Rail Accident Investigation Branch

Investigating Software Failures

Rail Accident Investigation Seminar

Richard Brown
Inspector, RAIB

10 November 2021

Safety critical/related or high integrity

- A **safety-critical system** is a system whose failure or malfunction may result in one (or more) of the following outcomes:
  - death or serious injury to people
  - loss or severe damage to equipment/property
  - environmental harm
- **High Integrity** systems are designed and maintained to have a high probability of carrying out their intended function
- A **safety-related system** is a system in which failure would cause a significant increase in the safety risk, but would only be as hazardous as safety-critical in conjunction with the failure of other systems or human error

Barriers to software investigation

Safety related

Safety critical

Barriers to software investigation

- **Nothing physical, system hardware to observe**
- **No software development experience**
- **No means or facilities to test independently**
- **Where would you even start with ‘code’**
- **Unfamiliar vocabulary**

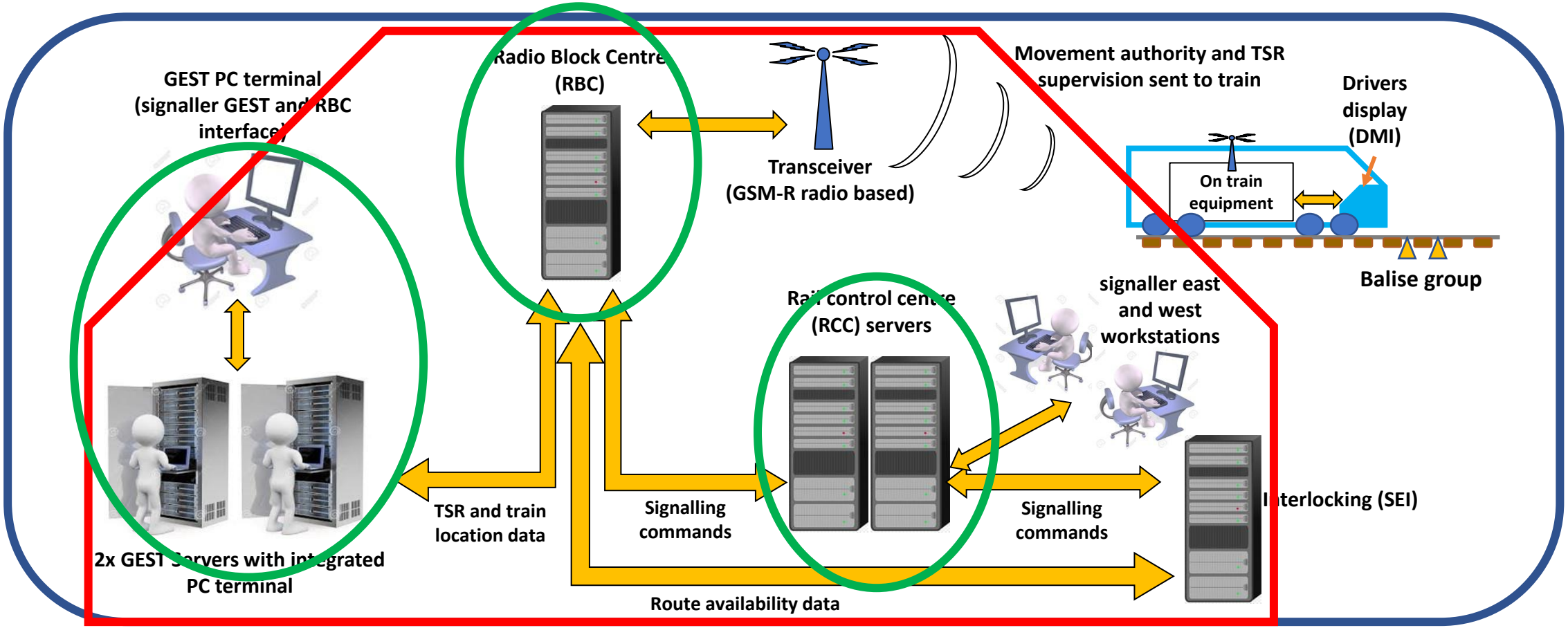
Challenge 1 - Understanding the architecture

- User and maintenance manuals, written for the uninformed, provide information on:
  - What sub-systems exist and what each bit does;
  - How and what data is passed between sub-systems;
  - What information is provided to users and maintenance staff; and
  - What data logs are available.

- Caution:
  - Manuals reflect how this system behaves to the user and not always how the system actually functions, but they are a good starting point.
  - Don’t forget the user, are they a part of the system integrity?

System concept

Socio-technical system = people, processes, organisation + hardware, software, data



ERTMS system = hardware, software, data relating to train control

Sub-systems = part of the bigger system, but which is itself a system

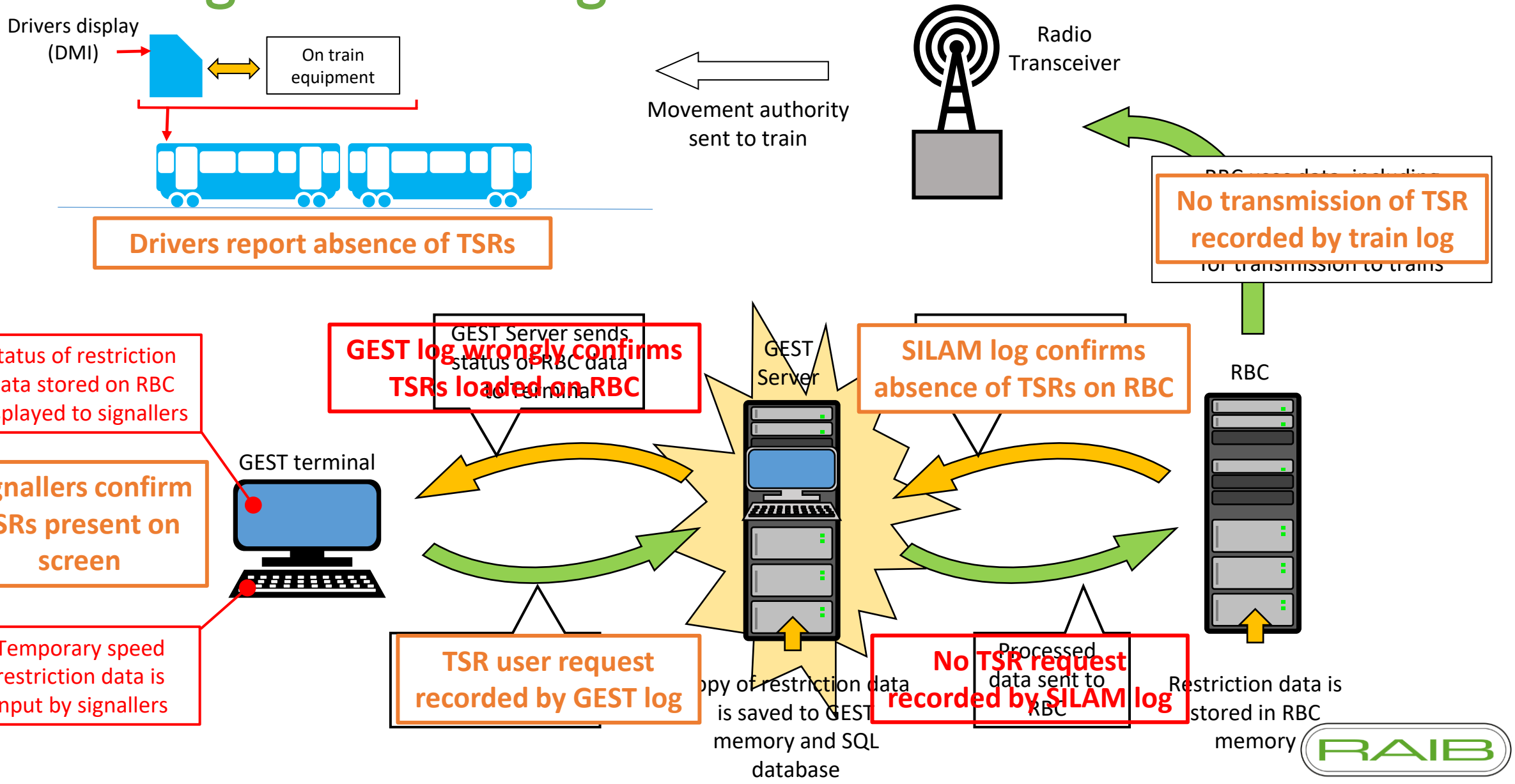
Loss of data on Cambrian Lines

- The signalling system was installed in 2011 and designed in accordance with the ‘European Rail Traffic Management System (ERTMS) level 2’ specification
- Signalling system was a UK pilot installation of ERTMS technology intended to provide operational learning for future installations
- Movement authorities and permitted speeds are displayed to drivers via a cab display

The incident

- Approx 23:04 hrs 19 October 2017, the ERTMS signalling system automatically rebooted.
- All TSRs stored within the signalling system disappeared; critically no indication was given to the signaller or train crew that the TSRs were lost.
- Next morning trains began operating, no TSRs were displayed to drivers and the train control system did not intervene to prevent over speeding.
- A train passed through a 30 km/h (19 mph) TSR at approximately 80 km/h (50 mph)
- The TSR had been applied at this location since 2014 to provide level crossing users with sufficient warning of approaching trains so they could cross safely.
- The driver of this train reported a fault with his DMI, while investigating this report, a signalling technician at the Machynlleth control centre discovered that TSR data was not being transmitted to any of the trains on the Cambrian lines.

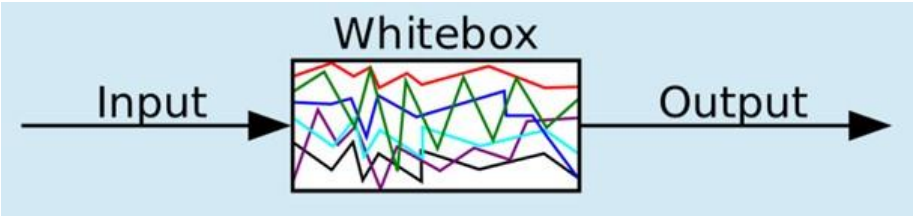
Challenge 2 – Isolating the issue



Challenge 3 – Identifying the failure cause

- All testing would need to be undertaken in a lab and by the supplier (on-site testing and re-creation not possible on the operational railway).
- Black box (functional) testing:
  - Lab based manipulation of inputs and operating environment;
  - Ensure undertaken on correct version;
  - Follow known conditions at time of failure.

- White box (internal software) testing, eg:
  - Lab based invasive freezing of software threads and injection of faults to observe reaction;
  - Must be justifiable by a 'naturally' occurring stimulus (exception handling/intolerable data error etc);

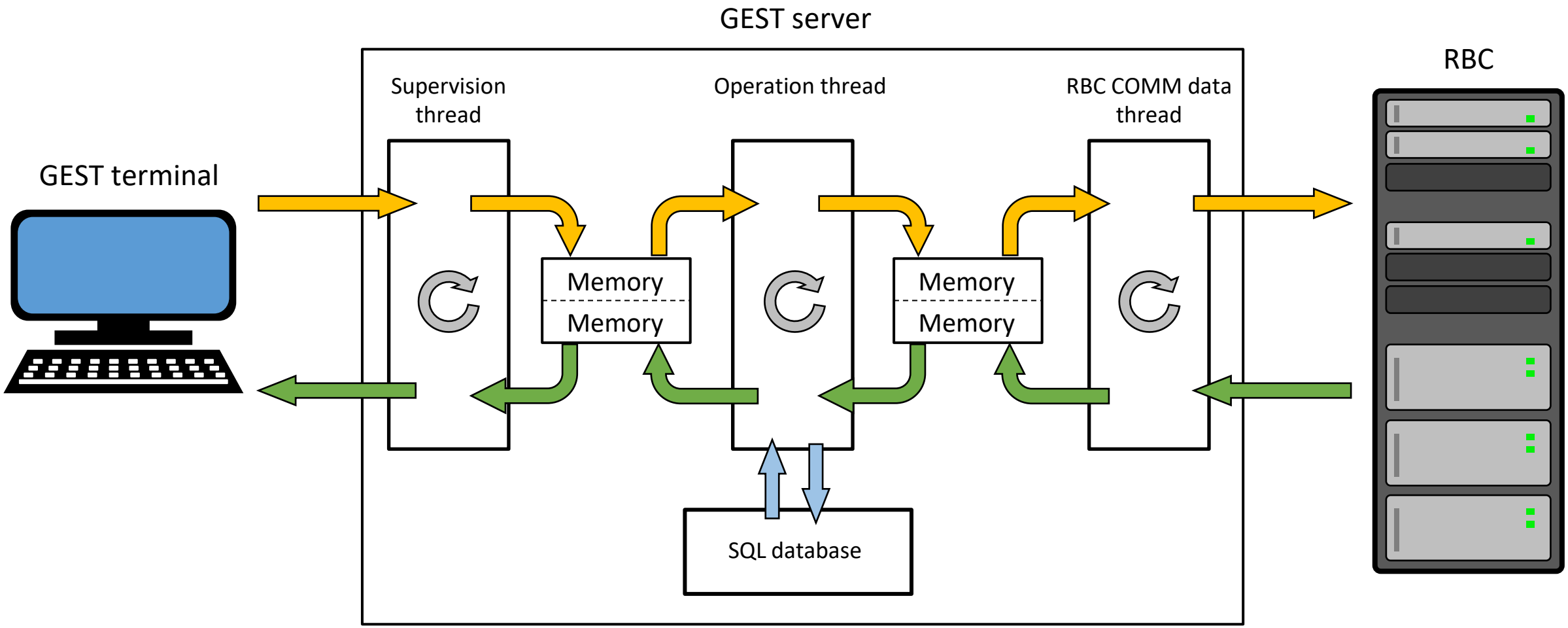


Establishing level of certainty

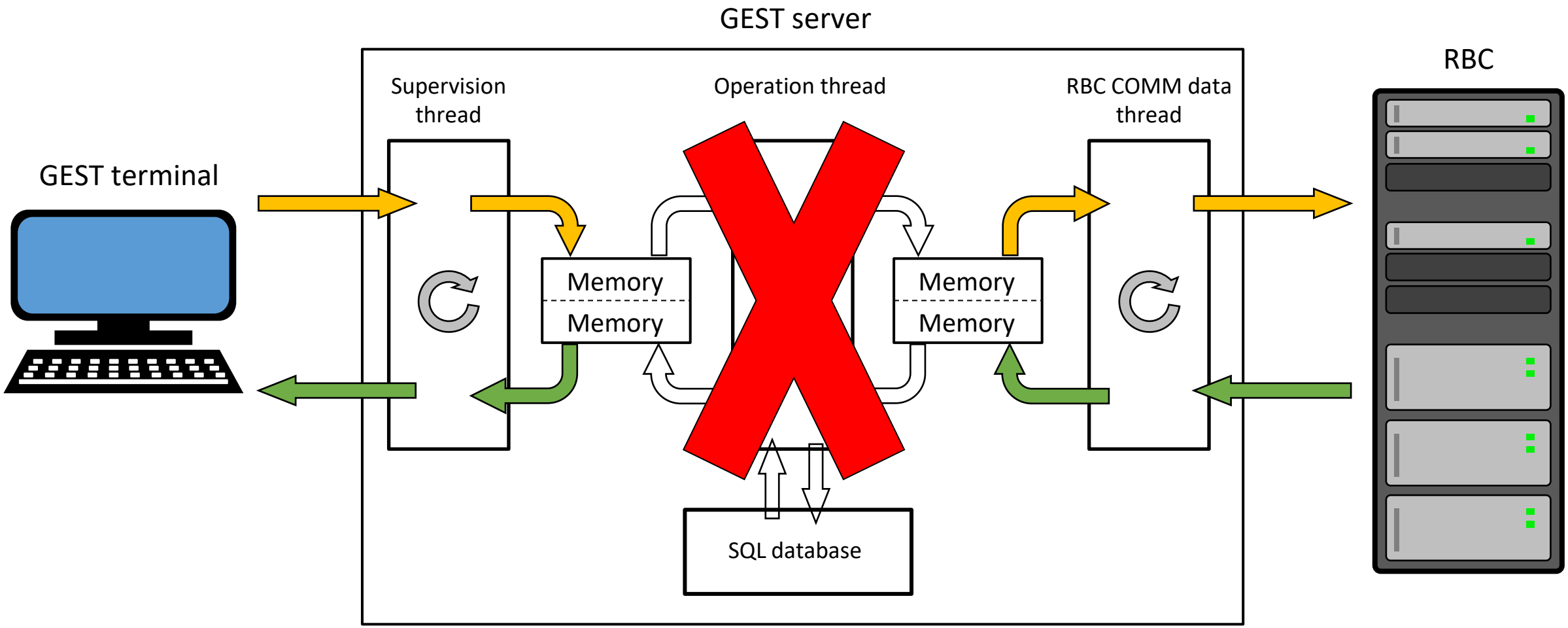
- A valid failure mode **MUST** meet available data at time of incident:
- Does the identified failure mode require conditions that are present or, absent in logged data from the failure;
- Does the failure mode present to users as described at the time; and
- Does the failure mode leave a data footprint which matches data logs.

- Are there any other failure modes which meet the failure conditions which cannot be discounted by the evidence?
- Finding one matching issue, doesn't mean there isn't another, the possible number of failure modes is limited only by the available evidence…!!

GEST Server failure



GEST Server failure



Interpreting the failure mode

- The GEST Server software was unable to detect and manage the corruption of its database which meant it could stop working with no indication that it had failed.
- Safety critical software systems are reliant upon barriers such as:
  - integrity of information passing between sub-systems;
  - checksum communication and encryption;
  - defensive programming to guard against corrupted inputs;
  - robust exception handling to deal with rogue operations;
  - independent watchdog and heartbeat monitoring;
  - action and feedback provided through independent paths.

GEST system failure summary

- The system was intended to have a high level of safety integrity but did not achieve this in the circumstances of this incident.
- These shortcomings had neither been detected nor corrected during the design, approval and testing phases of the Cambrian ERTMS project because:
  - the safety related software requirements for the GEST software were insufficiently defined;
  - the hazard analysis process did not mitigate against, the GEST software thread failure mode;
  - the validation process did not ensure that safety requirements for the correct display of TSRs were met; and
  - GEST was accepted into service without the production of a generic product safety case (or equivalent).

Designed in errors

- Software complexity can blind the lower level designers and software programmers leading to omissions in the intended integrity.
- Sub-system interaction can have the unintended consequence of moving a safety critical function from one sub-system to another, lesser integrity sub-system.
- Development is requirements driven, if a function of the software has not been specified it is highly unlikely to be provided in the end product.
- Missing requirements (therefore functionality) happens, especially those requirements which are obvious in hindsight!
- Development against V-model defined for safety critical railway applications in standards (EN 50126-50129), Additional guidance provided by RSSB in rail industry guidance note GEGN8650.

Omissions in approval process

- Approval/acceptance is based on safety case justifications, independent assessment is risk based sampling without ‘deep dive’ into design and programming.
- Safety cases can be very complex and subject to a commercial bias to justify a safe product rather than exposing potential risks/hazards and robust mitigations.
- Has the product been used elsewhere, was the omission found in other applications of the same product and effectively mitigated.

Any questions?

…….like so many ‘paper-based’ solutions, they [safety cases] are open to abuse and lassitude and can become a ‘comfort blanket’ to keep one warm from the chill of having to face the realities of multifarious risk.

In some domains, Safety Cases had become part of the problem, not the solution.

Taken from ‘The Nimrod Review’ Mr Justice Haddon-Cave