Week 10 lecture –
1) System and Human Errors
2) Recap and revision

11 December 2024

Dr Mo El-Haj

Lancaster University

System failures and errors: Human Failure

Lancaster University

Understanding (and designing for) human error

What are human errors?

- Can be difficult to distinguish between safe and erroneous behaviour
- For example, are the following actions erroneous?

➢ Deliberately not following a rule
  - ➢ Ignoring a "do not enter" sign during an emergency to assist others.

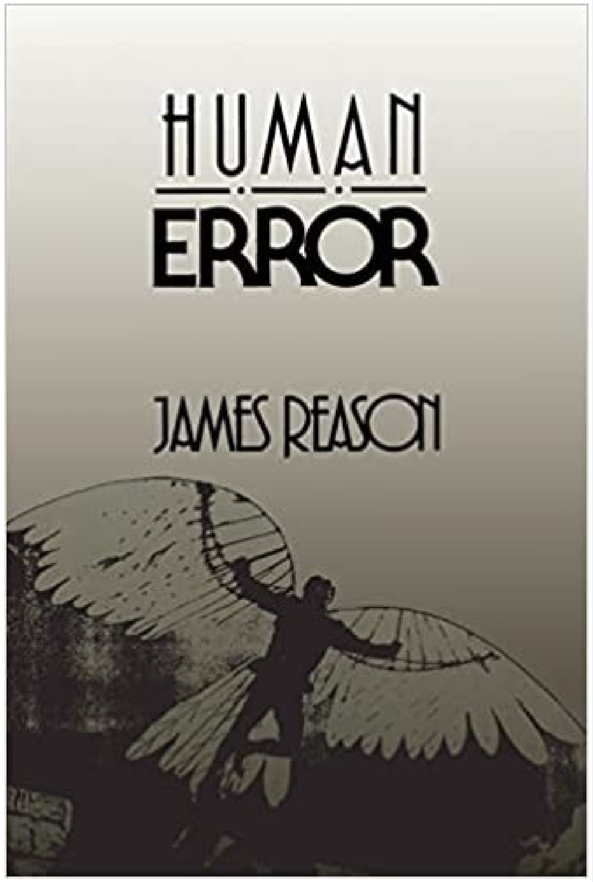
➢ Following a rule
  - ➢ Stopping at a red light in an ambulance, delaying emergency aid.

➢ Taking a short-cut
  - ➢ Skipping equipment checks to save time, risking accidents.

Generic Error Modeling System (James Reason)

•**Failure to perform a task or plan properly**
Example: Forgetting to close a database connection after a query, leading to resource leakage.

•**Failure to apply the correct plan**
Example: Using a sorting algorithm with high complexity for a large dataset, causing performance issues.



Generic Error Modeling System – types of action

Categories of human performance that *can* lead to errors under certain conditions, as defined by James Reason in the GEMS.

- **Skills-based performance**
- Routine, automatic actions requiring little cognitive effort.

Example: Typing on a keyboard without looking.

- **Rules-based performance**
- Following established rules or procedures.

Example: Running a script to back up a database.

- **Knowledge-based performance**
- Using knowledge to address unfamiliar situations.

Example: Debugging a novel software issue without prior guidance.

Generic Error Modeling System – types of error

Errors can be classified based on the type of performance they're associated with:

1. Slips (related to skills-based performance)

- "Execution failure"; the user's intentions are correct, but the actions are not carried out properly.

Example: Clicking the wrong button in a user interface despite knowing the correct one.

2. Lapses (related to skills-based performance)

- Another form of "execution failure," often involving forgetting to perform an action.

Example: *Forgetting to save a file after editing.*

3. Mistakes (related to rule- and knowledge-based performance)

- "Planning failure"; an inappropriate set of actions is executed due to incorrect reasoning or rule selection.

Example: *Using the wrong configuration settings when deploying a server.*

GEMS – Generic Error Modelling System

Advantages of the Model

• Provides a structured framework for designing systems that minimise, detect, correct, and tolerate human error.

Limitations

• Primarily focuses on non-deliberate errors and overlooks deliberate actions (e.g., taking shortcuts).

• High-level approach that neglects the importance of contextual factors.

Why study human error?

- Human error has many negative consequences
- Common reaction – blaming the user

However, systems engineers should be asking:

1. Designing systems that minimise the potential for human error.
2. Creating systems that detect and correct human error.
3. Developing systems that tolerate human error and mitigate its impact.

Addressing human error

Challenges

- Humans are inherently fallible, and errors are unavoidable.
- Human behaviour varies significantly (e.g., skills-based, rules-based, knowledge-based).
- A general approach emphasises **error-tolerance** over error-avoidance.

Key Insight

"It is now widely held among human reliability specialists that the most productive strategy for dealing with active errors is to focus upon controlling their consequences rather than upon striving for their elimination."— *James Reason*

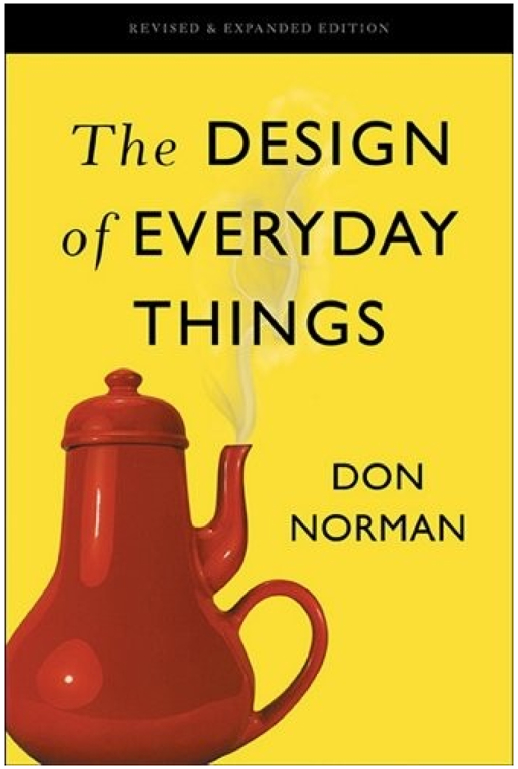
Planning for error

- **Increase System Visibility**
- Avoid hiding complexity; make system behaviour transparent.

- **Include Errors in Training**
- Train operators using error scenarios.

- **Design User-Centred Interfaces**
- Align interfaces with human behaviour.

- **Design for Errors (Norman)**
- Assume errors will occur; make actions reversible and recovery easy.



Is automation the answer?

- Not always!

1. **Limited Scope**

   Automation handles skills- and rules-based tasks but leaves complex, knowledge-based tasks to humans.

2. **Decreased Understanding**

   Automation can reduce system visibility, making it harder for users to understand and troubleshoot.

3. **Shifted Error Sources**

   Errors move from user/operator mistakes to design flaws, which are often harder to detect and fix.

System errors and failures – key points

- Many failures result from human errors. It is important to design in such a way to minimize human error

Suggestions for additional reading

- For theories of Human Error and Resilience, look for work by “Erik Hollnagel”

Term 1 Recap (SCC.141)

Today’s lecture

- Reflecting on what has been learnt over the module so far
- Drawing out the key crosscutting themes from weeks 1-9
- What to expect from the exam (even though it’s a long way off!)

What was this term all about?

- Largely, we've focused on understanding best practices (sometimes through identifying bad practices!)
- Next term, we'll start going a bit deeper and unpacking this a bit, considering some of the complexities that currently exist when we think about computing in society

Reflection on the module so far

Discussion exercise

Talk to the person next to you for 5 minutes, responding to the following questions:

- What did you think a computer scientist was before you started this module?
- Has your understanding of this now changed?
- If so, in what ways?

Lancaster University

Cross-cutting themes

Discussion exercise

- Talk for 5 minutes to the person next to you (a different person if possible!)
- What themes can you identify that cut across multiple lectures?

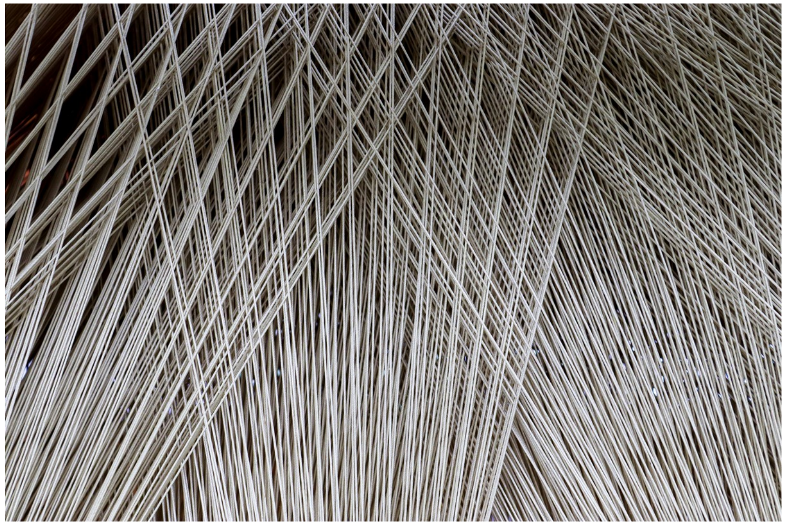
A few cross-cutting themes

1) Being a computer scientist is not just about programming

2) A computer scientist has legal and ethical obligations

3) Importance of managing risk

4) Importance of considering end-users



1: Being a computer scientist is not just about programming

- "Computer science (CS) is the study of computers and algorithmic processes, including their principles, their hardware and software designs, **their applications, and their impact on society**."
- Involves a whole range of other skills – requirements engineering, design, ethical thinking, user research etc.
- Even if you don't end up using all of these yourself, you'll likely be working with people who do and need to be able to communicate with them
- **Problem solving** is crucial, and this doesn't just mean solving technical problems

Tucker, A., 2003. A model curriculum for k--12 computer science: Final report of the ACM K--12 task force curriculum committee. ACM.

1: Being a computer scientist is not just about programming

Types of problem solving:

- Seeing the bigger picture
- Identifying potential risks and unintended consequences
- Understanding a current problem
- Identifying requirements
- Designing a solution that meets requirements
- Negotiating trade-offs

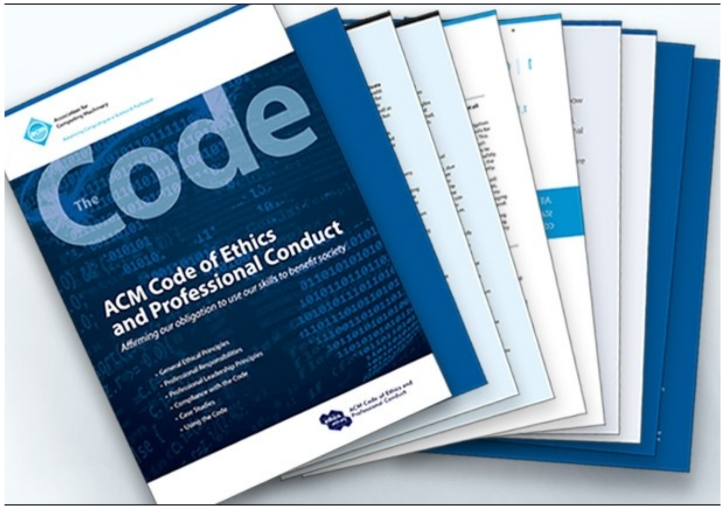
2: A computer scientist has legal and ethical obligations

- Legal requirements govern behaviours
- Ethics - moral principles. Ethics codes provide rules of conduct recognised by a given group, e.g., a professional body
- Best practice – not just sticking to legal requirements as a bare minimum, but trying to do what is ethically right

2: A computer scientist has legal and ethical obligations

Codes of conduct/codes of ethics:

- BCS code of conduct
- ACM code of ethics
- Codes of conduct suggest how members should behave, in relation to their employer and wider society


The Code
ACM Code of Ethics and Professional Conduct
Affirming our obligation to use our skills to benefit society

2: A computer scientist has legal and ethical obligations

BCS Code of Conduct:

- **Professional competence and integrity** - e.g., only completing work that is within your competency, commitment to continued professional development and learning
- **Public interest** – e.g., ‘have due regard for public health, privacy, security and wellbeing of others and the environment’, not discriminating against others, ‘promote equal access to the benefits of IT’
- **Duty to the profession** – e.g., upholding the profession’s reputation

2: A computer scientist has legal and ethical obligations

Legal frameworks:

- GDPR – EU's data privacy law (2018): grants rights to 'data subjects' and dictates how personal data should be stored and used
- Equality Act (2010): failure to design for inclusiveness and accessibility may constitute unfair discrimination
- Copyright, Designs and Patents Act (1988) - software copyright (covers licenses, etc.) (CDPA) – licenses of use, etc.

2: A computer scientist has legal and ethical obligations

Consequences of breaking the law when it comes to personal data storage and usage:

- Major fines! – 4% of turnover
  - ➢ British Airways – £183 million (2019)
  - ➢ Marriott Hotels - £99 million (2019)
  - ➢ Facebook (Cambridge Analytica scandal) - £500,000 (2018)
- Reputational damage
- Harm to end-users

3: Importance of managing risk

- Anticipating and minimizing unintended consequences
- Building dependable systems
- Building error-tolerant systems
- Significance of SDLC and requirements engineering

3: Importance of managing risk

Anticipating and minimising unintended consequences:

- Many possible unintended consequences to be aware of:
  - Potential misuse, malevolent actors
  - Negative effects on wellbeing
  - Inadequate protection of privacy and security
- **Anticipating these risks and thinking about how to minimise them is a key part of what it means to be a problem-solver**

3: Importance of managing risk

Dependability:

- Very important property for most systems
- “Dependability is defined as that property of a computer system such that reliance can justifiably be placed on the service it delivers.” (Mellor)
- There are ways of enhancing dependability (through, for example, removing faults, being fault-tolerant, etc.)

3: Importance of managing risk

Building error-tolerant systems:

- Design systems that minimize potential for human error, detect and correct human error, and tolerate human error
- Norman: design for errors. Assume errors will occur and plan for error recovery. For example, make it easy to reverse actions

"It is now widely held among human reliability specialists that the most productive strategy for dealing with active errors is to focus upon controlling their consequences rather than upon striving for their elimination"
(Reason)

3: Importance of managing risk

Significance of system development lifecycle and requirements engineering:

- Good planning and analysis stages should enable consideration of potential risk
- Incomplete requirements are a key cause of project failure, so identifying the right requirements is crucial

4: Importance of considering end-users

- Vital to consider end-users throughout the system development lifecycle

Why?

- Computing – used by many people (no longer just technical professionals)
- People have diverse needs, desires, capabilities and limitations

- Systems should be **accessible** and have good **usability**

4: Importance of considering end-users

Lancaster University

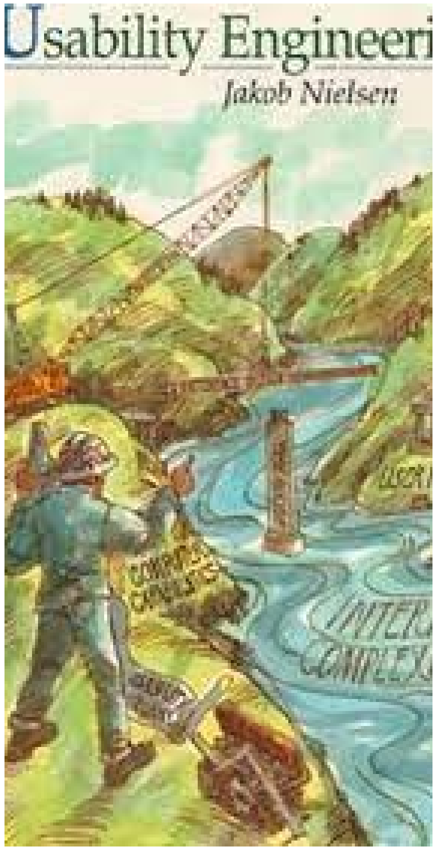
Accessibility:

- "Digital accessibility means designing and building your digital offerings so that, regardless of a person's mental or physical ability, they can still interact with your website, app, or other digital product in a meaningful and equal way." (https://web.dev/learn/accessibility/measure)
- Accessible design is often good for everyone – concept of universal design

4: Importance of considering end-users

Usability:

- Learnability – the system should be easy to learn
- Efficiency – the system should be efficient to use
- Memorability – the system should be easy to remember
- Errors - the system should have a low error rate
- Satisfaction – the system should be satisfying to use



4: Importance of considering end-users

However, there are many obstacles:

- Users may not know what they want, or struggle to articulate it
- Users may not be aware of the technical possibilities and constraints
- Tensions between different stakeholder and user needs

Lancaster University

Answering your questions!

How will seminars work in term 2? Is it still group work? if so, will they be the same groups?

- Still group work, and a similar structure of seminars that contribute to the group assignment
- But the groups will change!
- Why? So that you can experience working with other people, gain from other perspectives, etc.

How will seminars work in term 2? Is it still group work? if so, will they be the same groups?

Lancaster University

- We want to be encouraging you to discuss and interact with each other – teamwork is a vital skill for computing professionals
- The group work you do in 141 will prepare you for further group work in Years 2 and 3
- We think you can learn from each others' perspectives

- However, please note that the exam, the individual assessment component of this module, is worth 70% of your 141 grade

Do we need to remember every case study provided in the lectures?

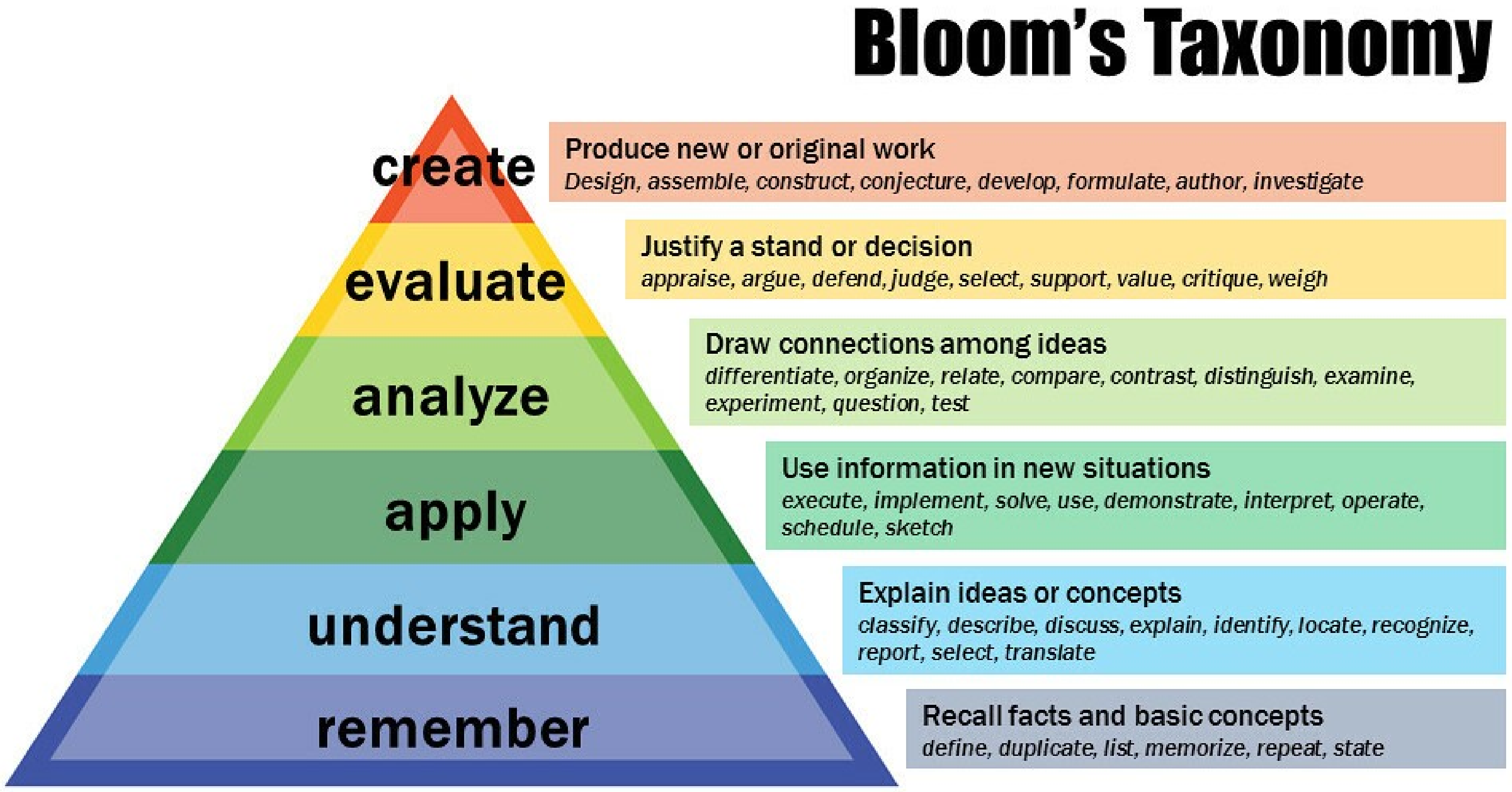
- No – what is important is understanding the ideas and the concepts
- Being able to discuss a relevant example to the question will be useful – this might be case studies from the literature, or examples you have come across in your own independent research

- More to follow on exams next!

Lancaster University

Looking forward (?!) to the exam....

What are we examining?


Bloom's Taxonomy
create
Produce new or original work
Design, assemble, construct, conjecture, develop, formulate, author, investigate
evaluate
Justify a stand or decision
appraise, argue, defend, judge, select, support, value, critique, weigh
analyze
Draw connections among ideas
differentiate, organize, relate, compare, contrast, distinguish, examine, experiment, question, test
apply
Use information in new situations
execute, implement, solve, use, demonstrate, interpret, operate, schedule, sketch
understand
Explain ideas or concepts
classify, describe, discuss, explain, identify, locate, recognize, report, select, translate
remember
Recall facts and basic concepts
define, duplicate, list, memorize, repeat, state

When applied to exams, Bloom's Taxonomy helps educators develop questions and tasks that target different levels of cognitive skills, ensuring a balanced evaluation of students' knowledge and abilities.

Image by Vanderbilt University Center for Teaching - https://www.flickr.com/photos/vandycft/29428436431, CC BY 2.0, https://commons.wikimedia.org/w/index.php?curid=96306577

The exam

- Will not primarily be a memory test, though there may be a small number of factual questions
- We want you to show that you've **understood** the content
- And we'd like to see you making connections between different ideas that have been explored across the lectures

Exam question examples – usability focus

The kind of questions you will <u>not</u> be asked

- List 5 of Nielsen's 10 usability heuristics
- What is the calculation for working out a usability metric?

✗

The kind of questions you could be asked

- Why is usability important?
- What kind of methods are used to evaluate usability?

✓

Lancaster University

Thank you!

Wishing you a happy holidays!