

 POLITECNICO DI MILANO

Dipartimento di  
Elettronica e Informazione

Planning and Managing Software Projects 2013-14  
Class 14

# Risk Management

**Emanuele Della Valle**  
<http://emanueledellavalle.org>

- These slides are largely based on Prof. John Musser class notes on “Principles of Software Project Management”
  - Original slides are available at <http://www.projectreference.com/>
- Reuse and republish permission was granted

- Problems that haven't happened yet
- Why is it hard?
- Some are wary of bearing bad news
  - No one wants to be the messenger
  - Or seen as “a worrier”
- You need to define a strategy early in your project

- Identification, Analysis, Control
- Goal: avoid a crisis
- Risk Mgmt. vs. Project Mgt.
  - For a specific vs. all projects
  - Proactive vs. reactive

- Project Risk
  - Characterized by:
    - Uncertainty ( $0 < \text{probability} < 1$ )
      - NOTE: If the probability is high, you may have planned the project in a wrong way.
    - An associated loss (money, life, reputation, etc)
    - Manageable – some action can control it
- Risk Exposure
  - Product of probability and potential loss
- Problem
  - A risk that has materialized

**TABLE 3-1** FMEA/CIL Criticality Classification

Criticality Category	Potential Effect of Failure
1	Loss of life or vehicle
1R	Redundant hardware element, failure of which could cause loss of life or vehicle
2	Loss of mission
2R	Redundant hardware element, failure of which could cause loss of mission
3	All others
For Ground Support Equipment only:	
1S	Failure of a safety or hazard monitoring system to detect, combat, or operate when required and could allow loss of life or vehicle
2S	Loss of vehicle system

# NASA Risk Management for Space Shuttle

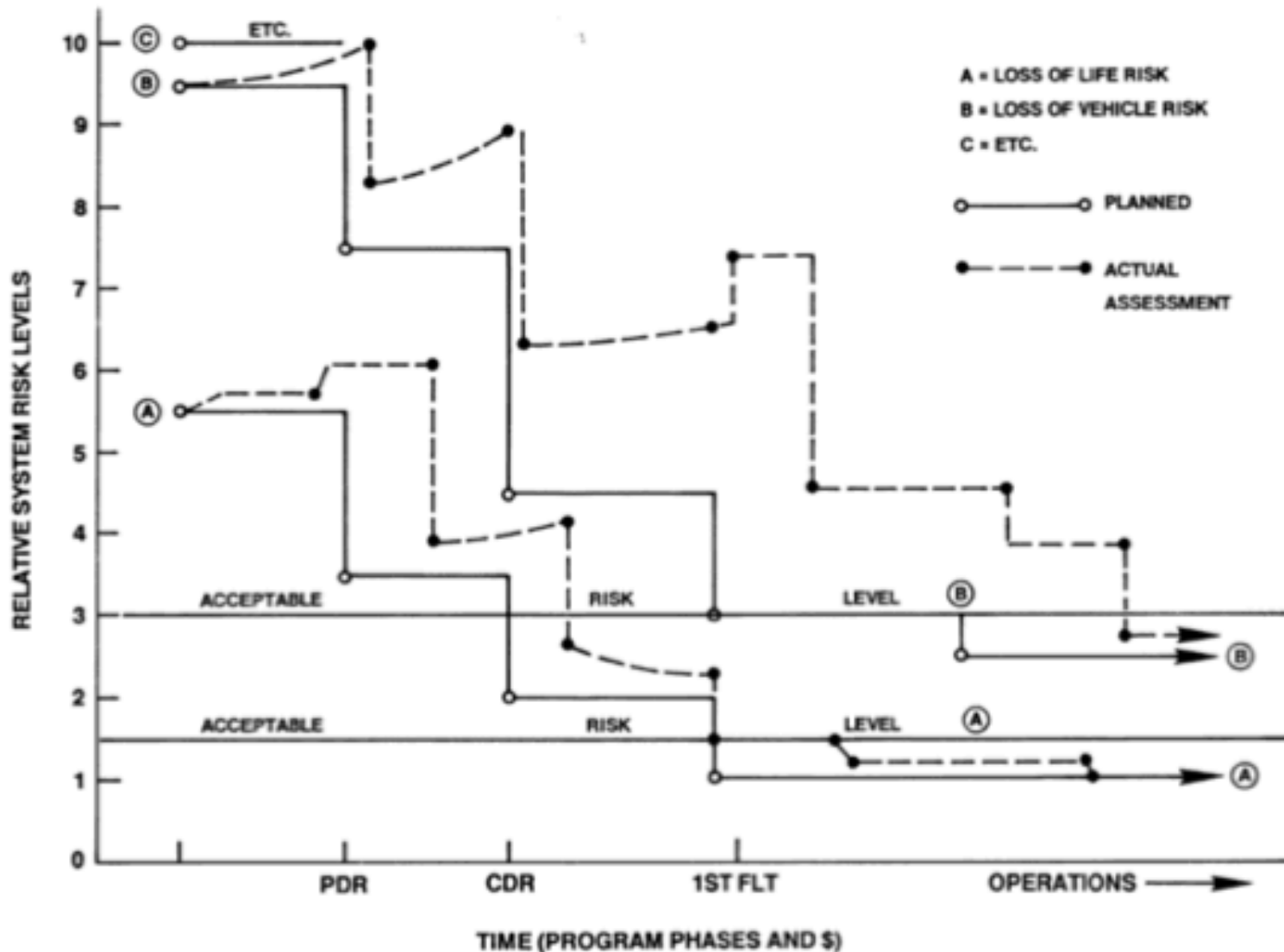


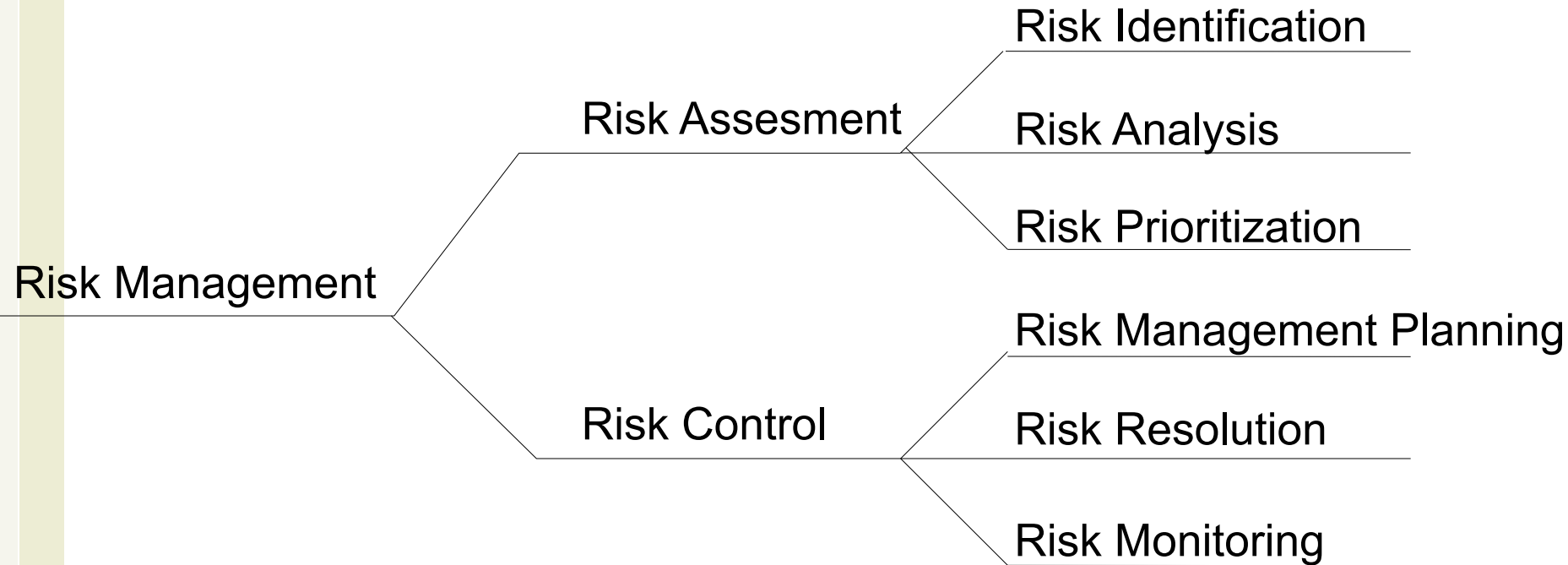
FIGURE 4-1 Conceptual diagram of risk management involving iterative steps taken to achieve specified levels of acceptable risk.

- Schedule Risks
  - Schedule compression (customer, marketing, etc.)
- Cost Risks
  - Unreasonable budgets
- Requirements Risks
  - Incorrect
  - Incomplete
  - Unclear or inconsistent
  - Volatile
- Quality Risks
- Operational Risks
- Most of the “Classic Mistakes”
  - Classic mistakes are made more often



# Types of Unknowns

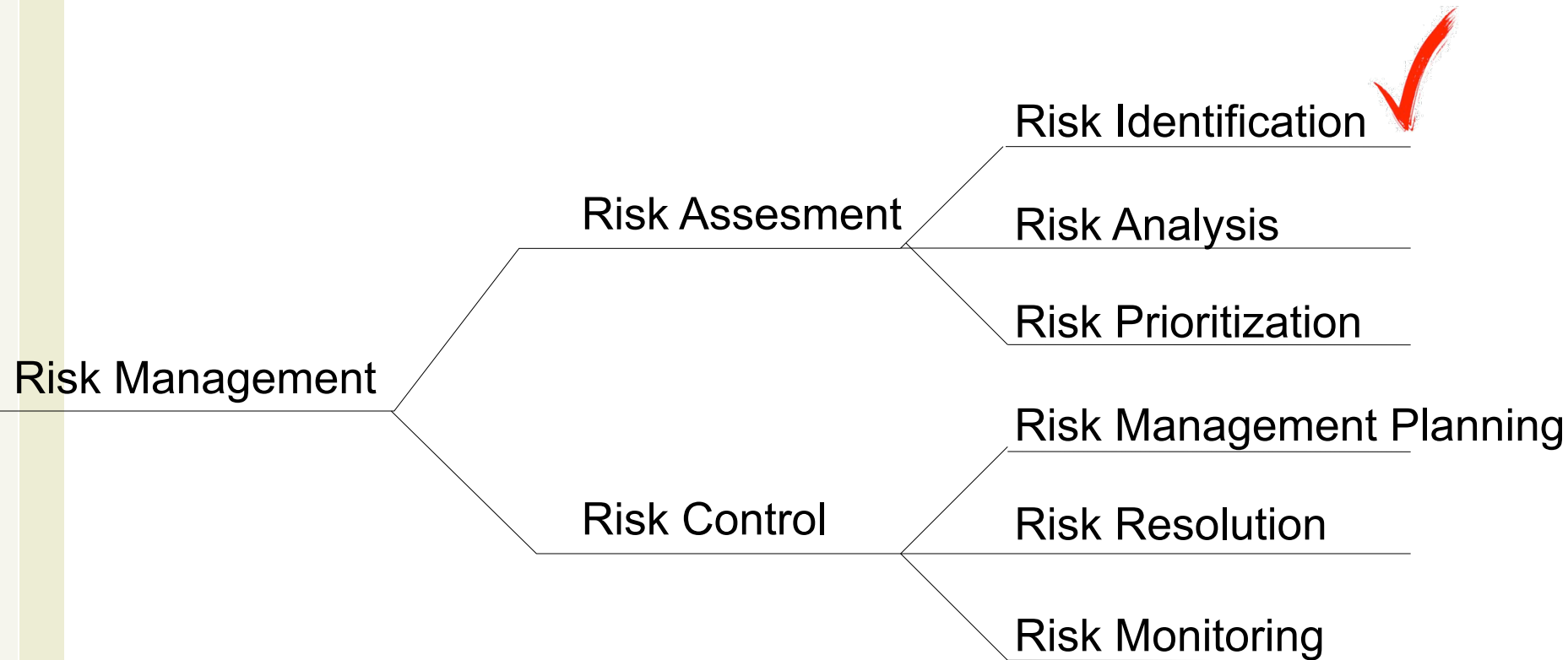
- Known Unknowns
  - Information you know someone else has
- Unknown Unknowns
  - Information that does not yet exist



[Source: “Software Risk Management”, Boehm, 1989]

# Risk Identification

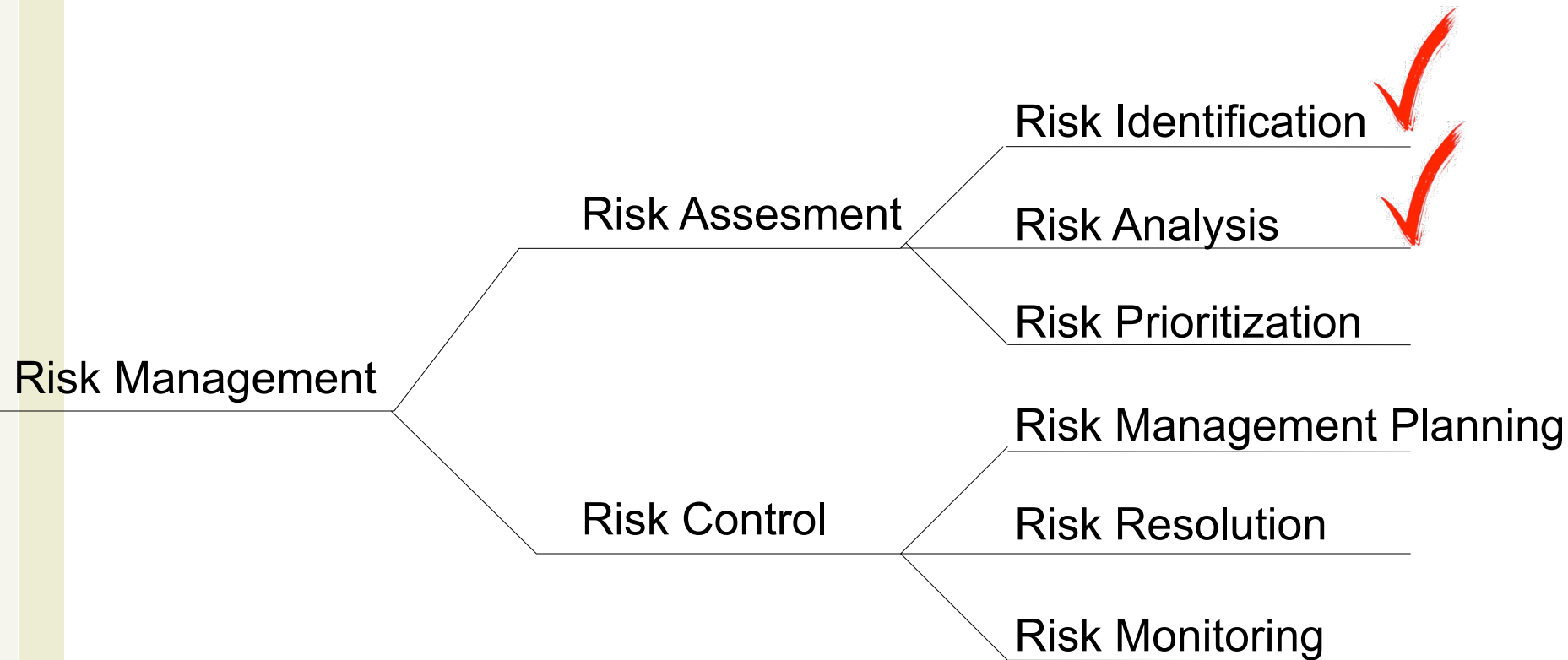
- Get your team involved in this process
  - Don't go it alone
- Produces a list of risks with potential to disrupt your project's schedule (but also budget, quality, ...)
- Use a checklist or similar source to brainstorm possible risks
  - <http://www.construx.com/Complete List of Schedule Risks/>



[Source: “Software Risk Management”, Boehm, 1989]

- Determine impact of each risk
- Risk Exposure (RE)
  - $RE = \text{Probability of loss} * \text{size of loss}$
- Examples
  - risk is “Facilities not ready on time”
    - Probability is 25%, size is 4 weeks, RE is 1 week
  - risk is “Inadequate design – redesign required”
    - Probability is 15%, size is 10 weeks, RE is 1.5 weeks
- Statistically are “expected values”
- Sum all RE’s to get expected overrun
  - Which is pre risk management

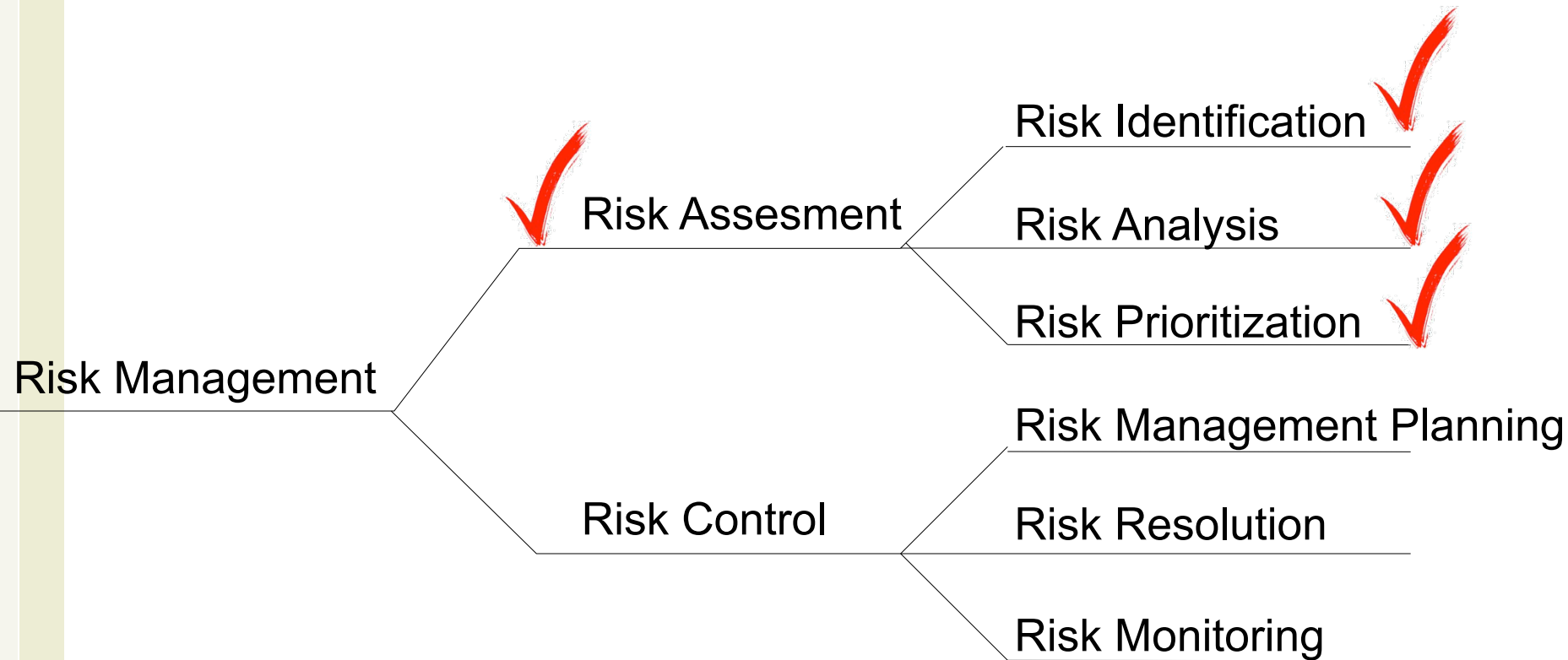
- Estimating size of loss
  - Loss is easier to see than probability
    - You can break this down into “chunks” (like WBS)
  
- Estimating probability of loss
  - Use team member estimates and have a risk-estimate review
  - Use Delphi or group-consensus techniques
  - Use gambling analogy” “how much would you bet”
  - Use “adjective calibration”:
    - highly likely
    - probably
    - improbable
    - unlikely
    - highly unlikely



[Source: “Software Risk Management”, Boehm, 1989]

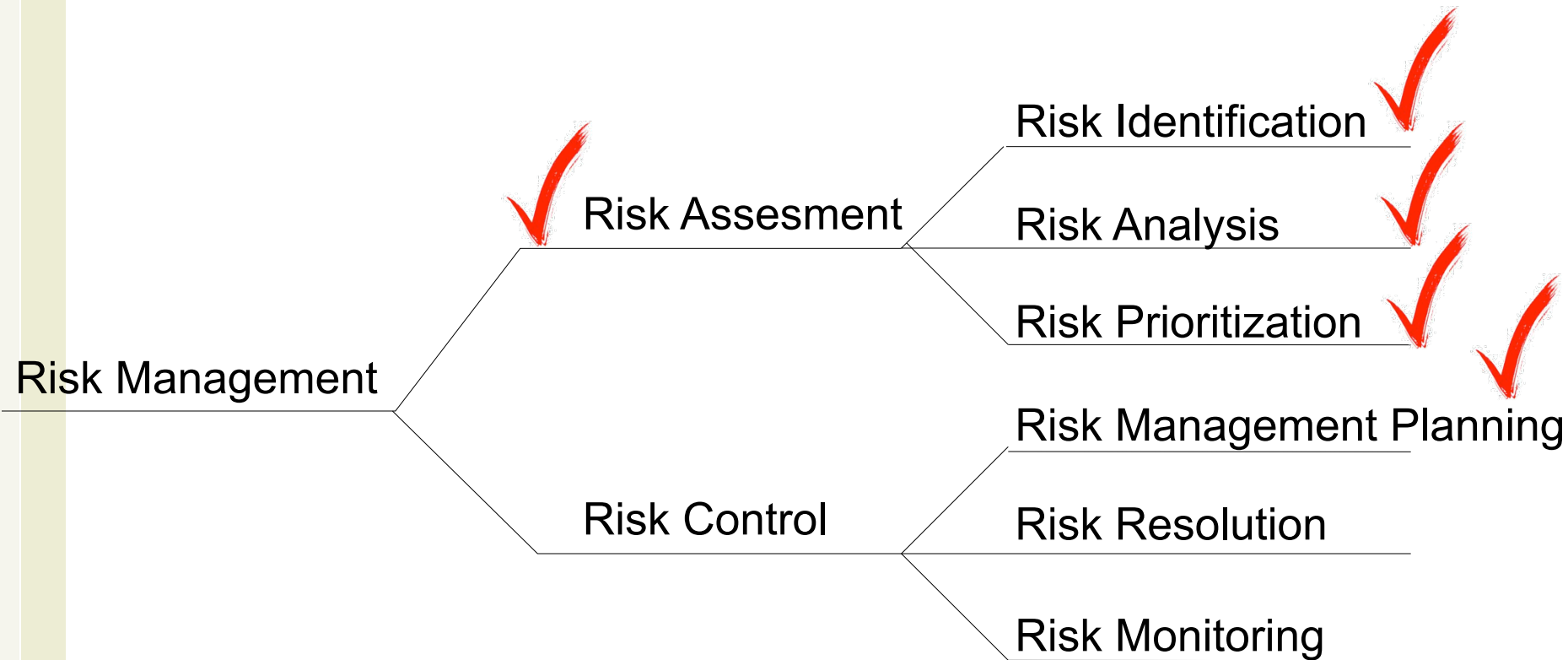
- Remember the 80-20 rule
- Often want larger-loss risks higher
  - Or higher probability items
- Possibly group ‘related risks’
- Helps identify which risks to ignore
  - Those at the bottom





[Source: “Software Risk Management”, Boehm, 1989]

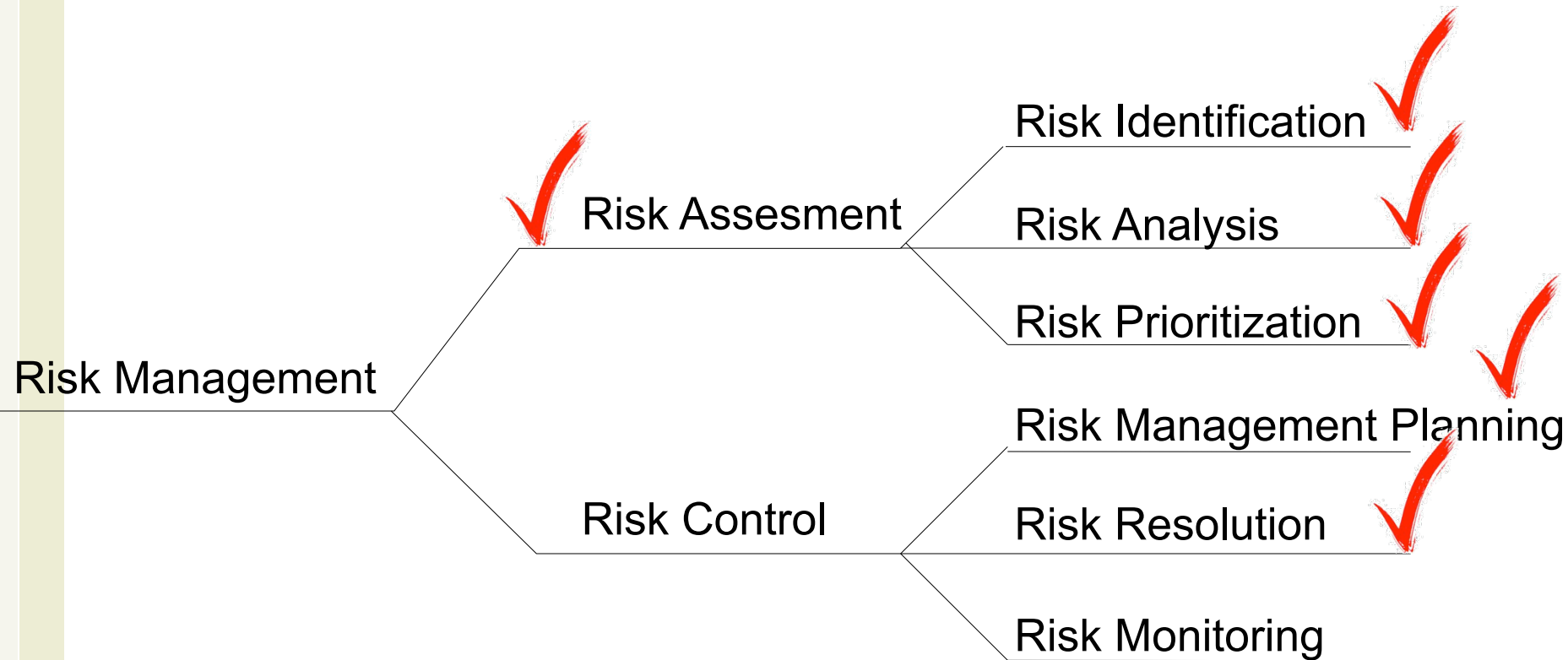
- McConnell's example
  - [http://www.construx.com/Thought\\_Leadership/Books/Survival\\_Guide/Resources\\_By\\_Chapter/Sample\\_Risk\\_Management\\_Plan/](http://www.construx.com/Thought_Leadership/Books/Survival_Guide/Resources_By_Chapter/Sample_Risk_Management_Plan/)



[Source: “Software Risk Management”, Boehm, 1989]

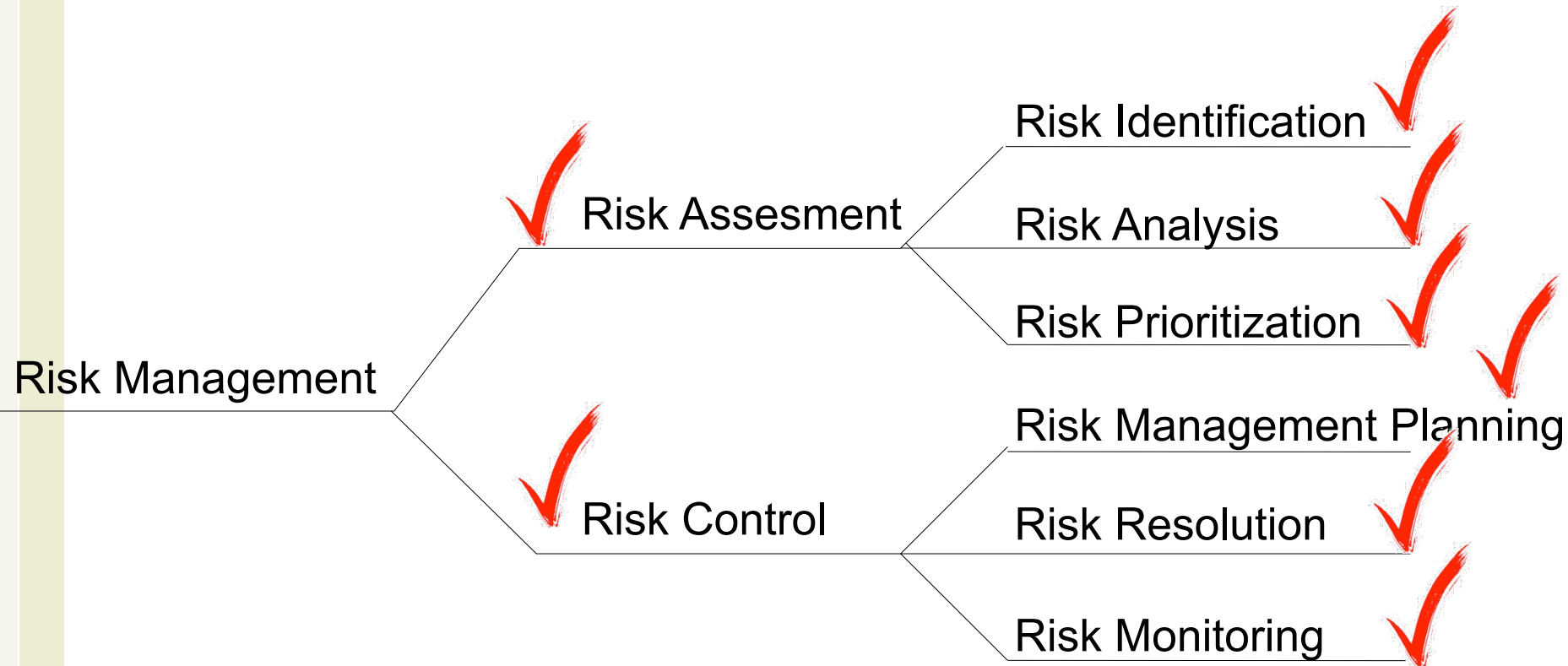
- Risk Avoidance
  - Don't do it
  - Scrub from system
- Risk Assumption
  - Don't do anything about it
  - Accept that it might occur
  - But still watch for it
- Problem control
  - Develop contingency plans
  - E.g., allocate extra test resources
- Risk Transfer
  - To another part of the project (or team)
  - Move off the critical path at least

- Knowledge Acquisition
  - Investigate
    - Ex: do a prototype
  - Buy information or expertise about it
  - Do research



[Source: “Software Risk Management”, Boehm, 1989]

- Top 10 Risk List
  - Rank
  - Previous Rank
  - Weeks on List
  - Risk Name
  - Risk Resolution Status
- A low-overhead best practice
- Interim project post-mortems
  - After various major milestones
- McConnell's example
  - [http://www.construx.com/Thought\\_Leadership/Books/Survival\\_Guide/Resources\\_By\\_Chapter/Sample\\_Top\\_10\\_Risks\\_List/](http://www.construx.com/Thought_Leadership/Books/Survival_Guide/Resources_By_Chapter/Sample_Top_10_Risks_List/)



[Source: “Software Risk Management”, Boehm, 1989]



- Don't be afraid to convey the risks
- Use your judgment to balance
  - Sky-is-falling whiner vs. information distribution

- A risk-reduction technique
- Use of small goals within project schedule
  - One of McConnell's Best Practices (Ch. 27)
- Fine-grained approach to plan & track
- Reduces risk of undetected project slippage
- Pros
  - Enhances status visibility
  - Good for project recovery
- Cons
  - Increase project tracking effort

- Can be used throughout the development cycle
- Works with hard-to-manage project activities or methods
  - Such as with evolutionary prototyping
- Reduces unpleasant surprises
- Success factors
  - Overcoming resistance from those managed
  - Staying true to ‘miniature’ nature
- Can improve motivation through achievements

- Requires a detailed schedule
- Have early milestones
- McConnell says 1-2 days
  - Longer is still good (1-2 weeks)
- Encourages iterative development
- Use binary milestones
  - Done or not done (100%)

# Questions?