The University of Western Australia
School of Computer Science and Software Engineering

CITS5502 Software Processes

Lecture 5

Risk Management

Key concepts

- Risk Exposure as Probability times Cost
  - Risk Leverage as reduction in Risk Exposure per unit cost
- Area and activities of Risk and Hazard Analysis
  - Theoretical model and underlying assumptions
    - Non-independence of risk factors
    - Dependence on underlying conditions
    - Non-linear relationship with project size, duration
  - Failsafing as falling back to a known, default, safe state
- Boehm's top ten software risk items
- Analysis of Risks
  - Identification, Estimation, Evaluation, Management of risks

Key concepts (cont.)

- Example of the quantification of software development risks
- Australian Standard AS4360 on Risk Management
- Strategies for risk minimization – reduce probability and/or impact monitoring, insurance, contingency plans, disaster recovery plans

Risk Management

- The word '**Risk**' comes from the Italian *risicare* [to dare]
- The word '**Hazard**' comes from the Arabic *al zahr* [dice]

- Risk management requires a careful analysis of the ways in which changes in conditions affect final outcomes.

Risk Management (cont.)

- For example, consider these three scenarios regarding driving conditions:
  1. In countries with hazardous driving conditions (e.g., Canada, Scotland, Scandinavia), as Summer turns to Winter, do road fatalities go up or down?
  2. As engineers build safer cars (fitted with ABS, seat belts, airbags), do road fatalities go up or down?
  3. As people buy more large, 4WD, off-road vehicles, do road fatalities go up or down?

Types of Software Engineering Risk

- There are three types of risk affecting the production of complex software systems. All three lead to unhappy customers and the possibility of project failure.

1. Methodological risk.

   This type of risk is concerned with the loss due to non-optimal management decisions in the use of a process. An example would be “When should we stop analysis and start design and coding?”

Types of Software Engineering Risk (cont.)

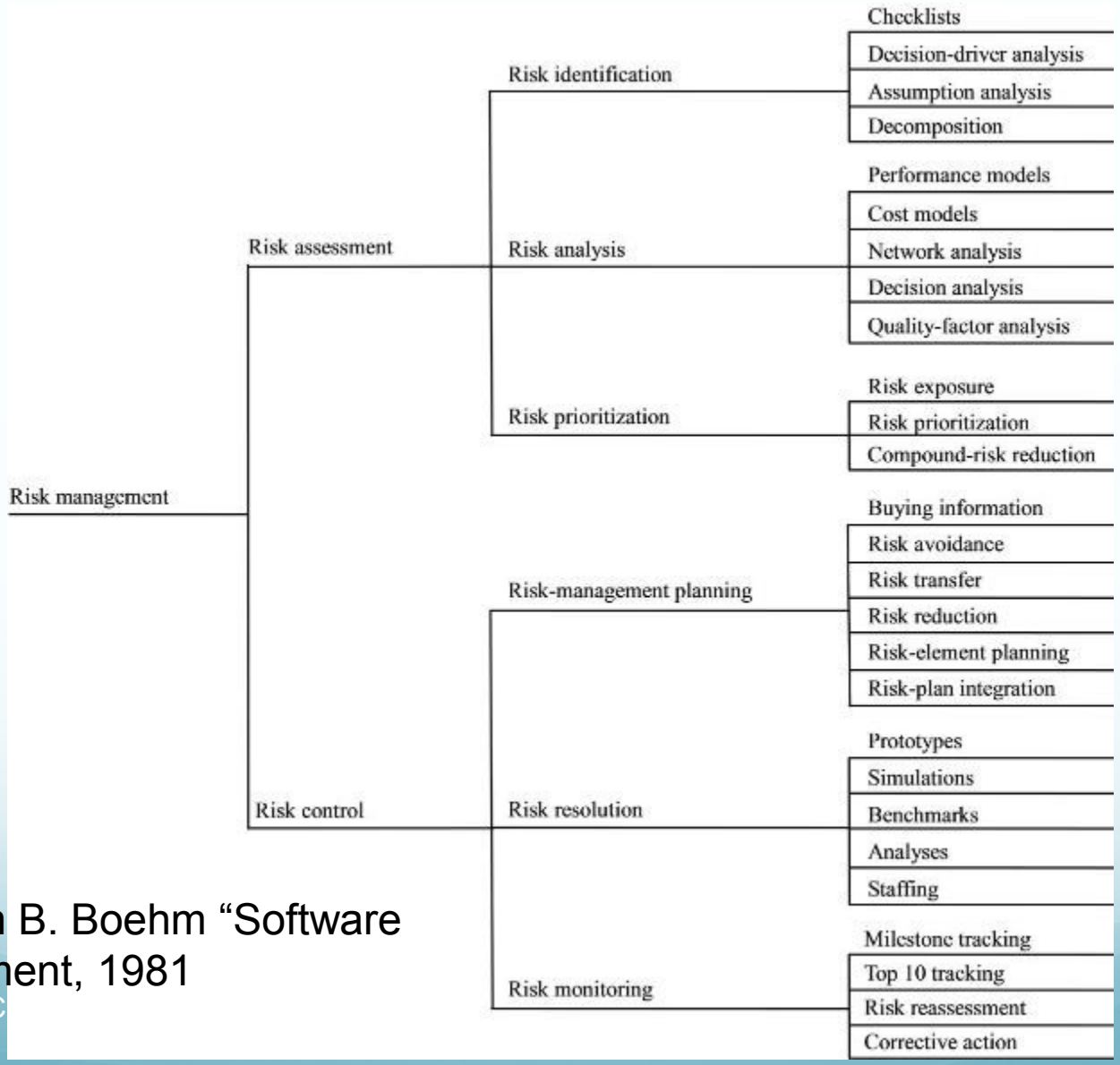
2. System failure risk.

   Defects in the software (Errors, Faults, Failures) lead to sudden loss of system – different to hardware which often 'degrades gracefully'. Covered by Reliability models.

3. Project risk.

   (See Boehm's top-ten list of risk items) Project risks are rarely sudden – they go through a number of stages (warning signs).

Software Risk Management Steps



Extracted from B. Boehm “Software Risk Management, 1981

A Prioritized top-ten List of Software Risk Items

Risk item	Risk management techniques
1. Personnel shortfalls	Staff with top talent, job matching; teambuilding; morale building; cross-training; pre-scheduling key people
2. Unrealistic schedules and budgets	Detailed, multisource cost and schedule estimation; design to cost; incremental development; software reuse; requirements scrubbing
3. Developing the wrong software functions	Organization analysis; mission analysis; ops-concept formulation; user surveys; prototyping; early users' manuals
4. Developing the wrong user interface	Task analysis; prototyping; scenarios; user characterization (functionality, style, workload)
5. Gold plating	Requirements scrubbing; prototyping; cost-benefit analysis; design to cost

A Prioritized top-ten List of Software Risk Items (cont.)

Risk item	Risk management techniques
6. Continuing stream of requirement changes	High change threshold; information hiding; incremental development (defer changes to later increments)
7. Shortfalls in externally furnished components	Benchmarking; inspection; reference checking; compatibility analysis
8. Shortfalls in externally performed tasks	Reference checking; pre-award audits; award-fee contracts; competitive design or prototyping; teambuilding
9. Real-time performance shortfalls	Simulation; benchmarking; modelling; prototyping; instrumentation; tuning
10. Straining computer-science capabilities	Technical analysis; cost-benefit analysis; prototyping; reference checking

Common Risks and Possible Corrective

Cause	Action
Lack of adequate definition of computer resource functional, interface, support, or performance requirements prior to structuring the program. Poorly defined, complex, or untestable intra- or inter-system interfaces, including human interfaces. Lack of stability in computer resource requirements during development.	System engineering techniques such as functional analyses, simulation, mathematical modelling, correctness proofs, and trade-off analyses. Incremental development strategies which tackle large, complex, and poorly understood requirements in smaller, more manageable parts.

Extracted from “Test and Evaluators’ Management Guide”,
US Defence Systems Management College 1988

Common Risks and Possible Corrective

Cause	Action
Lack of government visibility into the contractor's software development effort.	Rigorous application of traditional cost, schedule, and performance tracking techniques with careful attention to earned value progress against measureable milestones. Since these techniques are almost always driven by the WBS, visibility of critical and high risk computer resources is a primary criterion for determining the appropriate level within the WBS for these components of the system. Use of a risk tracking system to collect data on the status of identified high risk items. The output of this system should be a standard part of periodic reviews Use of independent verification and validation.
Performance requirements that push the state of the art	Prototyping or duplicate development of key algorithms, concepts, and components.

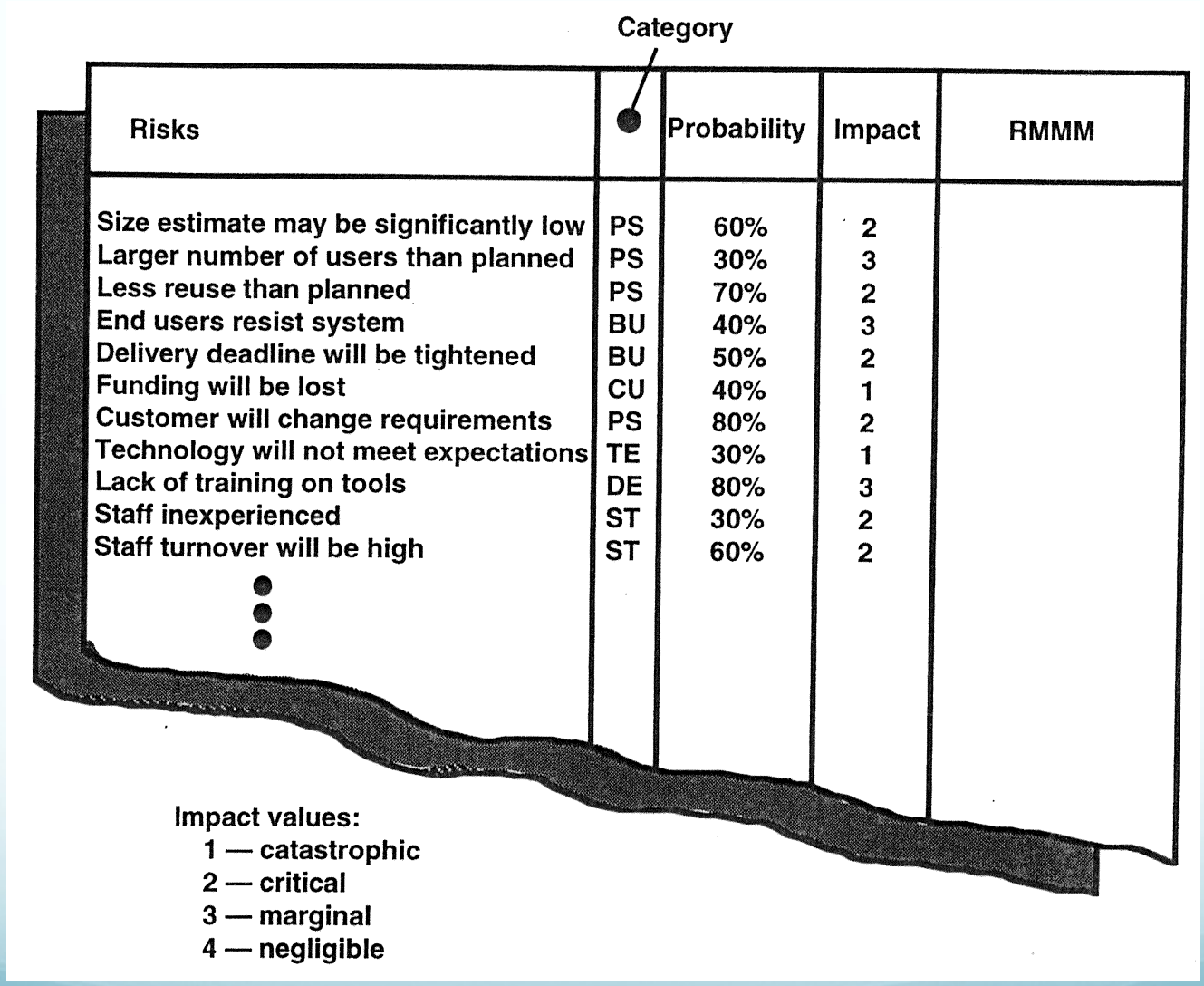
Common Risks and Possible Corrective

Cause	Action
Inaccurate, poorly defined, or non-existent cost and schedule estimates for computer resource development.	Prototyping of duplicate development of key algorithms, concepts, and components
Inadequate developer and acquisition manager capability or capacity for software development. Inadequate, immature, or poorly integrated software development tools (e.g., compilers, linkers, loaders) & programming support environment.	Reviews of offerors' sites to assess capability and capacity of development personnel, management structure and procedures, and facilities.
Lack of adequate spare computer hardware capacity (e.g., processor speed, memory, input/output, and secondary storage).	Early planning for spare capacity during development and support phases of the lifecycle; periodic reviews of capacity allocation, and projection of requirements trends.
Undefined or poorly defined software support concepts.	Rigorous adherence to the separation of mission software and system software into separate CSCIs.

Risk Projection

- Establish a scale that reflects the perceived likelihood of the risk (probability is often used).
- Define the consequences of the risk.
- Estimate the impact of the risk on the project and/or the product (usually on scale: 1 to 10).
- Note the overall accuracy of the estimates.

Building a 'Risk Table'



Risks	Category	Probability	Impact	RMMM
Size estimate may be significantly low	PS	60%	2	
Larger number of users than planned	PS	30%	3	
Less reuse than planned	PS	70%	2	
End users resist system	BU	40%	3	
Delivery deadline will be tightened	BU	50%	2	
Funding will be lost	CU	40%	1	
Customer will change requirements	PS	80%	2	
Technology will not meet expectations	TE	30%	1	
Lack of training on tools	DE	80%	3	
Staff inexperienced	ST	30%	2	
Staff turnover will be high	ST	60%	2	

Impact values:
1 — catastrophic
2 — critical
3 — marginal
4 — negligible

Extracted from *Software Engineering: A Practitioner's Approach*, 4/e, McGraw-Hill, 1997. For use in University teaching.

Failure Modes and Effects Criticality Analysis

Failure Modes and Effects Criticality Analysis

Subsystem _____ Prepared by _____ Date _____

Item	Failure Modes	Cause of Failure	Possible Effects	Prob.	Level	Possible Action to Reduce Failure Rate or Effects
Motor Case	Rupture	a. Poor workmanship b. Defective materials c. Damage during transportation d. Damage during handling e. Over-pressurization	Destruction of missile	0.0006	Critical	Close control of manufacturing processes to ensure that workmanship meets prescribed standards. Rigid quality control of basic materials to eliminate defectives. Inspection and pressure testing of completed cases. Provision of suitable packaging to protect motor during transportation.

A sample FMECA

Extracted from Leveson “Safeware”.

Recommended Reading

- Sommerville: Sections on “Risk Management”
- Pressman: Chapter on “Risk Management”