Problems, Solutions and Requirements

Michael Jackson
jacksonma@acm.org

16th IEEE International
Requirements Engineering Conference
Barcelona
Thursday, September 11, 2008

Requirements engineering conferences

RE'95 & RE'08 stalwarts

Al Davis, Steve Easterbrook, Martin Feather, Stephen Fickas, Anthony Finkelstein, Orlena Gotel, Anthony Hall, Connie Heitmeyer, Matthias Jarke, Marina Jirotka, Axel van Lamsweerde, Julio Leite, Robyn Lutz, Nazim Madhavji, Neil Maiden, John Mylopoulos, Bashar Nuseibeh, Colin Potts, Bjorn Regnell, Kevin Ryan, Alistair Sutcliffe, Eric Yu …

… and me! … and you? … RE 2021?

… (I had omitted Annie Anton, Dan Berry, Pere Botella, Leah Goldin, …)

The first requirements engineer

An old problem

"Requirements Engineering is a new label attached to an old problem that has been with the software profession since its inception."

Michael Harrison and Pamela Zave; Foreword to RE'95 Proceedings

David Caminer

- The first requirements engineer
- Died June 19 2008, aged 92

J Lyons and Co

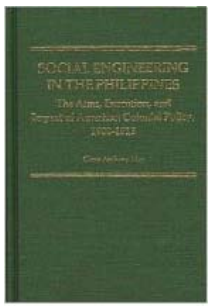
- Part-financed development of EDSAC
- Designed and built Leo I, Leo II, Leo III
- Supplies ordering system for teashops
  - First ran November 17, 1951
- Staff payroll system
- Whole system fully live in 1954

LEO 1

Requirements engineering

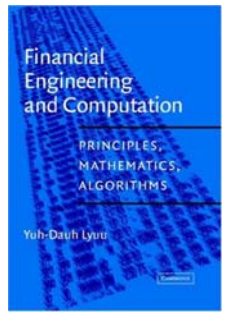
- What kind of engineering?
  - Are we like other engineers?
- Artifacts and problem worlds
  - What can go wrong?
- Learning from other engineers
  - Radical and normal design
- Specialisation
  - Artifacts: A missing dimension
- Software-intensive systems
  - Component structure
- It may be engineering …
  … but is it requirements?

What kind of engineering?



Social Engineering

Clinical Engineering

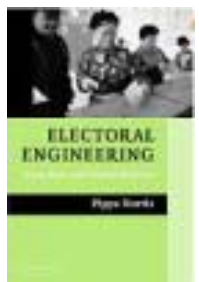


Financial Engineering

Psychological Engineering

Food Engineering

Fashion Engineering



Electoral Engineering

Political Engineering

Their kind of engineering

"... the design and construction of any artifact which transforms the physical world around us to meet some recognised need."

G F C Rogers

artifact

physical world (including human)

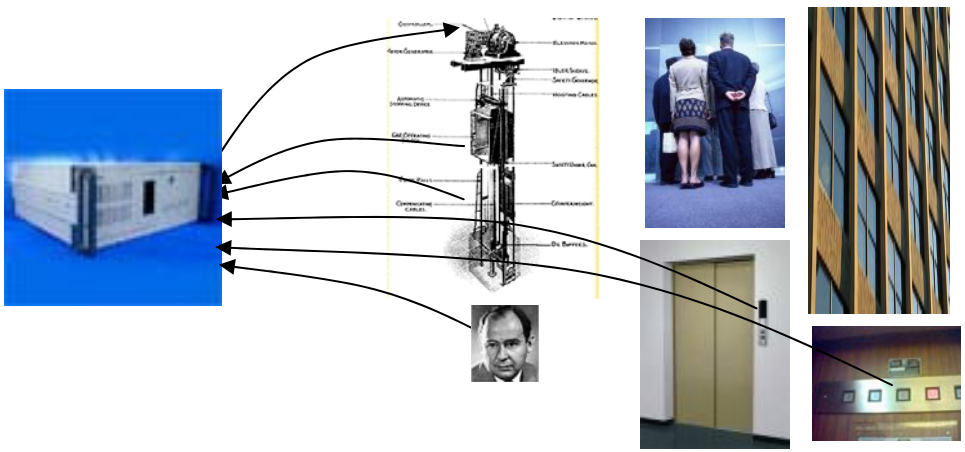
- moves people and baggage
- moves as driver directs
- moves on good roads
- moves on earth's surface
- protects from weather
- protects in a collision
- runs on portable power
- ... ...

recognised need

Our kind of engineering

"... the design and construction of any artifact which transforms the physical world around us to meet some recognised need."

G F C Rogers



- lift comes on demand
- goes to requested floor
- keeps passengers safe
- gives efficient service
- shows current location
- adjustable priorities ...
- ... ...

artifact

physical world (including human)

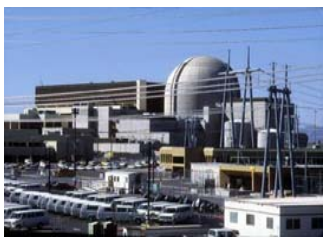
recognised need

Some of their engineering artifacts

Family Car

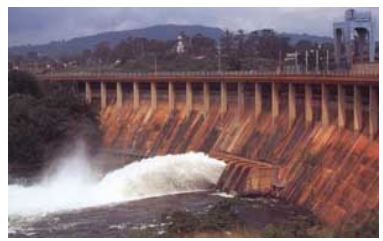
Ship

Disk Drive



Power Plant

Arch Bridge



Dam

Concorde

Bullet Train

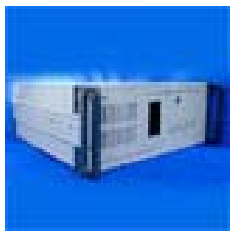
Chemical Plant

Some software engineering artifacts?


Software

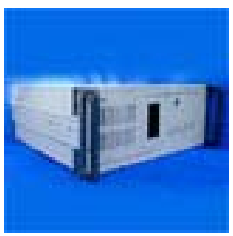

Software


Software


Software


Software


Software

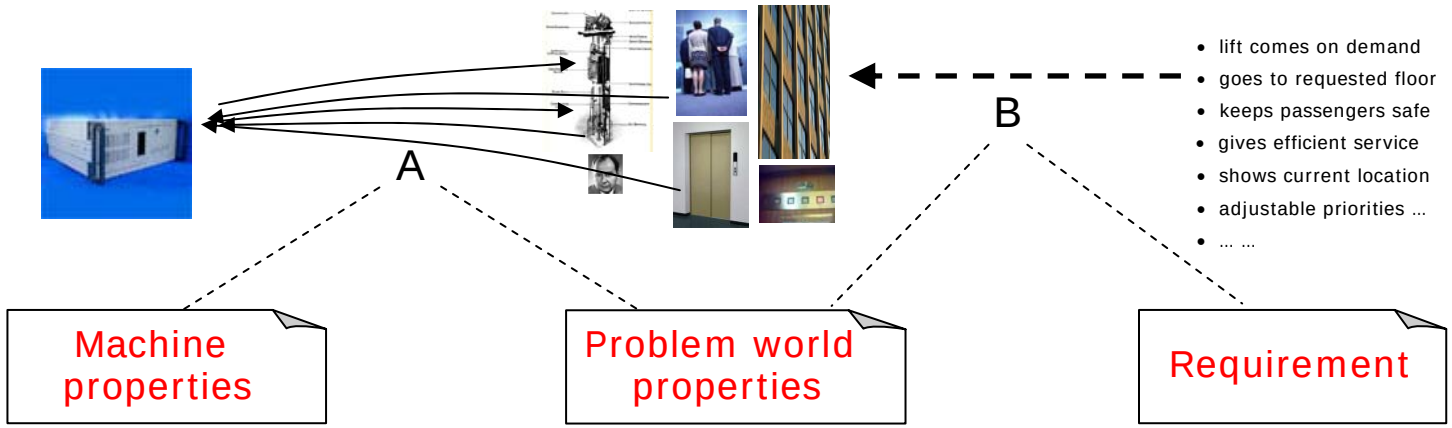

Software


Software


Software

The engineering artifact is the system

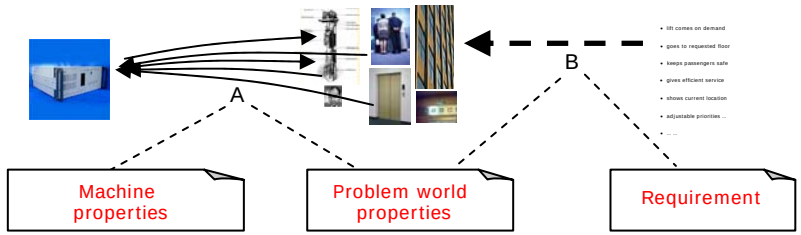
Requirements engineers know better!



- Our artifact is the system—machine + world + requirement
- The properties of the physical problem world are central
  - A: set of phenomena shared between machine and world
  - B: set of phenomena mentioned in the Requirement (A ≠ B)
- The purpose of the system is to monitor & control the world
  - We interact with the machine only through the world
- Different problem world properties ⇒ different systems

The engineering artifact is the system

How can we fail?



1. We didn't understand the requirements
   - Same priority scheduling applied to all lifts
2. We didn't understand the problem world
   - # call requests at floor F = # requesting users at F
3. The machine doesn't ensure the requirement
   - On reversing car direction all outstanding requests are lost
4. Total development failure gives an unusable system
   - Multiple unreliabilities and knock-on effects

- How do established engineering branches reduce these failures?
  - Mathematics? Science? Technology? Production control? … ?
  - Yes—all of these … and, crucially, learning from experience …
    - Specialisation and normal, not radical design

Normal design

Radical design

"In radical design, how the device should be arranged or even how it works is largely unknown. The designer has never seen such a device before … has no presumption of success: the problem is to design something that will function well enough to warrant further development."

W G Vincenti; What Engineers Know and How They Know It

- Karl Benz 1886 Patent MotorWagen

- 3 wheels
- Solid tyres
- Open cab
- Rear-wheel brakes
- No front springs
- Rear cart springs
- Belt drive
- No gearbox
- Driver in centre
- Steered by tiller

From radical to normal design

- Specialisation allows normal design to evolve
- Radical design, then successive normal designs …

Karl Benz 1886

Ford Model A 1930

Toyota Camry 1992

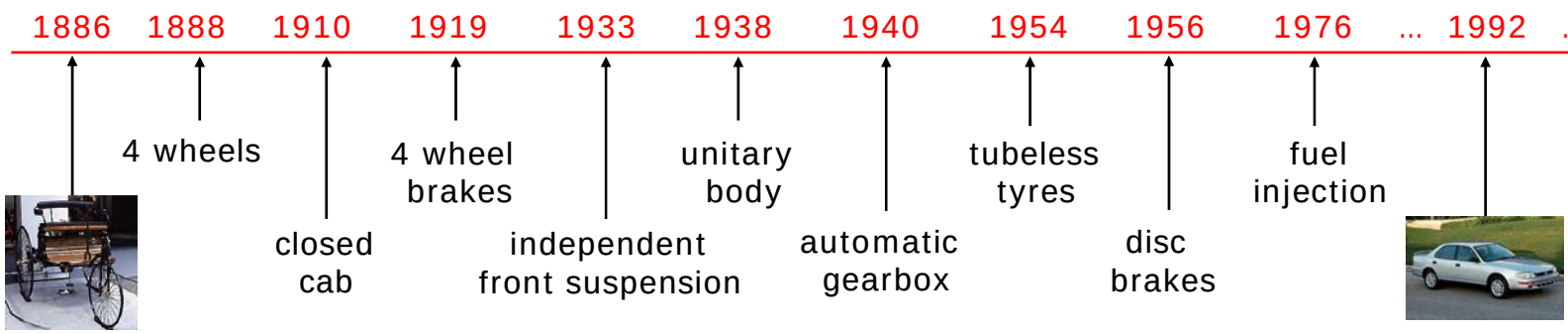
"… in normal design … the engineer … knows at the outset how the device in question works, what are its customary features, and that, if properly designed along such lines, it has a good likelihood of accomplishing the desired task.

W G Vincenti; What Engineers Know and How They Know It

- Normal design and design practice evolve by learning
- Normal design and design practice support learning

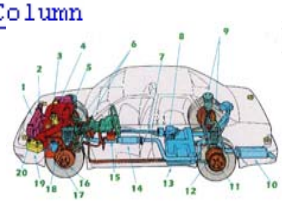
Component structure in normal design

- Evolution of standard component structure
  - Fixes learned functionality and efficiency improvements
  - Provides repository for failure avoidance lessons



- A symptom: named component types in a standard structure

Battery
Brake Drum
Brake Pads
Catalytic Converter
Cross-Flow Radiator
Disc Brake Caliper
Distributor
Engine
Fuel Lines
Fuel Tank
Fuel Tank Sending Unit
Gear Shift Selector
Intake Manifold
MacPherson Strut Suspension
Muffler
Rack&Pinion Steering and Column
Radiator Hose
Radiator Pressure Cap
Rotor
Wheel Mounting Studs



Normal design: learning to avoid failures

- "Engineering advances by proactive and reactive failure analysis, and at the heart of the engineering method is an understanding of failure in all its real and imagined manifestations."

  Henry Petroski; Design Paradigms

de Havilland DH106 Comet 1 (metal fatigue)

Tacoma Narrows Bridge (wind-induced roadway oscillation)

- Avoiding failure by evolving normal design
  - Radical Comet 1: turbojet, 500mph, pressurised
    - Metal fatigue cracks started at window corners
- Avoiding failure by evolving normal design practice
  - Radical Tacoma Narrows: span/width ratio of 1/72
    - Analyse vertical roadway oscillation

Attaching lessons to artifacts

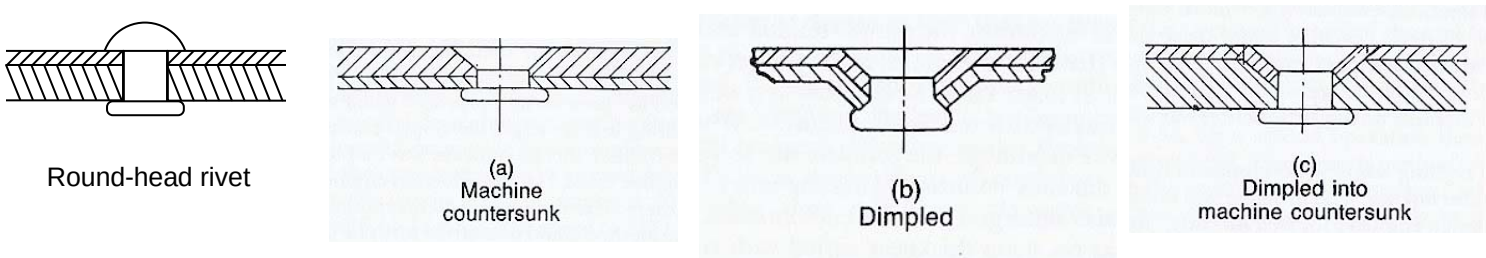
- Tacoma Narrows lesson
  - Suspension bridge … roadway … stiffness … span/width ratio … wind-induced … vertical oscillation
- Comet 1 lesson
  - Turbojet aeroplane … fuselage … torsion stress … pressure stress … metal fatigue … aperture corners

- Specialisation attaches lessons to artifacts and components
  - Normal design has standard components and … … clearly recognised design tasks for them
- Knowledge of general principles is very valuable but …
  - … harder to call to mind when it's important and …
  - … harder to apply for specific artifacts
- Artifact-specific knowledge
  - Particularises a general principle to specific cases
  - Ensures the principle will be applied to those cases

Intensive component specialisation

- Advances in wooden propellers
  - W F Durand & E P Lesley: analytical studies & wind tunnel tests, 150 propeller designs, 1915-1926

- Avoiding failures in flush riveting of metal skin
  - Douglas, Curtiss-Wright, Bell: research & practical experiments in production engineering, 1930-1950



W G Vincenti; What Engineers Know and How They Know It

- Small-scale artifact specialisation
  - Intensive specialisation in research and development
  - Focused on artifact/component dimension

Component specialisation cultures

- It takes more than a village …
  - A specialisation culture
    - Companies, personal careers, conferences, journals, engineering education, research

- Godden Structural Engineering Slide Library
  - Beam structures, Arch structures, Cable and suspension structures, truss structures, Domes and shells, Columns, frames, grids, slabs, Construction

- Journal of Automobile Engineering Vol 221, No 7 / 2007
  - Jeonghoon Song, Heungseob Kim and Kwangsuck Boo; A study on an anti-lock braking system controller and rear-wheel controller to enhance vehicle lateral stability
  - P Hosseini-Tehrani and S Pirmohammad; Collapse study of thin-walled polygonal section columns subjected to oblique loads

Specialisation

Established engineering specialisations

- automobile engineering
- aeronautical engineering
- chemical engineering
- civil engineering
- control engineering
- electrical power engineering
- electronic engineering
- mechanical engineering
- mining engineering
- nuclear engineering
- petroleum engineering
- railway engineering
- structural engineering
- ...

- Multiple dimensions of specialisation
  - theory: control, structural, fluid dynamics, electronics, ...
  - technology: µ-electronics, welding, pre-stressed concrete ...
  - problem world: civil, mining, human, ...
  - requirement: industrial, transportation, ...

- artifact: cars, power stations, aeroplanes, ...
  - component: wings, fuselages, engines, undercarriages ...
  - component: IC engines, electric motors, disk drives, ...

Software engineering specialisations — 1

- Capers Jones's list
  - Chiefly focused on the software development process
    - Architecture
    - Cost estimating
    - Customer support
    - Human factors
    - Integration
    - Measurement
    - Network
    - Performance
    - Planning
    - Requirements
    - Reusability
    - Standards
    - Testing
    - Configuration control
    - Database administration
    - Education and training
    - Function point counting
    - Information systems
    - Maintenance and enhancement
    - Package acquisition
    - Process improvement
    - Quality assurance
    - Systems software support
    - Technical writing
    - Tool development

Capers Jones;Software Specialization; Computer July 1995

Software engineering specialisations — 2

- Generic:
  - Technologies: eg FP, OO, AOP, …
  - Theory: eg concurrency, complexity, types, …
  - Languages: eg Ada, Java, php, …

- Artifacts: chiefly ‘system’ not ‘application’ software
  - Infrastructure: eg internet, LAN, relational DBMS, …
  - Tools: eg compilers, sat solvers, IDEs, model checkers, …
  - Universal components: eg GUIs, file systems, …

- Artifacts: ‘application’ software
  - Systems: eg Lift control, automotive, …
  - Components: eg ABS, CC validation, ATM, …
  - COTS: eg spreadsheet, WP, OS, …

- We need more specialisation in ‘application artifacts’

Lessons for software-intensive systems

Learning from Therac-25

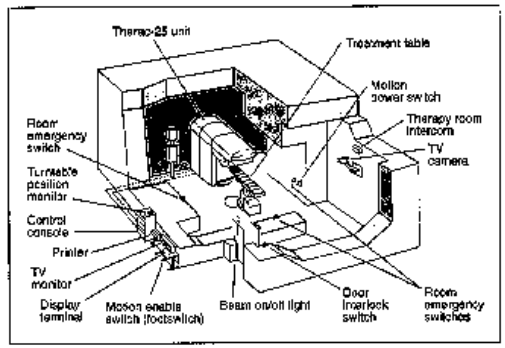


Figure 1. Typical Therac-25 facility.

- Overconfidence in software
- Failure to eliminate root causes
- Unrealistic risk assessments
- Lack of audit trails
- Safe versus friendly user interfaces
- Careless software reuse
- Confusing reliability (low failure rate) with safety
- Lack of defensive design (eg software checks)
- Complacency about radiation therapy machines
- Inadequate investigation or follow-up on accident reports
- Specification and documentation after development
- Complex designs, dangerous coding practices
- Inadequate module and regression testing
- Poor information display, especially of errors
- User and government oversight and standards

Nancy Leveson; Safeware: System Safety and Computers

Lessons for software-intensive systems

Learning from Ariane-5

- Hold software qualification review
- Complete closed-loop system testing
- Include external participants in reviews
- Review and extend test coverage
- Failing sensors should send best-effort data
- Switch offf alignment function immediately after lift-off
- Review assumptions about problem world sensor data
- Pay particular attention to on-board computer switchover
- Confine exceptions to tasks, devise backup capabilities
- More data to telemetry on any component failure
- Definition of ‘critical’ should include software failures
- Include trajectory data in spec’ns and test requirements
- Pay same attention to justification documents as to code
- Set up team to prepare software qualification procedure
- More transparent cooperation among programme partners

http://sunnyday.mit.edu/accidents/Ariane5accidentreport.html

Learning from London Ambulance System

- High risk implementation approach
- Poor consultation with users and clients
- Inadequate ownership of the system
- Poorly designed user interfaces
- Lack of robustness
- Software bugs or errors
- Poor performance
- Software incomplete and effectively untested
- Unjustified wrong assumptions in specification stage
- System didn't fit the organisational structure of LAS
- Reliance on inaccurate Automatic Vehicle Location System
- Changing working practices by new computer system.

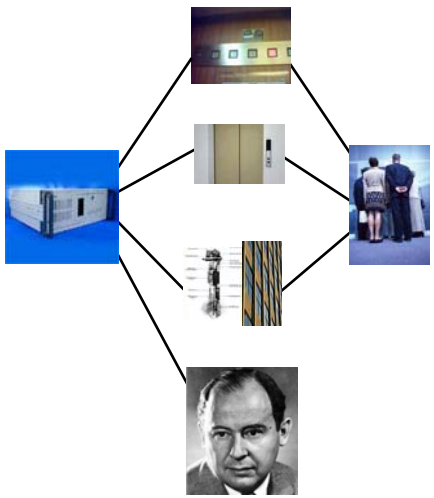
ftp://ftp.cs.city.ac.uk/pub/requirements/lascase0.9.ps.gz

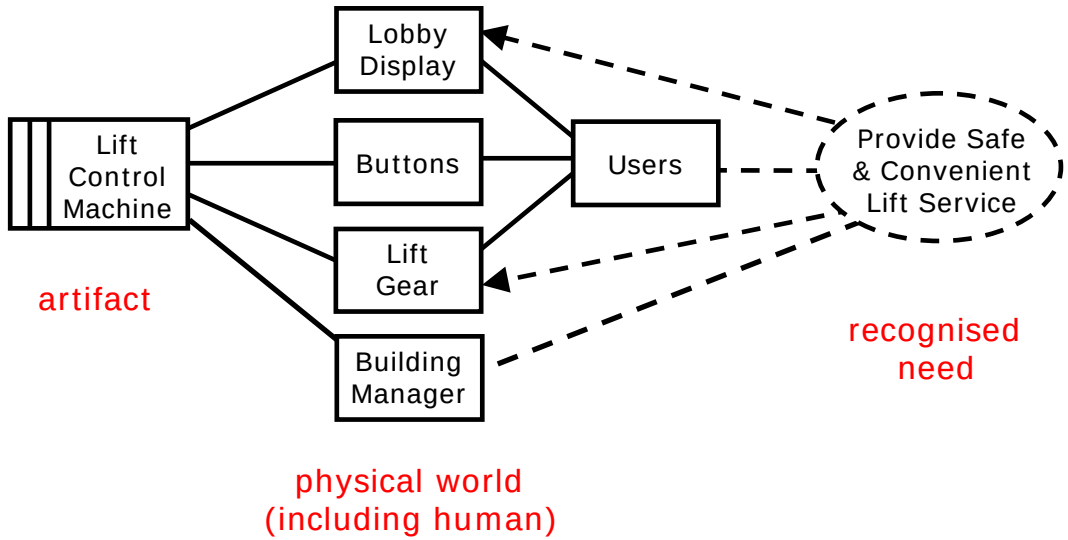
Attaching lessons to components

- The most effective lessons are attached to components
  - What's wrong with which part of the system?
- What are components in software-intensive systems?
  - One requirement = response to a stimulus?
    - "When this button is pressed the motor is turned on"
  - One requirement = use case?
    - "One use of the lift conforms to this scenario …"
  - One requirement = a desired system property?
    - "Idle cars wait at ground floor with doors open"

- One component = subproblem or system function
  - "The system must manage book loans"
  - "The system must provide lift service"
  - "The system must ensure passenger safety"

Problem, solution and requirement

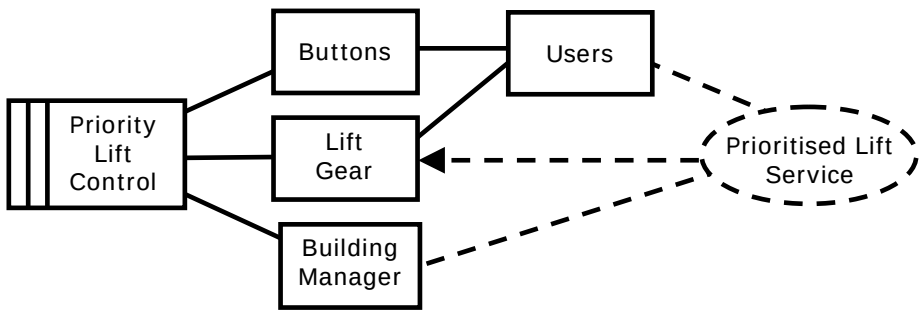


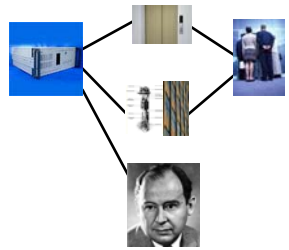


- Distinct parts of the problem world are problem domains
  - Domains and machine all interact by shared phenomena
- Requirement as a satisfaction criterion
  - Machine must constrain Lobby Display and Lift Gear ...
  - ... but not Users, Buttons, or Building Manager

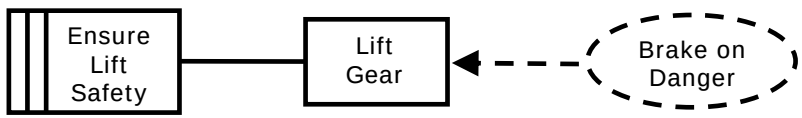
Functional requirements and subproblems

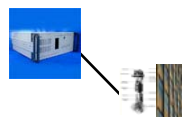
A: Provide lift service as prioritised by the building manager



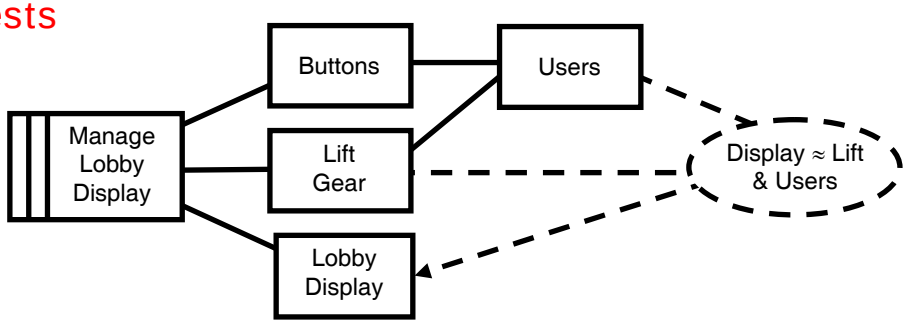


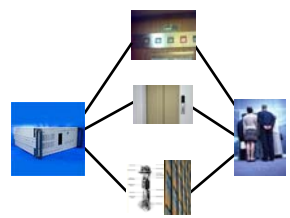
B: Brake on danger when fault found in lift equipment





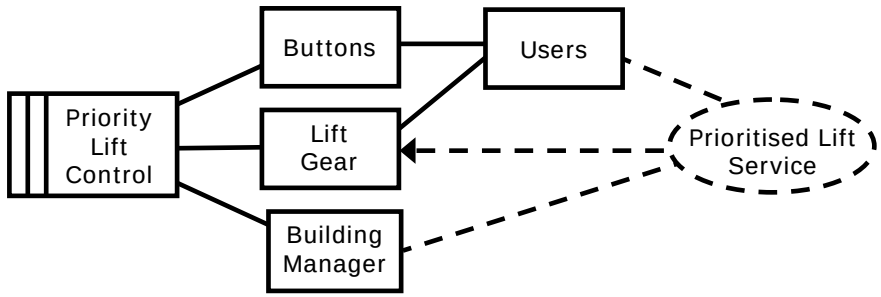
C: Operate lobby display of outstanding requests and lift positions for users' information

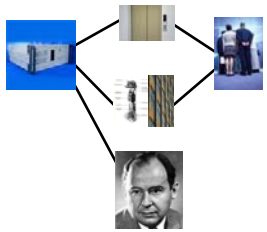




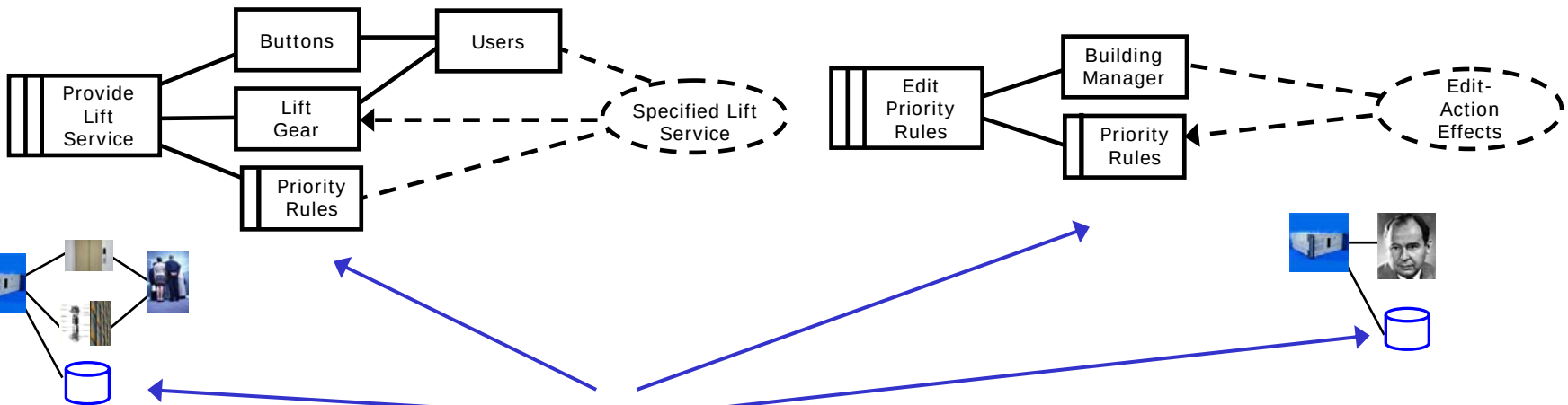
Further decomposition by data structure

A: Provide lift service as prioritised by the building manager





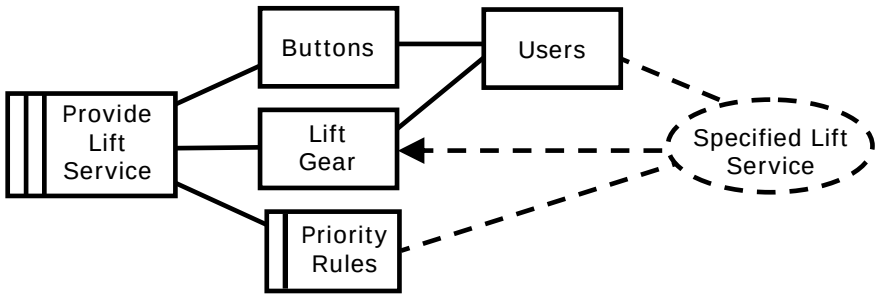
A1: Provide specified lift service + A2: Edit service priority rules



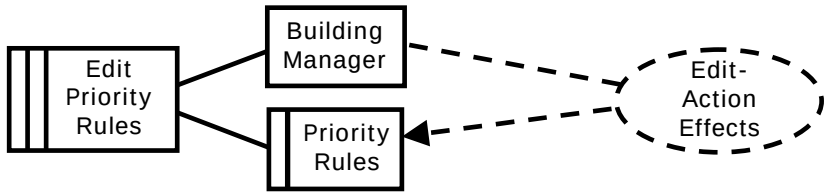
The priority rules data structure domain decouples the Manager from the Lift Service Provision

Six subproblems = six components

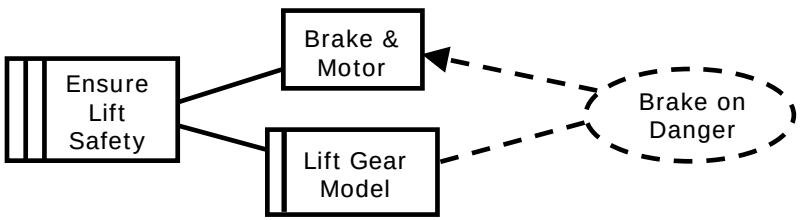
A1: Provide prioritised lift service



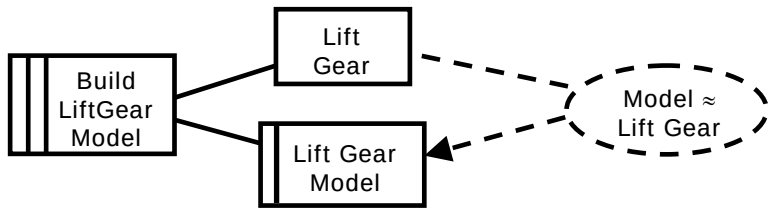
A2: Edit service priority rules



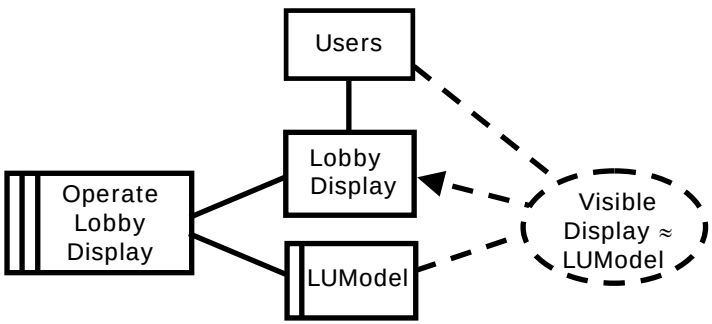
B1: Brake on danger (fault detected)



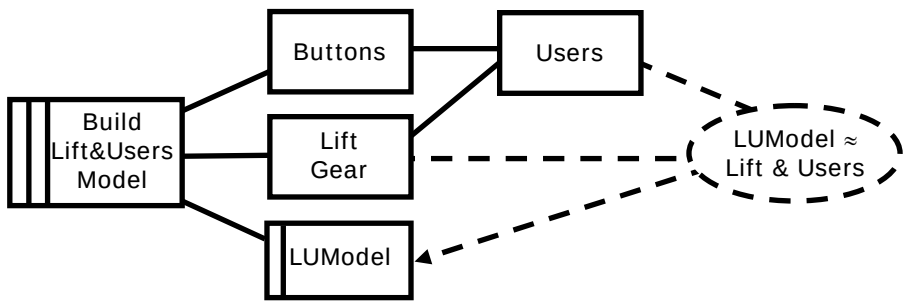
B2: Build lift gear model to detect faults



C1: Operate lobby display

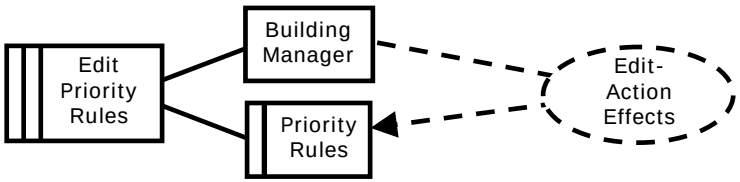


C2: Build model for running display

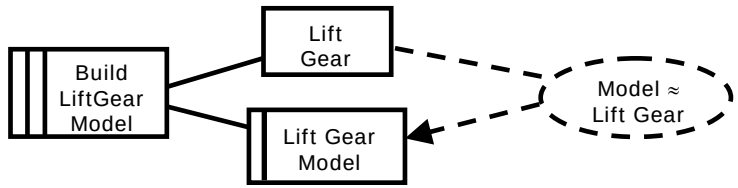


Specialised subproblem classes

A2: Edit service priority rules



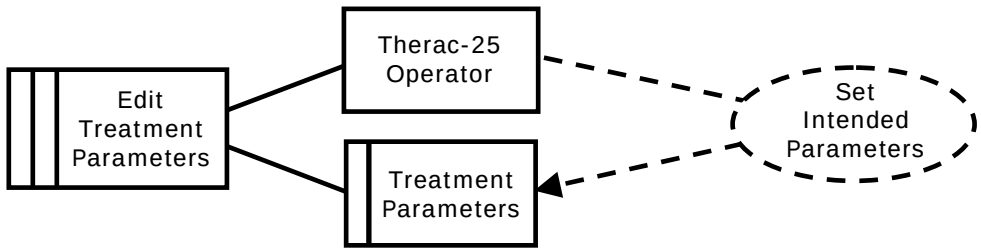
B2: Build lift gear model to detect faults



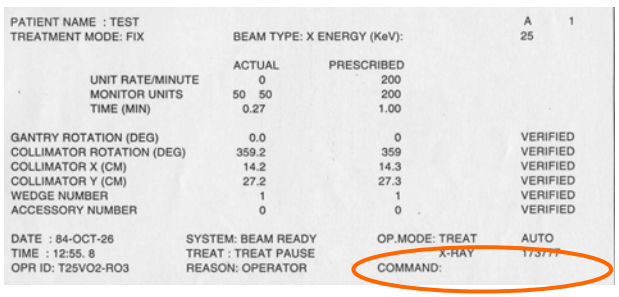
- A2 is a workpieces problem
  - wp problem ⇒ intention, convenience, completeness, …
  - A2 ⇒ semantic constraints, …
  - A2 composition ⇒ schedule switching, …
  - A2 failures …?
- B2 is a dynamic model building problem
  - dmb problem ⇒ initialisation, inference, …
  - B2 ⇒ fault diagnosis, …
  - B2 composition ⇒ mutual exclusion, atomicity, …
  - B2 failures … ?

Therac-25 workpieces subproblem

PT43: Edit patient treatment parameters



- PT43 is a workpieces problem
  - wp problem ⇒ intention, convenience, completeness, …
  - PT43 ⇒ semantic constraints, …
  - PT43 composition ⇒ editing/treatment switchover, …
  - PT43 failures …?



PATIENT NAME : TEST
TREATMENT MODE: FIX
BEAM TYPE: X ENERGY (KeV): A 1 25

	ACTUAL	PRESCRIBED	
UNIT RATE/MINUTE	0	200	
MONITOR UNITS	50 50	200	
TIME (MIN)	0.27	1.00	
GANTRY ROTATION (DEG)	0.0	0	VERIFIED
COLLIMATOR ROTATION (DEG)	359.2	359	VERIFIED
COLLIMATOR X (CM)	14.2	14.3	VERIFIED
COLLIMATOR Y (CM)	27.2	27.3	VERIFIED
WEDGE NUMBER	1	1	VERIFIED
ACCESSORY NUMBER	0	0	VERIFIED

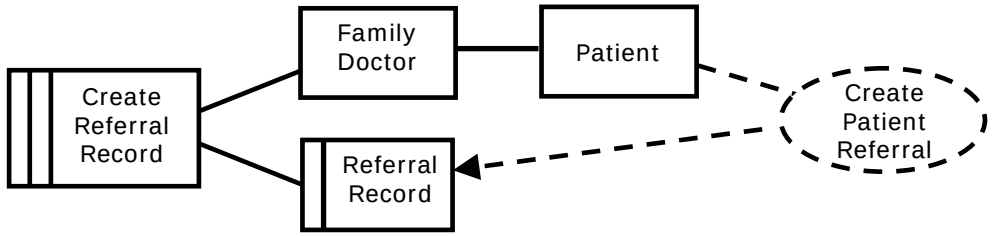
DATE : 84-OCT-26 SYSTEM: BEAM READY OP.MODE: TREAT AUTO
TIME : 12:55. 8 TREAT : TREAT PAUSE X-RAY
OPR ID: T25VO2-RO3 REASON: OPERATOR COMMAND:

"Under the right circumstances the data-entry phase can be exited before all edit changes are made on the screen."

Turner and Leveson; Therac-25

Family doctor referral subproblem

P8329: Edit patient consultancy referral

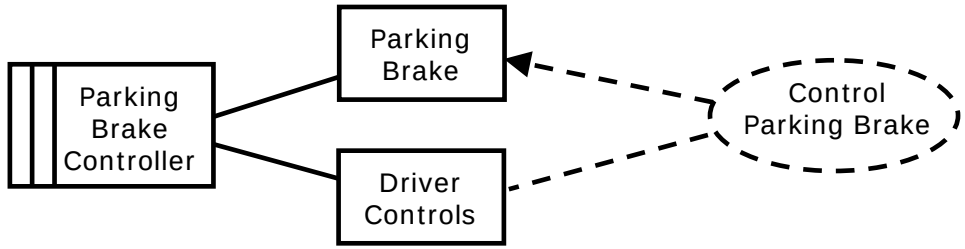


1. Select diagnosed condition on screen
2. View list of relevant consultants
3. Select consultant, make appointment

- Family doctor may be uncertain of diagnosis …
  - … but can refer patient to a known consultant who …
  - … can diagnose exactly and either treat or refer on
  - The system does not allow referral without diagnosis
- Staff response
  - Many referrals bypass the system
  - System data is contaminated or incomplete
  - Knock-on effects from resulting consultant appointment

Improved parking brake control

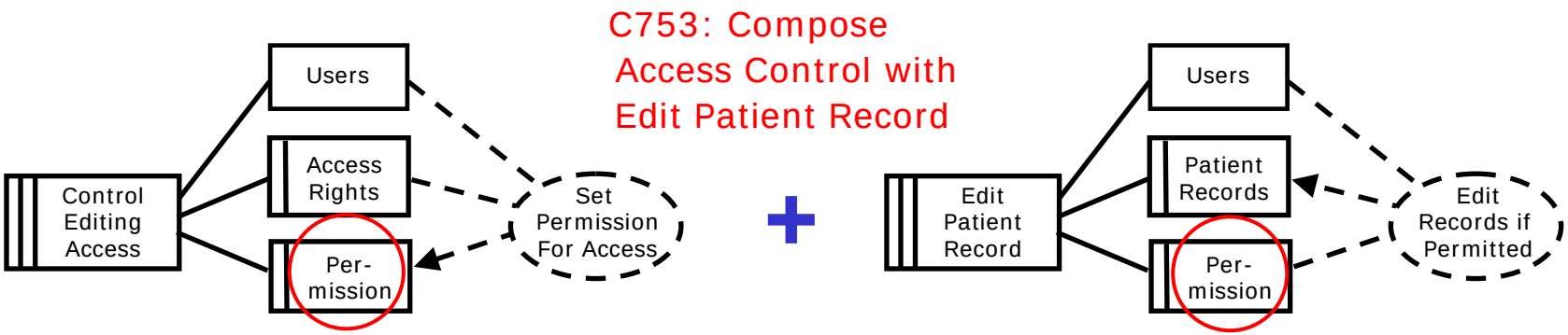
G1927: Control parking brake



- Requirement for new improved parking brake system *
  - Button on fascia to set brake
  - Automatic brake release when accelerator pedal depressed
- Test driver returns to factory
  - Stops car in front of gates
  - Presses brake-on button
  - Leaves car, walks to gates
  - Starts to open factory gates
  - Brake releases, car moves

* Manfred Broy recounted this story

Access rights & workpieces composition



- Patient records are confidential etc
- Staff log in and out to access record
- Full audit trail required: who did what?

- Treatment in A&E is often extremely urgent
- Login takes 15-30 seconds
- Staff response to difficulty
  - N staff log in at start, log out at end, of shift
  - All staff members use PCs with existing log-ins

A success in NFR engineering

- The flying qualities NFR
  - "Stable, responsive, unsurprising, satisfactory to fly"
- The questions: 1918
  1. What do pilots really mean by these NFRs?
  2. What artifact properties can achieve these NFRs?
  3. How can designers achieve those artifact properties?
- (Some of) the answers: 1943
  - Relationship between stability and control
  - Stick-force vs speed, stick-force/g
  - Elevator-angle gradient, short-period oscillation mode, …
  - Applies to normal configuration: lateral symmetry, straight wing, horizontal and vertical tail at rear

Walter G Vincenti; What Engineers Know and How They Know It, Chapter 3

Engineering or requirements? Don't ask!

← Extreme ideal requirements Extreme →

ab initio

- Complete
- Consistent
- Unambiguous
- Measurable
- Stake-holder-friendly
- Implementation-free
- Prerequisite for design
- …

Impossibly hard!
(assuming radical design)

email, photos, WP, …

- Dual Core 2GHz
- XP Pro
- 160GB HDD
- 4GB RAM
- DVD±R-DL
- 24" 1920×1200
- WiFi 802.11n
- …

Much easier!
(relying on normal design)

Engineering or requirements? Don't ask!

← Extreme ideal requirements Extreme →

- Complete
- Consistent
- Unambiguous
- Measurable
- Stake-holder-friendly
- Implementation-free
- Prerequisite for design
- …

Impossibly hard!
(assuming radical design)

- Dual Core 2GHz
- XP Pro
- 160GB HDD
- 4GB RAM
- DVD±R-DL
- 24" 1920×1200
- WiFi 802.11n
- …

Much easier!
(relying on normal design)

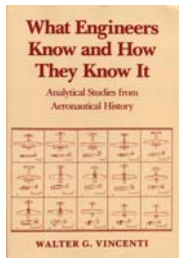
1995

Problems
and
Requirements

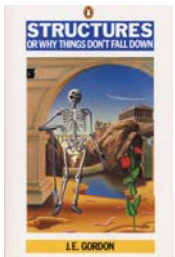
2008

Problems, Solutions
and
Requirements

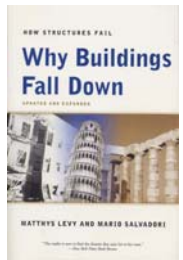
Thank you



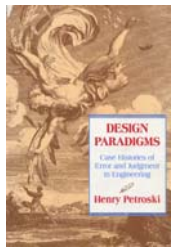
Walter Vincenti



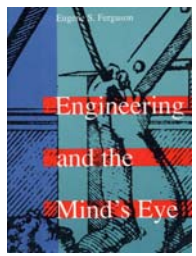
J E Gordon



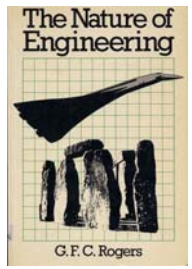
Levy & Salvadori



Henry Petroski

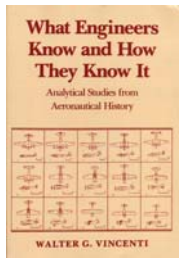


Eugene S Ferguson

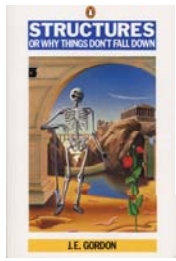


G F C Rogers

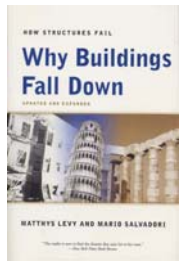
Thank you



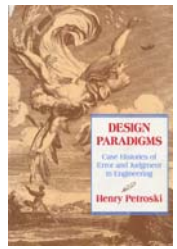
Walter Vincenti



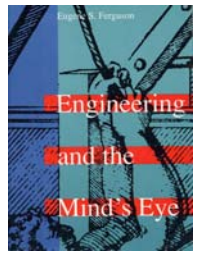
J E Gordon



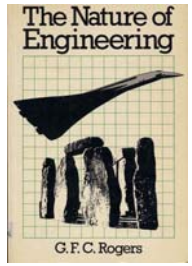
Levy & Salvadori



Henry Petroski



Eugene S Ferguson

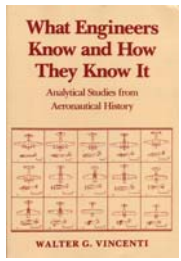


G F C Rogers

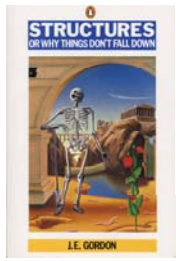
Thank you

David Caminer

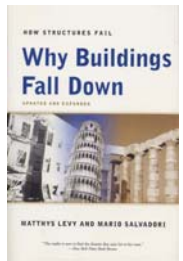
Thank you



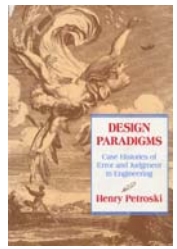
Walter Vincenti



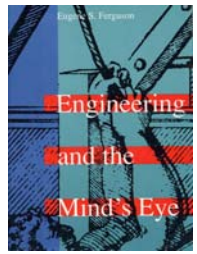
J E Gordon



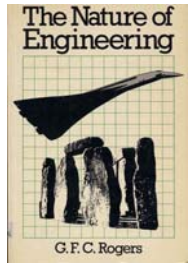
Levy & Salvadori



Henry Petroski



Eugene S Ferguson



G F C Rogers

Thank you

David Caminer

... and thank you all for listening!