because if they aren't identified, how can they be planned for?
- Two main sorts of risk:
  - Generic risks
    - Risks that are common to every software project
  - Product-specific risks
    - Risks that can be identified only by those a with a clear understanding of the technology, the people, and environment specific to the software that is to be built

Risk identification (2)

- One way of identifying risks – use a *risk checklist*
- Focuses on known and predictable risks in specific subcategories

Risk checklists

Some typical categories of items on risk checklists:

- Product size – large systems are simply more complex, and have more ways to go wrong
- Business impact – constraints may be imposed by management or the marketplace
  - e.g. medical products may require government approval
- Customer characteristics – risks associated with sophistication of the customer and the developer's ability to communicate with the customer in a timely manner
  - e.g. With an unsophisticated customer, there may be more chance that requirements are missed.

Risk checklists cont'd

- Process definition – risks associated with the degree to which the software process has been defined and is followed
- Development environment – risks associated with availability and quality of the tools to be used to build the project
- Technology to be built – risks associated with complexity of the system to be built and the "newness" of the technology in the system
- Staff size and experience – risks associated with overall technical and project experience of the software engineers who will do the work

A project risk questionnaire

1. Have top software and customer managers formally committed to support the project?
2. Are end-users enthusiastically committed to the project and the system/product to be built?
3. Are requirements fully understood by the software engineering team and its customers?
4. Have customers been involved fully in the definition of requirements?
5. Do end-users have realistic expectations?
6. Is the project scope stable?
7. Does the software engineering team have the right mix of skills?
8. Are project requirements stable?
9. Does the project team have experience with the technology to be implemented?
10. Is the number of people on the project team adequate to do the job?
11. Do all customer/user constituencies agree on the importance of the project and on the requirements for the system/product to be built?

Risk questionnaires (2)

- Other questionnaires useful for identifying risks can be found on the web and in the literature on software project management.
- Especially recommended:
    - McConnell, S. *Software Project Survival Guide*. Microsoft Press, 2014.
    - Applies to small-to-medium projects, but is useful reading even for the very large

Risk estimation

Risk estimation

- Risk estimation attempts to rate, for each risk:
  - The probability that it eventuates
  - The magnitude of loss associated with the risk, should it eventuate
- The project planner, managers, and technical staff perform risk estimation so that risks can be prioritized, and limited resources be allocated where they will have the most impact

Risk tables

- A *risk table* provides a project manager with a simple technique for risk projection
- It consists of five columns
  - Risk summary – short description of the risk
  - Risk category – (see the previous slides onf risk categorization)
  - Probability – estimation of risk occurrence based on group input
  - Categorization of impact – e.g. (1) catastrophic (2) critical (3) marginal (4) negligible
  - RMMM – pointer to a paragraph in the Risk Mitigation, Monitoring, and Management Plan

Developing a Risk Table

- List all risks in the first column (by way of the help of the risk item checklists)
- Mark the category of each risk
- Estimate the probability of each risk occurring
- Assess the impact of each risk based on an averaging of the four risk components to determine an overall impact value
- Sort the rows by probability and impact in descending order
- Draw a horizontal cutoff line in the table that indicates the risks that will be given further attention

Assessing risk impact

- Three factors affect the consequences that are likely if a risk does occur
  - Its nature – This indicates the problems that are likely if the risk occurs
  - Its scope – This combines the severity of the risk (how serious was it) with its overall distribution (how much was affected)
  - Its timing – This considers when and for how long the impact will be felt
- The overall risk exposure formula is $\text{RE} = \text{P} \times \text{C}$, where
  - $\text{P} =$ the probability of occurrence for a risk
  - $\text{C} =$ the cost to the project should the risk actually occur
- Example
  - $\text{P} = 80\%$ probability that 18 of 60 software components will have to be developed
  - $\text{C} =$ Total cost of developing 18 components is \$25,000
  - $\text{RE} = .80 \times \$25{,}000 = \$20{,}000$

Risk mitigation, monitoring, and management

Given a risk, you'll want to do some or all of the following:

- mitigate the risk – lower the chance of it happening, or its impacts if it does happen
- monitor the risk – keep track of the risk and factors influencing it
- manage the risk – prepare *contingency plans* for what to do if the risk eventuates

Risk mitigation

- Risk mitigation (avoidance) is the primary strategy and is achieved through a plan
- An example: risk of high staff turnover

Risk mitigation (2)

Mitigation actions might include:

- Meet with current staff to determine causes for turnover (e.g., poor working conditions, low pay, competitive job market)
- Mitigate those causes that are under our control before the project starts
- Once the project commences, assume turnover will occur and develop techniques to ensure continuity when people leave
- Organize project teams so that information about each development activity is widely dispersed
- Define documentation standards and establish mechanisms to ensure that documents are developed in a timely manner
- Conduct peer reviews of all work (so that more than one person is “up to speed”)
- Assign a backup staff member for every critical technologist

Risk monitoring and management

- During risk monitoring, the project manager monitors factors that may provide an indication of whether a risk is becoming more or less likely
- Risk management and contingency planning assume that mitigation efforts have failed and that the risk has become a reality

Safety risks and hazards

- Note that risk is not limited to the software project itself (i.e. development through to delivery)
- Risks can occur after the software has been delivered to the user
- Software is used to manage all sorts of systems that may pose a hazard – running nuclear plants, heart pacemakers, dialysis pumps, plane navigation systems, etc.
- There is a whole field of *software safety and hazard analysis* – the aim here is to identify hazards, and put in place measures to eliminate or control them.

The RMMM plan

- An RMMM plan may be a part of the software development plan or may be a separate document
- Once RMMM has been documented and the project has begun, the risk mitigation and monitoring steps begin
    - Risk mitigation is a problem avoidance activity
    - Risk monitoring is a project tracking activity
- Risk monitoring has three objectives
    - To assess whether predicted risks do, in fact, occur
    - To ensure that risk aversion steps defined for the risk are being properly applied
    - To collect information that can be used for future risk analysis
- The findings from risk monitoring may allow the project manager to ascertain what risks caused which problems throughout the project

Some principles of risk management

- Maintain a global perspective
  - View software risks within the context of the business problem being solved
- Take a forward-looking view
  - Think about risks that may arise in the future; establish contingency plans
- Encourage open communication
  - Encourage stakeholders and users to point out risks at any time
- Integrate risk management
  - Integrate consideration of risk into the software process
- Emphasize a continuous process of risk management
  - Modify and/or add risks as more becomes known
- Develop a shared product vision
  - This facilitates better risk identification and assessment
- Encourage teamwork when managing risk
  - Pool the skills and experience of all stakeholders when conducting risk management activities

Problems that can arise with risk management

- Lack of clear responsibility.
- If there isn't clear *responsibility* – i.e. actual consequences for someone – for identifying and managing risks, then using a checklist can become a "check-a-box" activity.
    - Something annoying management asks us to do before we can get started on the project
- (Interesting related reading: Xie et al, "Why do programmers make security errors?". It seems many programmers regard security as "someone else's problem".)

Problems that can arise with risk management (2)

- Optimism bias: on the whole, people tend to be over-optimistic about how long tasks will take, and believe that *they* are at a lesser risk of experiencing a negative event compared to others.
  - (“Well, yes, *other* projects always seem to run over-schedule. But not *this* one!”)
- How long did you estimate your last programming assignment would take? Did you over- or under-estimate?
- We seem to do this, even when we *know* things can go wrong; and it makes us bad at identifying risks, or not discounting them when we have identified them.

Problems that can arise with risk management (3)

- *Checklists*, which we have covered, are one way of dealing with this – they explicitly draw your attention to a risk
- Nobel-prize-winning economist Richard Thaler wants you to know about another: the "pre-mortem"
- Invented by applied psychologist Gary Klein
- We all seem to have excellent "20/20 hindsight"; so "assume we are at some time in the future when the plan has been implemented, and the outcome was a disaster. Write a brief history of that disaster."

Summary

- Whenever much is riding on a software project, common sense dictates risk analysis
  - Yet, most project managers do it informally and superficially, if at all
- However, the time spent in risk management results in
  - Less upheaval during the project
  - A greater ability to track and control a project
  - The confidence that comes with planning for problems before they occur