Computer Science
UNIVERSITY OF TORONTO

lecture 6: risk management

csc302h
winter 2014

Computer Science
UNIVERSITY OF TORONTO

administrative

- a1 is due tomorrow, before midnight
  - submit a single file, prefer pdf
  - submit on cdf, with the submit command
    - `submit -c csc302h -a a1 -f a1.pdf`
  - only a single member of each group should submit the report
  - email your report to me _only_ in emergency situation.
- a1 interviews start at 10 am sharp!
  - you _must_ have your peer eval form filled out before the interview

Computer Science
UNIVERSITY OF TORONTO

recap from last time

- showed a few real examples of (uml) diagrams for modeling
- reviewed some sdlc models
- talked a bit about the dichotomy:
  - agile vs. [traditional | planning-based | sturdy | disciplined]
  - what do they share? how are they different?
- deployment mechanism can sometimes make one method more suitable than another (ex. saas & agile)
- example feature workflow

Computer Science
UNIVERSITY OF TORONTO

recap from last time (2)

- discussed details of a couple models:
  - waterfall (& ugly Gantt charts)
  - prototyping lifecycle
    - how it fits with other models ("spikes")
  - phased lifecycle – really pipelined waterfall
  - spiral model – repeated waterfall, sometimes with more complexity (steps) in each "orbit"
  - rational unified process (rup)
  - scrum
  - xp

Computer Science
UNIVERSITY OF TORONTO

recap from last time (3)

- why do we need a process? helps to achieve quality
- listed 21 agile practices:
  - was it the whole list?
  - do other, non-agile, processes share some of these practices?
- which software process is best?
  - depends on the context of the project

Computer Science
UNIVERSITY OF TORONTO

risk management in software projects

Managing Risk

General ideas about Risk

Risk Management

- Identifying Risks
- Assessing Risks

Case Study:

- Mars Polar Lander

Risk Management

About Risk

- Risk is "the possibility of suffering loss"
- Risk itself is not bad, it is essential to progress
- The challenge is to manage the amount of risk

Two Parts:

- Risk Assessment
- Risk Control

Useful concepts:

- For each risk: Risk Exposure
  - RE = p(unsatisfactory outcome) X loss(unsatisfactory outcome)
- For each mitigation action: Risk Reduction Leverage
  - RRL = (RE_{before} - RE_{after}) / cost of intervention

Computer Science
UNIVERSITY OF TORONTO

risk reduction leverage ex.

- what does RRL < 1 mean?
- which choice below is best?
- what if RE_{after} must be < $50k?

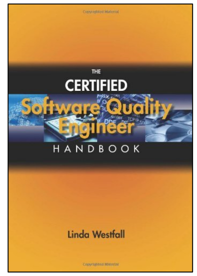


Table 17.8 Risk reduction leverage—example.

Risk #	$Probability_{Before}$	$Loss_{Before}$	RE_{Before}
143	25%	$300K	$75K

Alternative	$Probability_{After}$	$Loss_{After}$	RE_{After}	Cost	RRL
1	4%	$300K	$12K	$150K	0.4
2	25%	$180K	$45K	$20K	1.5
3	20%	$300K	$60K	$2K	7.5

Risk Assessment

Quantitative:

Measure risk exposure using standard cost & probability measures

Note: probabilities are rarely independent

Qualitative:

Develop a risk exposure matrix

Eg for NASA:

Undesirable outcome	Likelihood of Occurrence: Very likely	Possible	Unlikely
(5) Loss of Life	Catastrophic	Catastrophic	Severe
(4) Loss of Spacecraft	Catastrophic	Severe	Severe
(3) Loss of Mission	Severe	Severe	High
(2) Degraded Mission	High	Moderate	Low
(1) Inconvenience	Moderate	Low	Low

Top SE risks (with countermeasures)

Source: Adapted from Boehm, 1989

Personnel Shortfalls
- use top talent
- team building
- training

Unrealistic schedules/budgets
- multisource estimation
- designing to cost
- requirements scrubbing

Developing the wrong software functions
- better requirements analysis
- organizational/operational analysis

Developing the wrong User Interface
- prototypes, scenarios, task analysis

Gold Plating
- requirements scrubbing
- cost benefit analysis
- designing to cost

Continuing stream of requirements changes
- high change threshold
- information hiding
- incremental development

Shortfalls in externally furnished components
- early benchmarking
- inspections, compatibility analysis

Shortfalls in externally performed tasks
- pre-award audits
- competitive designs

Real-time performance shortfalls
- targeted analysis
- simulations, benchmarks, models

Straining computer science capabilities
- technical analysis
- checking scientific literature

Case Study: Mars Polar Lander

Launched

3 Jan 1999

Mission

Land near South Pole

Dig for water ice with a robotic arm

Fate:

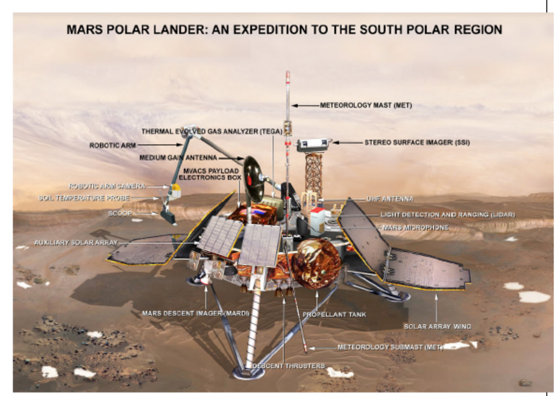
Arrived 3 Dec 1999

No signal received after initial phase of descent

Cause:

Several candidate causes

Most likely is premature engine shutdown due to noise on leg sensors



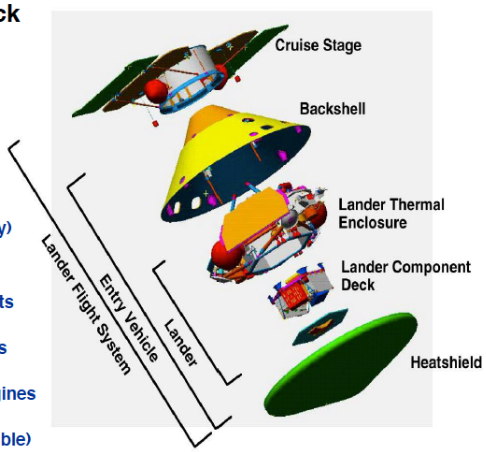
What happened?

Investigation hampered by lack of data

- spacecraft not designed to send telemetry during descent
- This decision severely criticized by review boards

Possible causes:

- Lander failed to separate from cruise stage (plausible but unlikely)
- Landing site too steep (plausible)
- Heatshield failed (plausible)
- Loss of control due to dynamic effects (plausible)
- Loss of control due to center-of-mass shift (plausible)
- Premature Shutdown of Descent Engines (most likely!)
- Parachute drapes over lander (plausible)
- Backshell hits lander (plausible but unlikely)



Premature Shutdown Scenario

Cause of error

- Magnetic sensor on each leg senses touchdown
- Legs unfold at 1500m above surface
- software accepts transient signals on touchdown sensors during unfolding

Factors

- System requirement to ignore the transient signals
- But the software requirements did not describe the effect
- Engineers present at code inspection didn't understand the effect
- Not caught in testing because:
- Unit testing didn't include the transients
- Sensors improperly wired during integration tests (no touchdown detected!)

Result of error

- Engines shut down before spacecraft has landed
- estimated at 40m above surface, travelling at 13 m/s
- estimated impact velocity 22m/s (spacecraft would not survive this)
- nominal touchdown velocity 2.4m/s

SYSTEM REQUIREMENTS

1) The touchdown sensors shall be sampled at 100-Hz rate. The sampling process shall be initiated prior to lander entry to keep processor demand constant. However, the use of the touchdown sensor data shall not begin until 12 meters above the surface.

2) Each of the 3 touchdown sensors shall be tested automatically and independently prior to use of the touchdown sensor data in the onboard logic. The test shall consist of two (2) sequential sensor readings showing the expected sensor status. If a sensor appears failed, it shall not be considered in the descent engine termination decision.

3) Touchdown determination shall be based on two sequential reads of a single sensor indicating touchdown.

FLIGHT SOFTWARE REQUIREMENTS

3.7.2.2.4.2 Processing

a. The lander flight software shall cyclically check the state of each of the three touchdown sensors (one at 100 Hz during EDL.

b. The lander flight software shall be able to cyclically check the touchdown event state with or without touchdown event generation enabled.

c. Upon enabling touchdown event generation, the lan flight software shall attempt to detect failed sens marking the sensor as bad when the sensor indicat "touchdown state on two consecutive reads.

d. The lander flight software shall generate the landin event based on two consecutive reads indicating touchdown from any one of the "good" touchdown sensors.

Adapted from the "Report of the Loss of the Mars Polar Lander and Deep Space 2 Missions -- JPL Special Review Board (Casani Report) - March 2000". See http://www.nasa.gov/newsinfo/marsreports.html

Lessons?

Documentation is no substitute for real communication

Software bugs hide behind other bugs
(full regression testing essential!)

Fixed cost + fixed schedule = increased risk

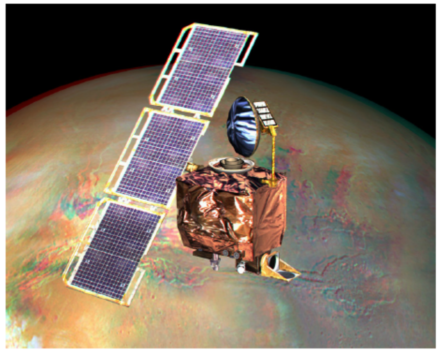
Case Study: Mars Climate Orbiter

Launched
- 11 Dec 1998

Mission
- interplanetary weather satellite
- communications relay for Mars Polar Lander

Fate:
- Arrived 23 Sept 1999
- No signal received after initial orbit insertion

Cause:
- Faulty navigation data caused by failure to convert imperial to metric units



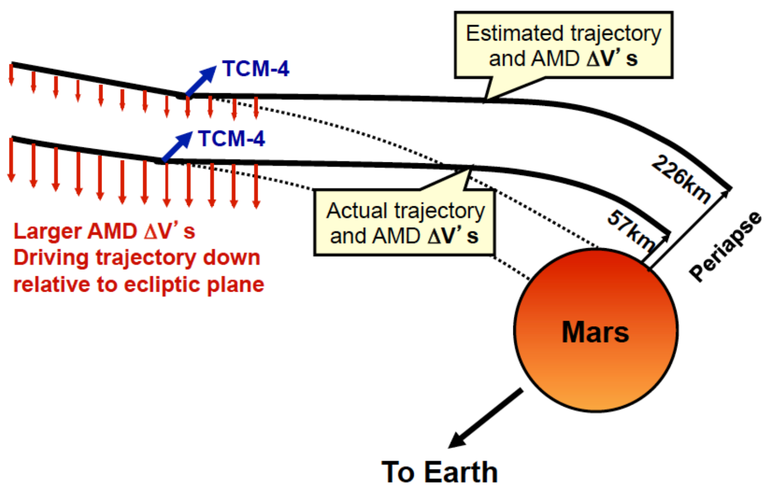
MCO Events

Locus of error
- Ground software file called "Small Forces" gives thruster performance data
  - data used to process telemetry from the spacecraft
- Angular Momentum Desaturation (AMD) maneuver effects underestimated
  - (by factor of 4.45)

Cause of error
- Small Forces Data given in Pounds-seconds (lbf-s)
- The specification called for Newton-seconds (N-s)

Result of error
- As spacecraft approaches orbit insertion, trajectory is corrected
  - Aimed for periapse of 226km on first orbit
- Estimates were adjusted as the spacecraft approached orbit insertion:
  - 1 week prior: first periapse estimated at 150-170km
  - 1 hour prior: this was down to 110km
  - Minimum periapse considered survivable is 85km
- MCO entered Mars occultation 49 seconds earlier than predicted
  - Signal was never regained after the predicted 21 minute occultation
  - Subsequent analysis estimates first periapse of 57km

MCO Navigation Error



Contributing Factors

For 4 months, AMD data not used (file format errors)
- Navigators calculated data by hand
- File format fixed by April 1999
- Anomalies in the computed trajectory became apparent almost immediately

Limited ability to investigate:
- Thrust effects measured along line of sight using doppler shift
- AMD thrusts are mainly perpendicular to line of sight

Poor communication
- Navigation team not involved in key design decisions
- Navigation team did not report the anomalies in the issue tracking system

Inadequate staffing
- Operations team monitoring 3 missions simultaneously (MGS, MCO and MPL)

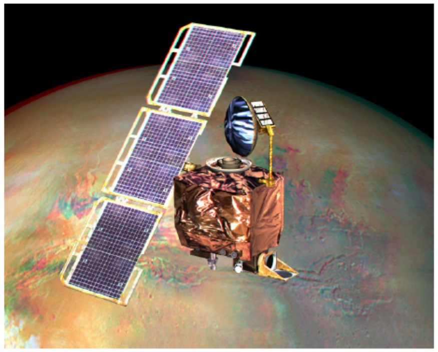
Operations Navigation team unfamiliar with spacecraft
- Different team from development & test
- Did not fully understand significance of the anomalies
- Surprised that AMD was performed 10-14 times more than expected

Inadequate Testing
- Software Interface Spec not used during unit test of small forces software
- End-to-end test of ground software was never completed
- Ground software considered less critical

Inadequate Reviews
- Key personnel missing from critical design reviews

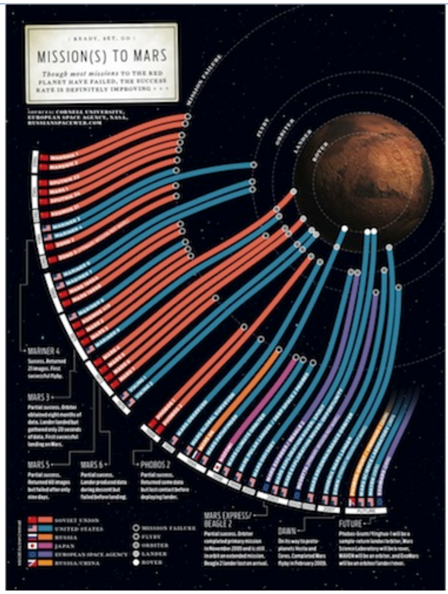
Inadquate margins…

Mars Climate Orbiter



Mars Global Surveyor

Mars is Hard!



down-to-earth examples

THE RISKS DIGEST

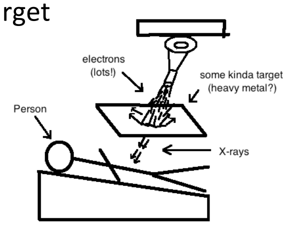
forum on risks to the public in computers & related systems

been around since 1985

http://catless.ncl.ac.uk/Risks/

down-to-earth examples (2)

- therac-25 from AECL, 1985-87
  - http://catless.ncl.ac.uk/Risks/3.11.html#subj1
  - http://en.wikipedia.org/wiki/Therac-25
- radiation therapy machine
  - two modes:
    - low dose, short period, electron-beam
    - megavolt x-ray therapy, collides high-dose, high-energy electron beam with target
- problem: could be made to operate w/o target in place!



Computer Science UNIVERSITY OF TORONTO

down-to-earth examples (3)

- a less tragic example...
- in 1995 an "*abandoned oil tank phone harasses ma woman for 6 months*"
  - http://catless.ncl.ac.uk/Risks/17.34.html#subj3.1
- old oil tank (???) rigged to call the oil company every 90 minutes when low
- configured with wrong number of poor unsuspecting woman
- pick up phone, say "*hello?*", no answer
- why did it take phone co. six months to trace? c'mon, really?

Lessons?

If it doesn't behave how you expect, it's not safe
(yes, really!)

If your teams don't coordinate, neither will their software
(See: Conway's Law)

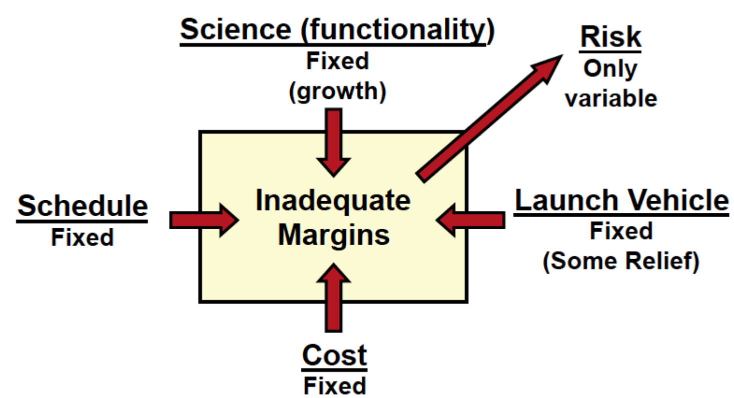
With software, everything is connected to everything else -- every subsystem is critical

Sidetrack: SNAFU principle

Full communication is only possible among peers; Subordinates are too routinely rewarded for telling pleasant lies, rather than the truth.

Not a good idea to have the IV&V teams reporting to the program office!!

Failure to manage risk



Adapted from MPIAT - Mars Program Independent Assessment Team Summary Report, NASA JPL, March 14, 2000.
See http://www.nasa.gov/newsinfo/marsreports.html

Principles of Risk Management

Source: Adapted from SEI Continuous Risk Management Guidebook

Global Perspective
- View software in context of a larger system
- For any opportunity, identify both:
  - Potential value
  - Potential impact of adverse results

Forward Looking View
- Anticipate possible outcomes
- Identify uncertainty
- Manage resources accordingly

Open Communications
- Free-flowing information at all project levels
- Value the individual voice
  - Unique knowledge and insights

Integrated Management
- Project management is risk management!

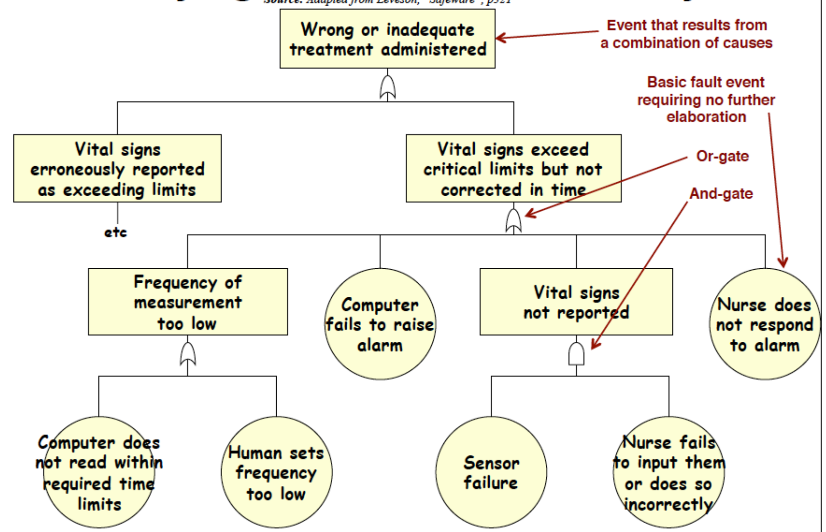
Continuous Process
- Continually identify and manage risks
- Maintain constant vigilance

Shared Product Vision
- Everybody understands the mission
  - Common purpose
  - Collective responsibility
  - Shared ownership
- Focus on results

Teamwork
- Work cooperatively to achieve the common goal
- Pool talent, skills and knowledge

Identifying Risks: Fault Tree Analysis

Source: Adapted from Leveson, "Safeware", p321



Continuous Risk Management

Source: Adapted from SEI Continuous Risk Management Guidebook

Identify:
- Search for and locate risks before they become problems
  - Systematic techniques to discover risks

Analyse:
- Transform risk data into decision-making information
- For each risk, evaluate:
  - Impact
  - Probability
  - Timeframe
- Classify and Prioritise Risks

Plan
- Choose risk mitigation actions

Track
- Monitor risk indicators
- Reassess risks

Control
- Correct for deviations from the risk mitigation plans

Communicate
- Share information on current and emerging risks

office hour in BA4237