Designing for Dispute Resolution

Steven Murdoch
University College London

Get these slides here

murdoch.is/:/shb21

Dispute resolution is commonly an afterthought for system design and implementation

- Disputes can occur when the system fails, so it's tempting to not specify for this situation
- Redundancy can identify the presence of a problem but does not necessarily make it feasible to resolve the dispute
- State of the art (since 13th century): double entry accounting
  - Each party maintains a balance. Every debit has a corresponding credit such that the system balances to zero at all points in time. Error detected if invariant fails.
  - Used for money, but also any sort of inventory management
  - Often used in systems where trust is shared, and hence disputes could occur
- When an imbalance is detected, what happened and who is to blame?

The Horizon computer system failed to properly resolve disputes

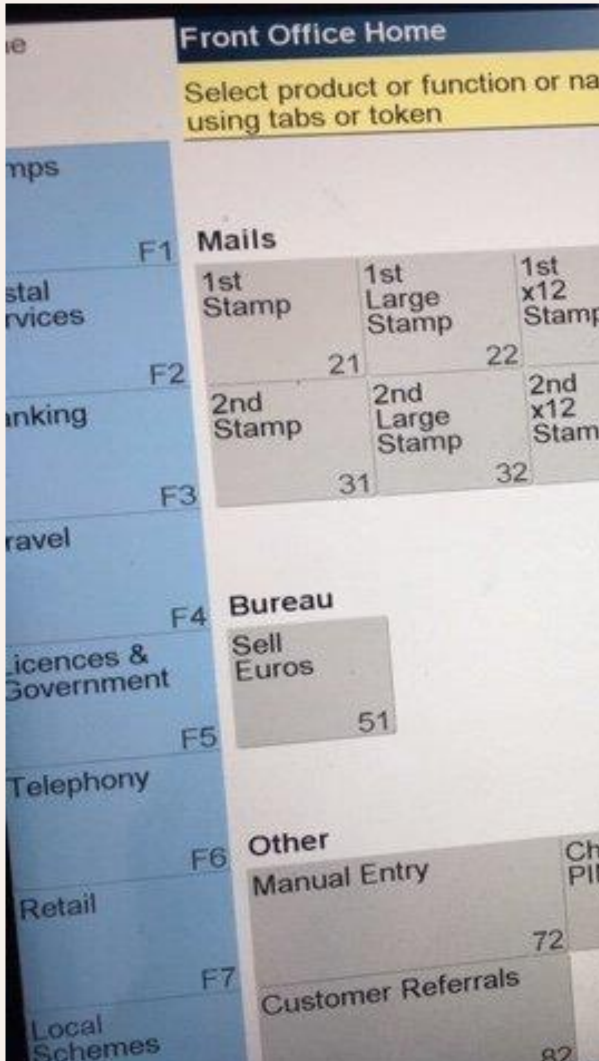
- Horizon is the system created by Fujitsu for the Post Office for managing Post Office branches
- Over 900 subpostmasters prosecuted for fraud and many more had to repay supposed losses identified by Horizon

- For decades the Post Office got away with inadequate disclosure and relied on the presumption that computers are reliable
  - Eventually they lost, and maybe there will be consequences for some individuals in the Post Office and the Fujitsu
- This approach might not be effective in the future
  - Those individuals who might be made liable might wish to change it too

murdoch.is/:/shb21

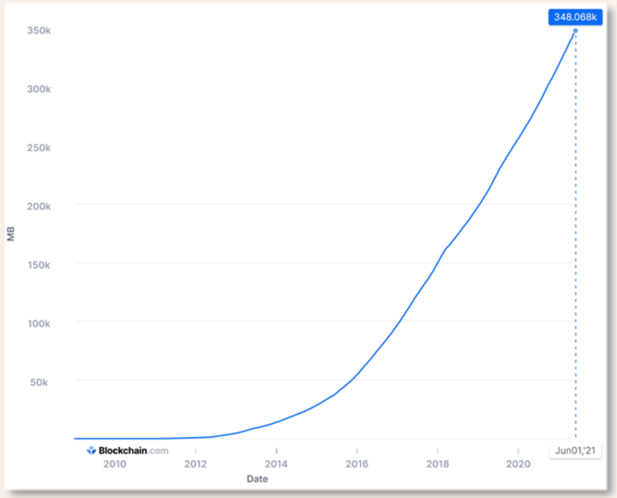
Horizon's problems occurred because disputes could not be effectively resolved

- Sometimes a branch did not balance, maybe due to error or fraud but sometimes due to a bug
  - e.g. a book of stamps is moved, but this results in a duplication
  - Horizon will identify an imbalance but actually tracing down the cause is challenging because we just have balances of accounts
- Imbalance is resolved through a suspense account which eventually becomes profit (maybe over £100 million)
- Fine as long as the company accepts the losses
  - In practice a bug-induced loss could be held the responsibility of the subpostmaster while the credit becomes profit



murdoch.is/:/shb21

Might there be (some) lessons to learn from blockchain based currencies?



- Double-entry accounting was designed to be efficient on paper and was translated directly to computers
- Bitcoin was designed solely to be used by computers
- For implementation reasons, Bitcoins are not fungible: each can be traced from creation through the network
  - Blockchain size is a few hundred GB: a problem on paper but fine for computers
- What if double-entry accounting was replaced by end-to-end traceability of items, not just balances
  - A 64 bit ID for each item of UK post would be about 7GB (smartwatch size)
  - Duplicates, deletions, incorrect movements all easier to diagnose

murdoch.is/:/shb21

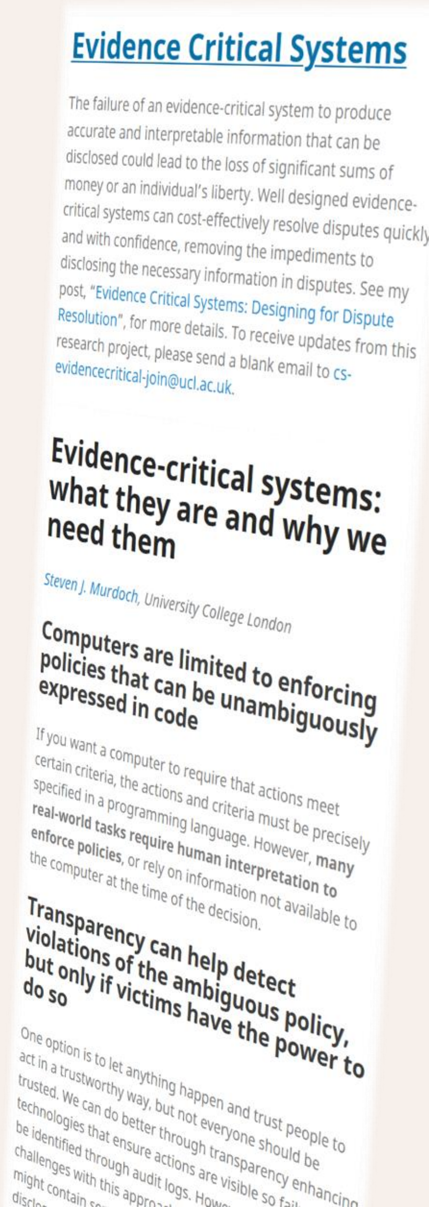
Traceability of items could be a valuable addition to evidence critical systems

- An evidence critical system is one for which its failure to produce reliable and interpretable evidence could cause serious harm
- As computers are increasingly recognised to be fallible the ability to demonstrate proper functioning will become more important

- *Traceability is one design pattern, but what are others?*
- *What resources should people interested in this topic look at?*

More details at evidencecritical.systems (work in progress)

Join the mailing list (instructions on the above page) for updates



Evidence Critical Systems

The failure of an evidence-critical system to produce accurate and interpretable information that can be disclosed could lead to the loss of significant sums of money or an individual's liberty. Well designed evidence-critical systems can cost-effectively resolve disputes quickly and with confidence, removing the impediments to disclosing the necessary information in disputes. See my post, "Evidence Critical Systems: Designing for Dispute Resolution", for more details. To receive updates from this research project, please send a blank email to cs-evidencecritical-join@ucl.ac.uk.

Evidence-critical systems: what they are and why we need them

Steven J. Murdoch, University College London

Computers are limited to enforcing policies that can be unambiguously expressed in code

If you want a computer to require that actions meet certain criteria, the actions and criteria must be precisely specified in a programming language. However, **many real-world tasks require human interpretation to enforce policies**, or rely on information not available to the computer at the time of the decision.

Transparency can help detect violations of the ambiguous policy, but only if victims have the power to do so

One option is to let anything happen and trust people to act in a trustworthy way, but not everyone should be trusted. We can do better through transparency enhancing technologies that ensure actions are visible so fail
be identified through audit logs. How
challenges with this appro
might contain s
disclo

murdoch.is/:/shb21