Introduction to Requirements Engineering

Before we start …

- Who is this guy in front of me?
  - Christopher Power
  - Canadian (In Yorkshire they say he's "not from 'round here")
  - Lecturer in Human Computer Interaction – I look at how people use technology, especially people with disabilities
  - Has a couple of publications around the requirements area, in particular in traceability

Wait … where am I?

- Requirements Engineering (REQE)
  - Lectures
    - Monday 11:15-13:15 (CSE 266/267)
    - Tuesday 15:15-17:15 (CSE 266/267)
  - Practicals
    - Monday 16:15-18:15 (CSE 266/267)
- As this is your first module you should know:
  - Lectures are optional, but in my opinion a pretty good idea to attend
  - Practicals are mandatory and a register will be taken for each one
- Readings will be assigned for each lecture from the text books
  - You must do these readings if you wish to pass the module
  - You will not be able to pass the exam only by reading the lecture notes

Text Books for this module

- Elizabeth Hull, Ken Jackson and Jeremy Dick (2011) Requirements Engineering, Springer-Verlag.
  - Available online through the library catalogue
- Gerald Kotonya and Ian Sommerville (1998) Requirements Engineering: Processes and techniques
  - 7+ copies available in library, 2 will be put aside for the class on limited loan

Systems Engineering

- Wait … I thought we were talking about requirements engineering???
- Yes … we are … RE is just a part of systems engineering, but a very important part
  - Before any system can be built, you need to understand what it is that is wanted or needed by all of the stakeholders involved

System?

- You probably are thinking right now

  System == Software

- In fact it is a lot more than that – software makes up one part of the systems that we build in this world
  - Complex systems have physical hardware, processes and most importantly people involved in them
  - Many of these people have an interest in the system that is being built – we call these people stakeholders

Stakeholders?

- Direct stakeholders – these are people who will be impacted by the system directly through what it provides
  - Customers & clients
  - Operators & users
- Indirect stakeholders – these people are impacted by the system but it is usually not through the direct use of the system
  - Developers, system architects, regulatory authorities, professional bodies, domain experts, technical experts
- All of these groups will have requirements as to what the system should do or properties the system should have to satisfy their needs and goals

What is a requirement?

- Well let's start with an example:
- You want to take a trip to the seaside – how are you going to get there?
  - Ask yourself (as a stakeholder) – what do I need to know before I choose how I'm going to travel?
- Informally it is all the capabilities and qualities that something must have in order to satisfy the people who will use it.

Formal definitions of requirements

- OED: "*Something called for or demanded, a condition which must be complied with.*"
  - *That's not very useful for building something*
- IEEE-STD-1220: "a statement that identifies a product or process operational, functional, or design characteristic or constraint, which is unambiguous, testable or measurable, and necessary for product or process acceptability (by consumers or internal quality assurance guidelines)"
  - Well that's a lot to take in – can we break that down some?

Breaking down the definition

- Statement
  - We need to be able to write it down somehow
- Product or Process
  - In reality it may apply to both some real product and process(es) related to a product
- Operational, functional, or design characteristic or constraint
  - Basically it is something the product/process has to do (e.g. it must drive 4 people from point A to point B) or it is a quality that it must have (e.g. it must be blue)

Breaking down the definition

- Unambiguous
  - Requirements should be clear as to what they are about for all involved.
- Testable and measurable
  - We need a way to know if a requirement has been satisfied – we must be able either test for it (yes/no) or measure it quantifiably
- Necessary
  - Requirement should specify a capability or quality a product/process must have … Not a "nice to have"
- Acceptability
  - We need to have everyone agree to all of the above

Examples of Requirements

Requirements are things to discover before starting to build your product:

- Things a product must do
  - The product shall produce an amended road de-icing schedule when a change to a truck status means that previously scheduled work cannot be carried out as planned.
- Qualities a product must have *(fast, usable, secure, …)*
  - The product shall use company colours, standard company logos and standard company typefaces.
- Constraints a product must obey *(standards, laws, …)*
  - The product will run on the Department's existing Linux machines.
  - The website must meet to MoD saftey standards

What is Requirements Engineering?

Zave:

"Requirements engineering is the branch of software engineering concerned with the *real-world goals* for, functions of, and constraints on software systems. It is also concerned with the relationship of these factors to *precise specifications* of software behaviour, and to their *evolution over time and across software families*."

Note:

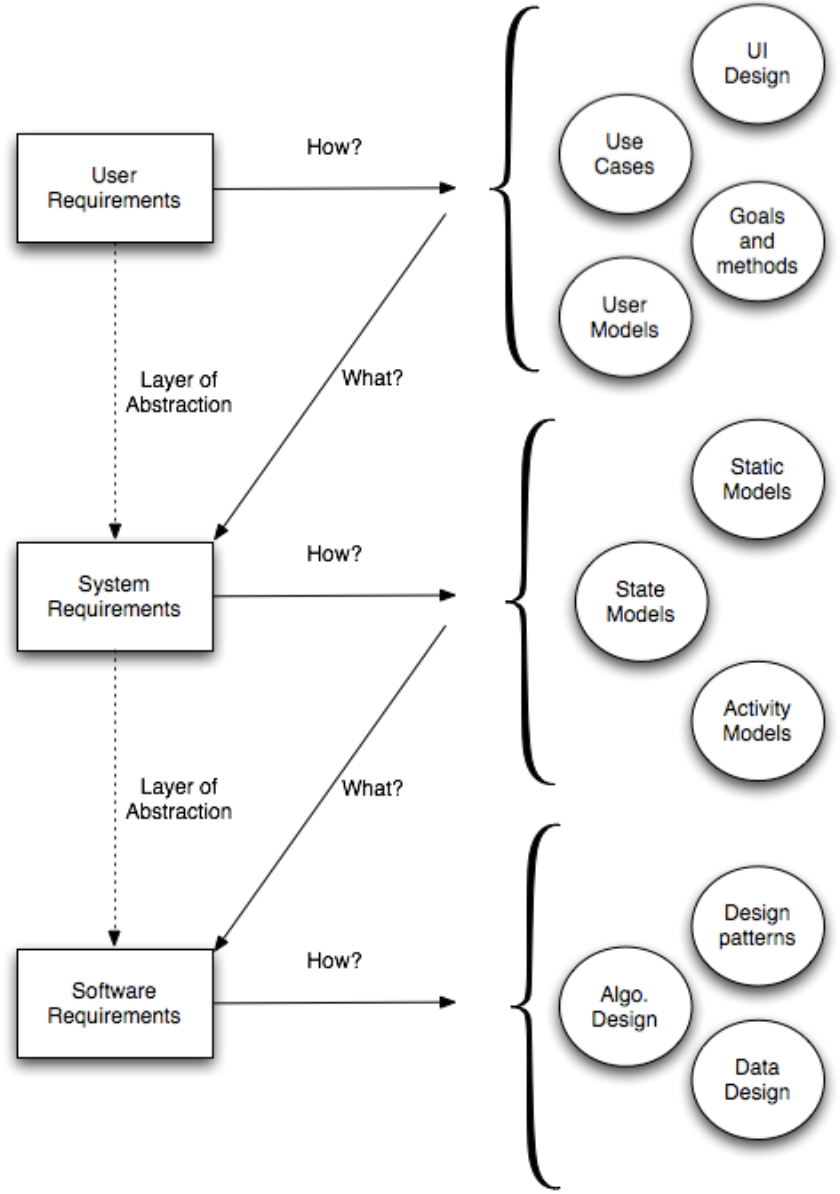
- Real-world goals focus on the `why' as well as the `what'
- Requirements change and are often reused in later projects
- Even though Zave talks about precision, he neglects to mention why we need that precision – we want to be able to measure when we have been successful

Requirements Engineering

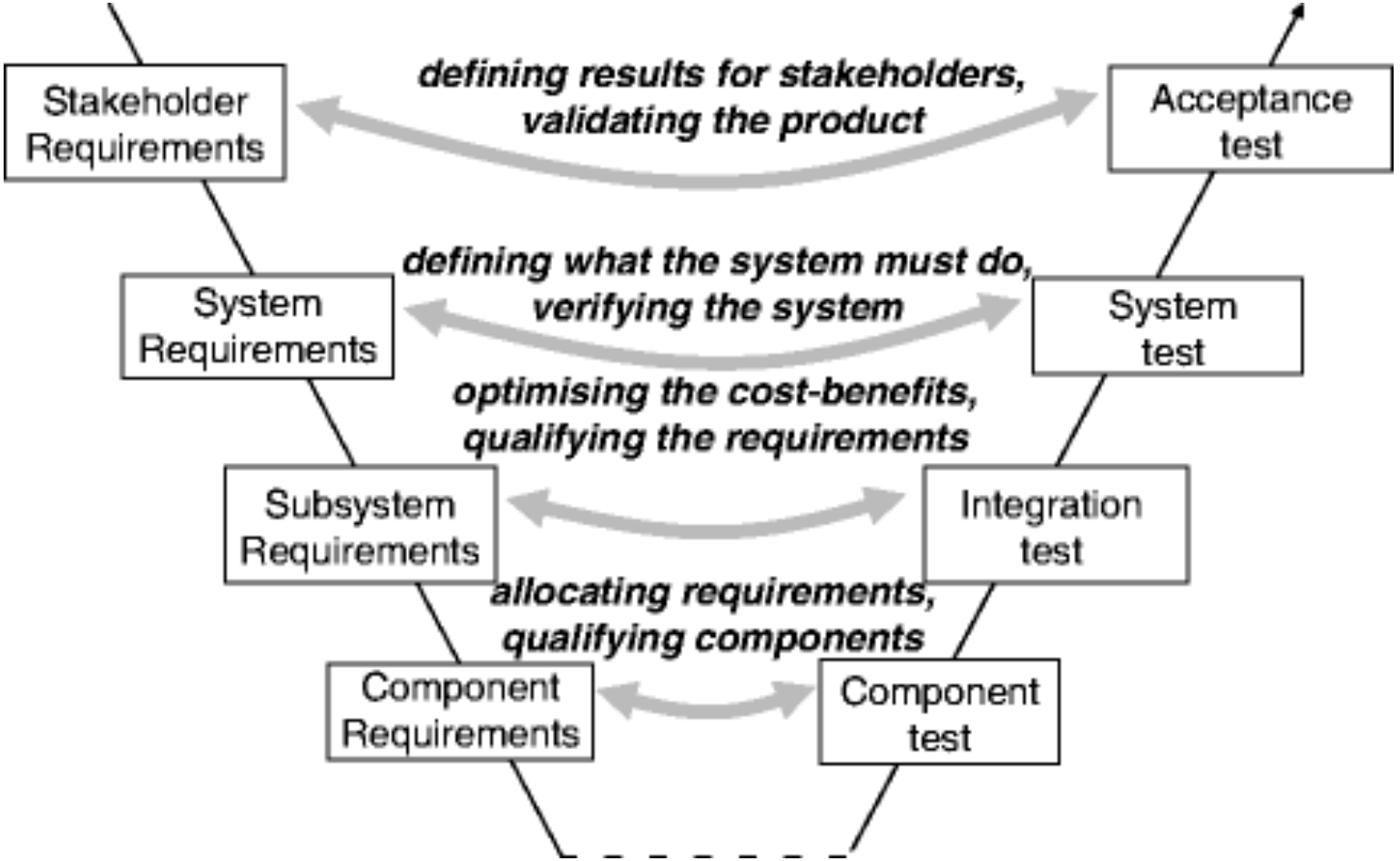
- *Requirements engineering:* the subset of systems engineering concerned with discovering, developing, tracing, analyzing, qualifying, communicating and managing requirements that define the system at successive levels of abstraction

Levels of abstraction

- For large systems, it is hard to specify `what' without specifying `how'.
- The 'how' of one abstraction layer forms the 'what' for the next layer.
- The lower you go in the layers of abstraction, the harder it is to trace back to make sure you are delivering what you originally wanted to deliver.



Requirements in the Engineering Lifecycle



Questions in RE

- How do we …
  - Discover new requirements
  - Analyse and Model different levels of abstraction to provide views of the system
  - Trace requirements from their inception through to their fulfillment
  - Understand if we have reached met a level of quality/fitness for a requirement
  - Document requirements in a way that is useful
  - Negotiate requirements between all of the different stakeholders

Do You Remember the Mars Polar Lander?

Mission:

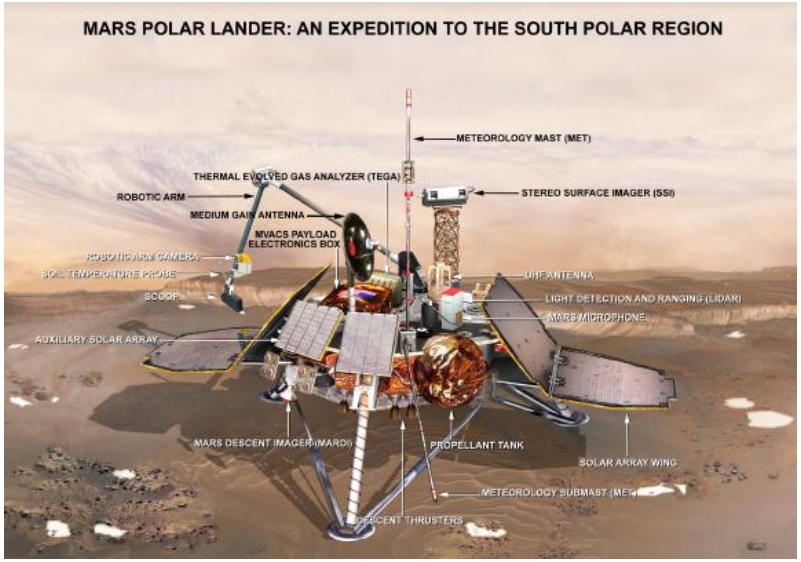
- Land near South Pole
- Dig for water ice

Launched:

- 3rd January 1999

Arrived & lost at Mars:

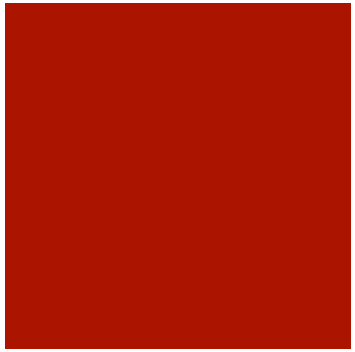
What happened?



[JPL 1999]

- 3rd December 1999
- Investigation hampered by lack of data, since the spacecraft was not designed to transmit telemetry during descent
- Most likely cause is a premature engine shutdown due to noise on leg sensors

Premature Engine Shutdown Scenario



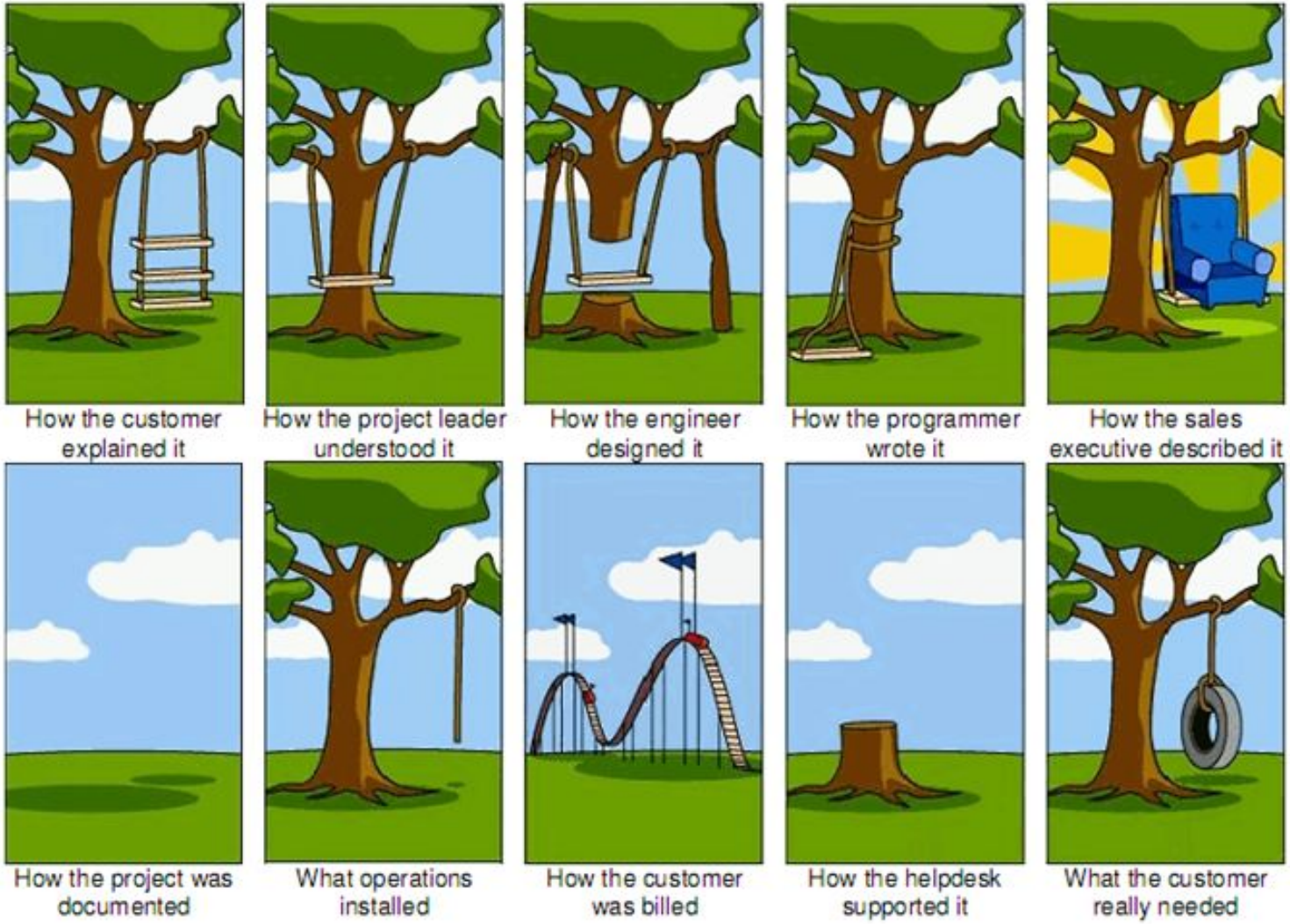
[Easterbrook 2002]

- Cause of error:
  - Magnetic sensor on each leg senses touchdown
  - Legs unfold at 1500m above surface
    - Transient signals on touchdown sensors during unfolding
    - Software accepts touchdown signals if they persist for 2 timeframes
    - Transient signals likely to be this long on at least one leg

Premature Engine Shutdown Scenario

- Factors leading to the error:
  - System requirements ignore the transient signals
  - Engineers at code inspection did not understand the effect
  - Testing did not reveal error
    - Unit tests, based on software requirements, did not include transients
    - Integration test was conducted with improperly wired sensors
- Result of error: Engines shut down before spacecraft has landed
  - When engine shutdown software is enabled, flags indicate that a touchdown has already occurred
  - Estimated impact velocity 22m/s; normal touchdown velocity 2.4m/s

When requirements go bad …



(Modern homage to a pre-1970 cartoon; origin unknown)

Readings

- Hull, Jackson and Dick (2011) – Chapter 1