The Data Risk Zoo



In the October 2022 edition of the SCSC Newsletter, the authors introduced the concept of a "Data Risk Cygnology" that categorised types of *data safety risk*. Here they discuss further ideas in this area and work towards a more formalised taxonomy of terms.

Classifying data safety risks can help tease out some unlikely but severe failures of systems involving data. It is possible that sometimes these risks are already considered in system hazard analysis sessions but discounted as too unlikely to happen. We believe that having a more formalised and visual approach provides a useful "Tool in the Box" when trying to identify extreme data-related risks of systems. Due to their serious impact, we believe that they should be included, assessed and managed alongside more "normal" risks.

In our 2022 article we defined four new classifications of data safety risks:

1. Black Swan Data
2. Perfect Storm Data
3. Dragon King Data
4. Pudding Lane Data

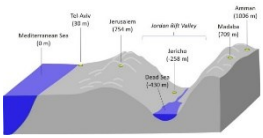
Since then, we have identified many other different types of data safety risk, some new and some variants of others. These broadly fall into three general categories:

- Dead Data
- Hidden Data
- Legacy or Change-derived Data.

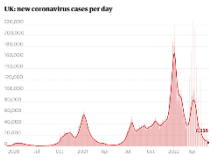
In this article we describe some of these new concepts and our proposed classification model to help corral the emerging "zoo" of data safety risks.

Data Risk 'Cygnology'[1]

The following table shows the original classification terms introduced in the previous article, the imagery that was developed and a reference to an example accident.

Type	Brief Explanation	Real-life Example
 **Black Swan Data**	*Data that is totally unexpected by those receiving it (i.e. 'out of the blue') and has huge (detrimental) impact, but in retrospect should have been anticipated.*	In 2013 a C-130 Hercules aircraft landed at Bar Yehuda airport near the Dead Sea [1], a saltwater lake astride Israel and Jordan. The airfield lies 1,210 feet below sea level.  The plane's navigation electronics were not designed to operate at altitudes less than 400 feet below mean sea level.
 **Perfect Storm Data**	*Combinations, sets or occurrences of data (or sets of absences of data) that would never have been thought possible to occur together, and when they do, have a large and undesirable impact.*	China Airlines A333 at Taipei on 14th June 2020 [2] when all primary computers, reversers and auto brakes failed on touchdown. 
 **Pudding Lane Data**	*Data that is unknowingly and unexpectedly critical to the whole operation and if missing or incorrect in some way, has a dramatic and negative impact. After the event it may be obvious that it was critical.*	 In 2015, an Airbus A400M on a test flight crashed with reports suggesting that as many as 3 of the aircraft's 4 engines failed during departure from Seville [3]. The reported cause of the failure was the torque calibration parameter data being wiped on 3 engines as the engine software was being installed, stopping the software from operating properly. In this case, the seemingly innocuous calibration data had a critical role to play in the safety of the entire platform.

[1] This is the term used in Risk Management for these types of risk

Type	Brief Explanation	Real-life Example
 **Dragon King Data**	*Data that might have been foreseen, but leads to an unexpected and explosive escalation with major impact ('things get rapidly out of control').*	 The loss of data relating to Covid-19 test results in the UK due to a row limit in Microsoft Excel led to the loss of 15,841 positive test results, which in turn meant that 50,000 potentially infectious people were missed by contact tracers and not told to self-isolate [4]. Because of the exponential way that viruses spread, this may have led to many more infections and deaths.

It is noted that the recent flight plan outage at NATS [7] which led to a week of flight disruptions, cancellations and delays, and cost approximately £300m in extra costs for airlines, would definitely count as one of these types above, probably Dragon King Data or possibly Black Swan Data. The knock-on effects of a small outage due to a data problem (duplicate Waypoints outside UK airspace in the flight plan) quickly escalated as the airlines had little resiliency in their operations. This incident was discussed by both the Data Safety Initiative and Safer Complex Systems working groups.

The Plot Thickens…

We don't always have the opportunity to work on new or 'green field' systems. We often need to maintain legacy systems that may have been running for many years if not decades. Imagine therefore that you have been assigned to maintain a large, complex and critical software system, which has evolved over many years. You are faced with many code modules and data files, some of which are undocumented, with the original development team long gone; you simply don't know what some of the data does in the system.

Typical problems we might see with such systems are:

- Configuration or initialisation files – what combinations of values are allowable and meaningful?
- Repurposing of database fields from their original intended use
- Translations and mappings built into the code with no explanations
- Ill-defined default values
- Special data workarounds done manually (it always does this, just enter X and carry on)
- Not all values are used or reachable ("dead")
- "Magic numbers" – literal items of unknown derivation

From these examples, we can draw three new classifications of data safety risk:

- **Dead Data**: data that is not, (or thought not to be), available or used – but in some cases should be
- **Hidden Data**: data which is intentionally or mistakenly concealed or hidden
- **Legacy or Change-derived Data**: data that is related to change and evolution of systems.

Dead Data

Data that is not, (or thought not to be), available or used – but in some cases should be

Computer science has had the concept of "Cruft" since at least the 1950's and is defined in Wikipedia [5] as:

anything that is left over, redundant and getting in the way. It is used particularly for defective, superseded, useless, superfluous, or dysfunctional elements

There are analogies in the aviation world, for example, the software assurance guidance for airborne systems (RTCA/DO-178C) describes the need for consideration and handling of:

- **Deactivated Code**: code that cannot be executed, such as unselected functionality or unused library functions, or data that is not used
- **Dead Code**: Executable Object Code (or data) which exists as a result of a software development error, but cannot be executed (code) or used (data) in any operational configuration of the target computer environment

We therefore can draw these ideas into the data world by defining the general class of **Dead Data** and a subtype of Dead Data called **Deactivated Data:**

- Data intentionally unused due to current configuration of the system
- However, it could be used due to a reconfiguration, system change, change of use, or a failure
- The data may be used in the wrong configuration (or if the deactivated configuration fails to be tested) when there are functional changes

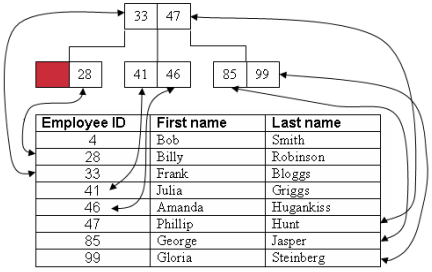
```
a=10;
if mode is "debug" then
   a=100;
```

Hidden Data

Data which is intentionally or mistakenly concealed or hidden

Data might exist in a system but the method of access: e.g. index, map or hash table has become lost or corrupted and so there is no easy way of getting access to the data.

In the example opposite, the loss of an intermediate index node might make it impossible to find the attributes for a given search key (Id=4).



Employee ID	First name	Last name
4	Bob	Smith
28	Billy	Robinson
33	Frank	Bloggs
41	Julia	Griggs
46	Amanda	Hugankiss
47	Phillip	Hunt
85	George	Jasper
99	Gloria	Steinberg

We therefore define this as a subtype of Hidden Data called **Treasure Island Data**

Data may also be concealed intentionally within other data and might even be considered "half-dead" if you know how to find it. Examples of this are hidden messages or watermarks in pictures and we use the term **Steganographic Data** for this type of Dead Data.

Legacy or Change-derived Data

Data that is related to change and evolution of systems

Systems that have been in service for long periods are often changed over time by different people and the 'memory' of the original purpose and intent of the data can be partially or fully lost. Sometimes there is a desire to modernise the system with new contemporary systems that have different models for managing the data. However data doesn't therefore always map or fit in a field, it may be mapped incorrectly, lost, truncated or set to a default or null value.

Consider the following healthcare example, where a legacy text-based prescription system is being replaced by a new digital system where the number, type and frequency of medication dosage is held in specific fields. This translation works for most until we encounter a PRN instruction (Pro Re Nata – "as and when required"):

Old system (text)

Medication Description
25 ml twice per day
1 tablet once per day
1 tablet PRN



New system (digital)

Num.	Type	Freq.(hrs)
1	25 ml	12
1	tablet	24
How to represent PRN?		

There is no obvious translation for this, and so special provision will need to be made and this subtlety may be lost in the migration process.

In a sense this is trying to squeeze data into something it doesn't fit and so we define the term **Square Peg Data** for this class of data problem.

"Magic Numbers" were mentioned earlier and defined (amongst others) in Wikipedia as:

- A unique value with unexplained meaning or multiple occurrences which could (preferably) be replaced with a named constant

The use of magic numbers can result in the provenance of the data being lost and hence it is difficult to verify that the value is correct and continues to be correct. (e.g. an unexplained value of 365 will be incorrect 25% of the time if it is actually representing number of days in the year, due to leap years).

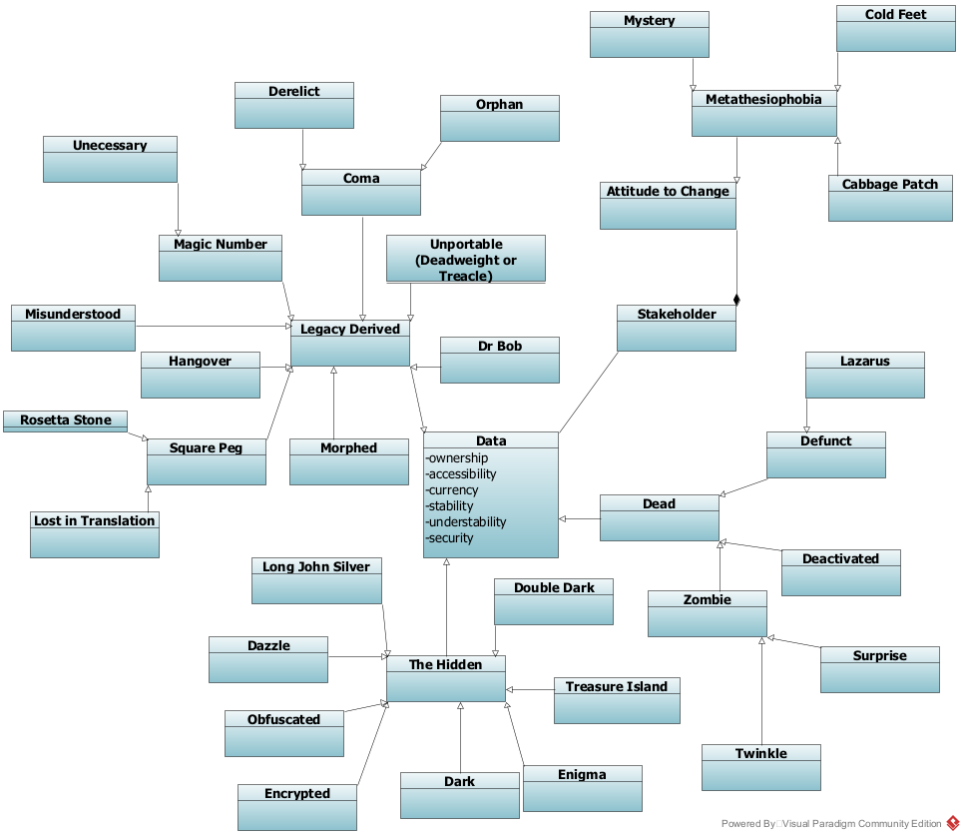
```
clear all;clc;
%Simulation of Transformance Performance
zw=4+5*i;
zl=80:10:200;
V1=220;
V2=V1/0.5;
I2=V2./zl;
n=(V2.*I2)./((V2.*I2)+(I2.^2)*4+40);
xlabel('Load')
ylabel('Efficiency')
title('variation of efficiency with loadi
plot(zl,n)
```

If the value is also used in different places, then it will be duplicated, which places a burden on the developers to ensure the values are all consistent. If the value does need to be updated, then the system may become inconsistent unless all cases are found. We therefore define the term: **Magic Number Data** for this type of data risk.

Even if good practices are adopted and literal constants are used instead of hard-coded values, then it is possible that new literal constants are introduced into a system during maintenance, when an already-defined constant should have been used (usually because the maintainer is unaware of the previous definitions). For this scenario, we define this sub-type of Legacy or Change-derived Data as **Unnecessary Data.**

Zoo Keeping

As the number of different terms expand, it is useful to try and corral these and identify the relationships more visually. The following taxonomy has been developed to encompass all the terms identified so far.



The following table describes the other terms not already covered.

Dead Data Subtypes

Name	Explanation
Defunct Data	Data that may have been initially used in the system by design, but subsequent changes make the data unneeded, unreachable or unusable.
Lazarus Data	Data that should not be defunct, i.e. it is currently unused due to a mistake (e.g. incorrect change applied), but actually should be currently used, incorporated or available within the system.
Deactivated Data	Data intentionally unused due to current configuration of the system. However, it could potentially be used due to a reconfiguration, change, planned reactivation or a failure. There are therefore risks if the data is used in the wrong configuration (or if the deactivated configuration fails to be tested) when there are functional changes.
Zombie Data (Undead Data)	Data thought to be defunct but in fact is used, (perhaps in an obscure way), perhaps in parts of the system rarely used.
Surprise Data	Data that is assumed lost, deleted or unavailable but can resurface and become unexpectedly available when conditions arise.
Twinkle Data	Data that is temporarily unavailable or intermittently available for whatever reason, on a regular or irregular basis.

Hidden Data Subtypes

Name	Explanation
Treasure Island Data	Data to which the method of access (e.g. index, map or hash table) has become lost or corrupt so that, although the data still exists, there is no easy way of finding it (or getting to it).
Enigma Data	Data to which the encryption key or keys have become lost so that, although the data still exists, there is no way of decrypting it.
Steganographic Data	Data that is concealed intentionally within other data.
Encrypted Data	Data that is specially coded and only accessible using a key or keys.
Obfuscated Data	Data that is purposely made difficult to understand, read or use.
Double Dark Data	Data you are not aware of and have no idea what it does, and therefore cannot be changed intentionally. However it may be changed by mistake and have significant effects on the system. For example, deleting an empty folder that contains a hidden configuration file used by the system.
Dark Data	(Requires further work on Dark Data types).
Dazzle Data	(Requires further work on Dazzle Data Types).

Legacy or Change-Derived Data

Name	Explanation
Square Peg Data	This is data where on migration or on translation between systems data doesn't map to a sensible new category or type, or doesn't fit in a field width, so it is mapped either to something which is not quite right, truncated or set to a default or null value. I.e. you are trying to squeeze it into something that it doesn't quite fit, and losing information in the process.

Name	Explanation
Rosetta Stone Data	1-1 mapping could exist but has to be identified or defined.
Lost in Trans-lation Data	No 1-1 mapping exists – some fudging has to be done, or data has to be rejected.
Coma Data	Data that is unused for now, and as such never gets updated as it should to current values or formats. If it is ever used (e.g. due to change, system re-configuration, or change of use) it may therefore present a problem.
Derelict Data	Data that is known about but has not been updated / maintained and is no longer fit for purpose, and so is 'worked around' (e.g. overridden) at every use or invocation of the system.
Orphan Data	Data that is in current use but no longer has an owner or sponsor so may never be maintained or updated.
Hangover Data	Legacy data that has been left over from installations or previous versions of the system and should not be needed or used, but it may be.
Magic Number Data **(Disassociated Data)**	Data that is rarely changed and is replicated and distributed throughout a system (typically obscure numbers in software or configuration files). These items of data should in fact all refer to the same value or set of values. Because it is duplicated, individual updates can be applied and the system becomes inconsistent.
Unnecessary Data	This is where new literal constants (numbers, strings) are introduced into a system during maintenance or upgrade, when in fact an already defined constant value should have been referenced. The person adding them is usually unaware of the previous definitions.
Dr Bob Data	Data, often obscure values found in software files, which nobody in the maintaining organi-zation knows the source or derivation.
Unportable Data (Deadweight or Treacle Data)	Data that, for whatever reason (e.g. derivation is unknown or doesn't work in new schema), cannot be ported or migrated to a new system and therefore is left out.
Morphed Data	Data that originally had a defined single purpose or set of purposes, but has since been re-used and re-purposed so that it may be influencing many more things and you don't know what it impacts.
Misunderstood Data	Data you are aware of and think you know what it does (but actually you don't), therefore any changes made may be incorrect and cause problems.
Cold Feet Data	Data you are aware of and understand what it does, but are afraid to change its format or representation because you are unsure what the impact will be (e.g. update to latest schema, version of software or version of database).
Mystery Data	Data you are aware of but have no idea what it does, and therefore are reluctant to modify or update.
Cabbage Patch Data	Data where you know what it does but not why, therefore are reluctant to update. This is generally data that you don't know where it came from or how it was generated or derived so it is a matter of trust that it is correct.

Future Work



This work is still subject to ongoing development but is hopefully starting to provide some structure and insights around the many issues that can affect the data in systems and therefore different routes to contributing to safety-related hazards. The use of a taxonomy to help visualise the relationships seems useful but other methods of visualisation are possible.

In particular the "Risk Zoo" opposite [6] for standard risk management provides an effective means of showing how different types of risk relate to likelihood and severity of consequence and follows the animal theme. This particular zoo contains animals not previously considered, such as: Boiling Frogs, 800-lb Gorillas and Red Swans!

Cockroaches: events that are certain because they are pervasively present and difficult to extinguish. Because impacts are low, they are dealt with one-by-one and don't generally overwhelm the systems or people that they affect.

White Elephant: known events that we fail to act against based on the classic "elephant in the room" saying. These are big problems that we could do something about, but a combination of availability bias, optimism bias, confirmation bias, groupthink, myopia, and status quo bias can lead potential efforts astray.

White Swan: an event that is known. It is also highly certain, with occurrence that is well understood, and has a resulting impact that can be estimated easily. In short, we expect this, we understand it, we should be equipped and prepared for these kinds of events.

800-lb Gorilla: existing forces (people, businesses, governments) that can dominate the culture around them and the hands of others because of their influence and power.

Gray Rhino: a highly probable, high impact yet neglected threat: kin to both the elephant in the room and the improbable and unforeseeable black swan. Gray rhinos are not random surprises but occur after a series of warnings and visible evidence.

Boiling Frog: risks that change incrementally in time, but those changes are not taken seriously until it is too late.

Ostrich paradox (or effect): a common disaster risk metaphor used to illustrate how humans have difficulty dealing with looming disaster risk by burying their head in the sand when danger looms.

Butterfly Effect: small changes in the state of a complex systems can result in significant differences somewhere else, sometime later.

Black Elephant: a cross between a Black Swan and White Elephant where we know that one day it will have vast, black-swan-like consequences.

White Leopard: a stealthy creature that stays hidden through effective camouflage. You know it exists, but it remains hidden until it is almost too late.

White Whale: very low probability events that are known to exist as a foil to the black Swan event that is symbolic of an event that is completely surprising.

Black Turkey: an event that is everywhere in the data – it happens all the time – but to which one is wilfully blind.

Gray Swan: A rare, but known hazard, with some predictability (though perhaps not certainty), and has the potential to cascade into a high impact event.

Red Swan: a known, probable event that could create a significant impact, but eventually does not happen. These are dangerous for the future because they can erode trust between the institutions giving the warning (e.g., scientists) and the public.

We could easily imagine this 'zoo' risk model being mapped to cover data safety risks!

Work is also underway to provide more guidance and recommendations around how the risks around each item in the taxonomy can be managed. Mike and Paul would be very interested to hear from you if you would like to contribute to this novel and interesting area of research – are there more exotic animals out there we've yet to discover? Let us know!

Image attributions
Top image © Jpldesigns | Dreamstime.com
C-130: © Mr Paul Hanley | Dreamstime.com
A400M © Abhishek Singh | Dreamstime.com
Air China: https://commons.wikimedia.org/wiki/File:China_Airlines_A330-300%28B-18352%29_%285213874397%29.jpg share under Creative Commons Attribution-Share Alike 2.0 Generic license.
Covid chart: data.gov.uk published under the Government OGL license v3.0
Swan icon: © Skalapendra | Dreamstime.com
Storm icon: © Андрей Морозюк | Dreamstime.com
Pudding icon: © Amonrat Rungreangfangsai | Dreamstime.com
Dragon icon: © Kuzzie | Dreamstime.com
Index: adapted from https://en.wikipedia.org/wiki/File:Btree_index.PNG shared under Creative Commons Attribution-ShareAlike 3.0 License.
Code extract: https://commons.wikimedia.org/wiki/File:Matlab_Code_-_Variation_Of_Efficiency_Vs_Loading.jpg shared under Creative Commons Attribution-Share Alike 4.0 International license.
Risk zoo: © Yolanda C. Lin (2020). The Risk Zoo. Available at: datartathon.com/theriskzoo/
Blackett review risk classes: https://assets.publishing.service.gov.uk/government/uploads/system/uploads/attachment_data/file/278526/12-519-blackett-review-high-impact-low-probability-risks.pdf, © Crown copyright, shared under the Open Government Licence.

References

[1] C-130 Crew Gets a Rude Shock When They Fly Their Plane Below Sea Level, https://www.popularmechanics.com/military/aviation/a26598/c-130-sea-level-dead-sea, accessed August 2022

[2] A China Airlines Airbus A330 Lost Primary Computers After Landing, https://simpleflying.com/china-airlines-a330-lost-computers, accessed August 2022

[3] 2015 Seville Airbus A400M crash, https://en.wikipedia.org/wiki/2015_Seville_Airbus_A400M_crash, accessed August 2022

[4] PHE statement on delayed reporting of COVID-19 cases, https://www.gov.uk/government/news/phe-statement-on-delayed-reporting-of-covid-19-cases, accessed August 2022

[5] Definition of "Cruft": https://en.wikipedia.org/wiki/Cruft, accessed Sep 2023

[6] Welcome to the Risk Zoo! https://datartathon.com/theriskzoo/#Welcome_to_the_Risk_Zoo, accessed Oct 2023

[7] Coincidentally-identical waypoint names foxed UK air traffic control system https://www.flightglobal.com/safety/coincidentally-identical-waypoint-names-foxed-uk-air-traffic-control-system/154824.article, accessed Sep 2023

Paul Hampton Safety Consultant at CGI and SCSC Newsletter Editor

Mike Parsons, Safety Consultant at Ebeni and Director of the Safety-Critical Systems Club

Paul is a Chartered Engineer with over 30 years' experience in IT. He has spent 15 of those designing and developing enterprise systems in sectors such as Energy & Utilities, Government, Criminal Justice, Health and Aviation. He has been involved in Systems Safety for 15 years in a variety of capacities including: safety engineering, safety management, independent auditing and corporate governance and assurance.

Mike has over 30 years of Safety Engineering and Management including: Development of Safety Management Systems, Assessment of Software, Services and Data Risk, Systems Safety Analysis on Development and Service Projects, Safety Assessment, Audit and Reviewing and Systems Engineering. In 2019 he became the Managing Director of the SCSC and is lead for several of the clubs working groups including Data Safety, Service Assurance and Safer Complex Systems.