international journal of critical infrastructure protection 38 (2022) 100535

Contents lists available at ScienceDirect

International Journal of Critical Infrastructure Protection

journal homepage: www.elsevier.com/locate/ijcip

The drift of industrial control systems to pseudo security

Peter Donnelly *, Mabrouka Abuhmida, Christopher Tubb

Computing, and Mathematics, University of South Wales, Pontypridd CF37 1DL, United Kingdom

ARTICLE INFO

Keywords:
Industrial control systems
Security
Safety
Complexity
Interdependence
Integration

ABSTRACT

With the promise of a synergistic impact on the Efficiency-Thoroughness Trade-Off, Marketing is increasingly promoting types of Industrial Control Systems (ICS) that by some means combine the two usually segregated core ICS functions into one. A Basic Process Control System (BPCS) is combined with a Safety-Instrumented System (SIS) in a physically integrated form factor using shared resources of some kind. This paper suggests such a strategic choice of technology can result in functional safety (FS) hazards or security vulnerabilities, giving rise to resilience concerns. It takes a sceptical view of such an approach and instead proposes strict segregation of such functions and resources. In the context of critical national infrastructure (CNI), where potentially high consequence events (HCE) may arise from unplanned incidents, the outcome of this paper is to warn against the use of such architecture - even beyond that arena. Both ancient and modern, yet similarly strategic, historical decisions are used as metaphors to illustrate how sometimes insufficiently scrutinised technologies can be later regretted. Practical technical, organisational, and cultural measures are offered to steer against the headwind of commercial pressures in promoting integrated FS and security of the BPCS-SIS environment. A contribution is made of evidence-based, business intelligence gathering measures for BPCS-SIS vendor selection together with a proposal for an alternative, adopted application of proven Uncertainty Assessment Reporting techniques for industrial certification bodies and business stakeholders alike.

1. Introduction

CNIs include energy, sewage, water, food chains, manufacturing systems; and infrastructures that manage or control them are regarded as safety-critical systems. ICSs are used a) to control the physical equipment under control (EUC), b) to protect them against losses including human, material and environmental, and to ensure the minimum operations of the economy and society. ICSs help to ensure EUCs and protected processes will function within their design limits when called on to do so.

Functional safety (FS) concerns itself with safety hazards caused by ephemeral maloperation of control systems on EUCs, as opposed to physical safety, which concerns itself with hazards that are constant and can be removed, for example, by good design. The focus here is on both security and FS, with physical safety being seen as a consequence of each.

While the objective of FS is to prevent losses arising out of the mainly unintentional actions of benevolent actors, the overlapping objective of security is to prevent losses arising out of the mainly intentional actions of malevolent actors, yet with a common objective to prevent losses – human, material, environmental, and hazard manifestation. They differ, however in the intent of the actors. The resultant characteristics are described as being emergent properties that arise out of the interaction of an interdependent and complex system or system of systems [33,19]. The emergent properties are dynamic.

Independent of any BPCS and often software-driven, are safety systems, or SISs, that enable an additional layer of functional safety by implementing one or more Safety Integrated Functions (SIF). This layer rests on foundations of probability calculations and statistical extrapolations used as proof of some theoretically achievable Safety Integrity Level (SIL) of a vendor's safety system. These intervene when necessary and without influence from any BPCS. Notably, the safety proofs cannot usually be demonstrated empirically, e.g., 1 failure in 25 years or more of theoretical service is unprovable by proof testing means. At the outset, therefore security and safety are based on trust, confidence, and assumptions.

Characteristically, an SIS should exhibit at least five aspects of complete independence from a BPCS, including independence of 1) power supply, 2) cabling, 3) sensors/actuators, 4) logic solver, and 5) memory. Others may include human division of dedicated safety

* Corresponding author.
E-mail address: Peter.Donnelly@southwales.ac.uk (P. Donnelly).

https://doi.org/10.1016/j.ijcip.2022.100535
Received 3 November 2021; Received in revised form 13 April 2022; Accepted 15 May 2022
Available online 16 May 2022

responsibility, physical remoteness from the BPCS, segregated and special physical and virtual security protection measures. Separation of maintenance and test procedures are mandatory but not always adhered to. An SIS, therefore, ensures the EUC remains in a safe state by only intervening to return an out-of-bounds EUC behaviour to some safe state by triggering alerts, alarms, interlocks, safety or emergency measures [18]. An ICS forms the entire, combined BPCS and SIS safety control structure, a system which includes organisational and human behavioural aspects that helps to ensure the absence of unacceptable levels of risk of hazardous events occurring due to fault-driven functional failures to ensure safety.

Sometimes defined as the absence of unacceptable levels of hazardous risk, yet unlike principles of many physical engineering disciplines, FS and its close ally security are ephemeral in both time and space. They can drift away or suddenly vanish in a pernicious way [20]. However, security and FS differ fundamentally with regards to their aetiology. Whereas FS is rooted in events that, in the main, are accidental and only sometimes occur as a consequence of a security incident [20], security is rooted, in the main, in events that are intentional. Secondly, while FS concerns itself with the primary consequences of events, security is concerned with the secondary consequences and less so with any immediate physical damage. For example, the immediate consequences of a safety incident are often self-evident, but those of a security incident are not [20]. In the case of an ICS, the causes of a security breach can be accidental or malicious and its consequences either immediate or deferred and vary in magnitude from the benign to the catastrophic.

2. Faults, failures, and hazards

The defensively safety-conscious mindset of industrial engineering practice assumes that a BPCS will harbour unknown faults that may accumulate to become worse, at which point a SIS must intervene. Apart from hardware-based or solid-state (and software free) logic solver SIS's such as Yokogawa's Prosafe SLS, used in very high integrity environments, e.g., very high-pressure pipes, a SIS will usually use some element of software, though not always use safety-critical software that may, in turn, also harbour unknown faults – depending on how well, i.e., its integrity, it was produced. Dispensing with software, the analogue computing development community such as Bernd Ulmann's [41] shows promise in this respect. In the event of a SIS failure where software is used, its quality and how well or poorly its failure modes have been designed will influence the extent of losses.

Whether hardware or software-based a fault is any change in the state of an item that is considered anomalous and may warrant corrective action. On the other hand, a failure is the inability of a system or component to perform its required functions within specified performance requirements. It is useful to recall that a fault may or may not lead to a failure, one or more faults can become a failure, but all failures are the result of one or more faults, and shutting down an entire system based on a small number of faults may not always be a feasible option to follow. Depending on a specific scenario, any number of faults may lead to a critical failure, leading to a hazard defined by [32] as a state or set of conditions of a system or object that will lead to a loss event.

3. Safety-critical SIS software

Software is hazardous if it can directly lead to a hazard or is used to control a hazard i.e., (a) hazardous software includes all software that is a cause of a hazard, (b) is a hazard control, (c) provides information upon which safety-critical decisions are made, d) is used as a means of failure or fault detection.

Software is also safety-critical if it complies with any of the many categories explained in detail by NASA [36]. Safety-critical software includes hazardous software (which can directly contribute to or control a hazard). It also includes all software that can influence hazardous software. If the hazardous software resides with safety-critical software on the same physical platform, it must also be considered safety-critical unless adequately partitioned from the safety-critical portion. The term "adequate" remains contentious.

Non-safety-critical software could lock up the computer or write over critical memory areas when sharing a CPU or any routines with safety-critical software. An SIS is or should therefore be regarded as safety-critical and any BPCSs, which are commonly general-purpose controllers, and which may share resources with a SIS has to be similarly regarded as safety-critical. Techniques such as firewalls and partitioning can be used to ensure non-safety-critical software does not interrupt or disrupt the safety-critical functions and operations [36], though these devices are not perfect.

Firewalls are not infallible in securing networks and, given how some vendors can be reluctant in sharing details of firewall security test reports, or sometimes making implied and difficult to verify security claims, then strong credibility cannot always be attached to them. Using them to secure a SIS is therefore itself sometimes questionable. Secondly, the 2018 Survey of Embedded Systems Safety and Security conducted by the Barr Group reveals some insights into the absence of quality in the production processes of a range of safety-critical software [4].

The selection of BPCS and SISs is strategic in nature and cannot be swapped and changed at short notice or at low cost: choice of vendor is critical. Such BPCSs and SISs are costly to design, build and operate and any savings which can be achieved are often exploited (as indicated by the above survey). Hence, the drive for "Faster, Better, Cheaper" (FBC) can be so enticing that security in supporting safety and resilience requirements may be jeopardised when BPCS-SIS integration is chosen as a solution path. This can lead to any number of faults up to and including a functional failure of high severity.

As complexity shows little sign of decreasing with integration a noticeable tendency, then functional failure caused by faults arising from security, resilience or safety, can be encountered at the very intersection where system complexity and interdependence meet [21]. Balancing the requirements of security, resilience, safety, and complexity vs. cost is, therefore, challenging and results in outcomes that are never, ever certain; an element of uncertainty will remain.

4. Sources of uncertainty in SIS design

Just as there is no guarantee of complete security, neither is there any guarantee of complete safety. In process design, the Safety Requirements Specification (SRS) produced early-on in a project will detail responses to three strategic questions: (a) what can go wrong? (b) what can be the causes? (c) what are the consequences? i.e. – the failure modes and effects analysis (FMEA). Responses to all three will always be extensive, detailed, and comprehensive, yet remain incomplete. There will remain a residual level of the acceptable, perceived risk of loss which cannot in every case be calculated in advance – before even the system is operational. There will also be a level of unperceived risk – that which was not anticipated. In distinguishing between risk and uncertainty [40] describes how, in a given scenario, risk assessment believes to know all possible outcomes and attaches probabilities to them versus uncertainty where there is the awareness that all possible outcomes of a scenario cannot be known in advance. This is reflected in the reliability assessments of SIS calculation models (not covered here), which explicitly addresses three areas of mathematical assessment uncertainties; model uncertainty, data uncertainty, and completeness uncertainty [24]. At the very least, it is completeness uncertainty that not only remains knowingly incomplete but is dynamic and growing as the interaction of hardware, software, networking, integration and human activity, such as maintenance or optimisation of productivity, merge at a rate not seldom driven by the power of Marketing departments and 'C' level executives. Yet, even without BPCS-SIS integration, security and safety are implied permanent sources of uncertainty. Not seldom do the complete safety and security of a production system rest on the

probability calculations of the Safety Integrity Level (SIL) of a vendor's safety device, the very same device security devices attempt to protect. In the case of integrated BPCS-SIS, a fourth area of explicit uncertainty should also be addressed: the emergent, resonant, dynamic property of complexity-interdependence-uncertainty (CIU) countered only by its inherent system resilience or its absence.

5. Resilience

Systems exhibiting an absence of unacceptable levels of risk (implicit FS) are not necessarily resilient, for it is not unusual for FS to be evident in a system yet not be resilient [21]. For example, a system that frequently fails safe (a spurious or nuisance trip) will be implicitly classed as being unreliable and, by extension, non-resilient. Moreover, we cannot talk of a resilient system unless it has evidenced this over some appropriately lengthy period and has demonstrated its survival or value based on conventional performance or economic measures or its sustained resistance for example, to penetration testing or "hacking". Just as security and FS enjoy some common ground and are mutually reinforcing, they also have a common Achilles Heel; it is not possible to anticipate all possible FS or security scenarios at the design (SRS) outset [7] and, as complexity and interdependency change and grow, they increasingly become the enemy of both. It is resilience, however, which keeps things going, but in return, it demands a level of temporal knowledge of the past, the present, and the future in order to do its job.

Resilience has been defined by Erik Hollnagel as an organisation's ability to adjust its functioning before, during, and after a significant disturbance, thus enabling adaptation and operation under both anticipated and unanticipated circumstances [9]. Nevertheless, the fundamental characteristic of a resilient system is that it does not lose control of what it does but can continue at some level and recover and rebound to some previous level of performance, if need be. To remain in control it is necessary to know about past, present, and likely or unlikely future events together with knowing what to do and having the wherewithal to do something about it. Any loss of control is caused by a lack of time, knowledge, competence or resources. A critical pre-requisite of understanding levels of resilience, or its opposite brittleness, is knowledge of overt and covert dependencies as have been uncovered for example by the Australian Government in [3].

One marker of resilience capability is the ability to create foresight – to anticipate the changing face of risk before harm even occurs [40]. Resilience engineering offers here the potential for improved control by application of its four adaptive characteristics, namely: the potential to respond to disturbances, to monitor, to learn from, and to anticipate disturbances. [20]. Resilience engineering avoids an over-reliance on analysis techniques but examines the interactions across factors and sees accidents, including security incidents, as an emergent result of tight couplings of variable behaviours as described by [31] and constructively illuminated as uncontrollable variability of performance (UVP) or *functional resonance* by [19]. UVP is in mankind's nature.

Table 1.

6. Hazards of singleton technologies

Responding, for example, to ancient Rome's constraints on Imperial growth and the perceived potential threats to it, the first Roman aqueducts were built below ground for three reasons. a) concealed protection from enemies, b) as an additional layer of protection from deterioration, and c) to offer less disruption to normal life [2]. However, as architectural prowess came to dominate thought-leadership and complacency grew, aqueducts were built above ground. In time, as unanticipated hostilities escalated, those above ground were largely sabotaged or destroyed, with only a few below ground still functioning to some degree. Choice of technology can therefore have serious consequences, critically so a singleton-technology. Hence Rome's earlier and unfortunately limited choice of subterranean diversity of technology proved wanting. The Empire's aqueducts serve as a reminder of a) their strategic role, b) their vulnerability c) an example of how a singleton-technology approach can promote risk. Further historical events from the early 17th century Swedish Navy's doomed maiden voyage of the Vasa to 21st century Nasa's catastrophic failures of the Challenger in 1986 and Columbia in 2003, all bear witness to FBC-driven examples of the absence of resilience, namely brittleness [21].

The long-term, economically irreversible, and holistic characteristics, strategic nature of modern process design demand adherence to regulatory standards such as IEC 61508 and IEC 61511. But, while risk analysis forms a fundamental element of performance-based security and safety management, without which it is impossible to calculate the level of investment required to keep a system safe, there are many techniques used. The overall objective is to marry economics with resilience, safety engineering, and security in such a way as to ensure resiliency while avoiding over-spend. However, the seductive temptation of FBC always looms. Using independent layers of safety protection is regarded as one of the most desirable approaches.

Protection layers designed for a given hazard are said to be independent of each other if the probability of failure of one layer of protection X is completely independent of the probability of failure of another layer Y. On the other hand, if the probability of failure of one layer of protection X is not completely independent of the probability of failure of another layer Y then layer X is said to be dependant upon Y [26], and therefore risk may not be abated i.e. *The wellspring of risk is dependency* [15]. Three dependency determinants stand out prominently: (1) shared or jointly used components, e.g., the same electronic device or shutdown valve used by two safety systems offering protection from the same hazard, (2) same technology types such as the use of devices of the identical type used in two safety systems offering protection from the same hazard and (3) common operator or the same person operating both the control systems and the safety systems such as an emergency shutdown control. Case (1) is said to be a dependant protection layer where two or more layers share a common component, and that component fails or is compromised, then all safety layers change to a degraded or failed state. Cases (2) and (3) are referred to as a pseudo-independent layer of protection and are treated using common-cause safety calculation techniques, not pursued here. Irrespective of whether routable or non-routable digital technology is used, cases (1), (2) and (3) are said to represent singleton-technology phenomena. In the context of an ICS therefore, singleton-technology solutions tend to promote risk, do not promote safety, but instead the opposite of resilience, namely brittleness.

7. Resilience and brittleness

In their analysis of systems, Rasmussen-Sveding [39] describe a drift-to-danger model implying a permanent striving for maximum performance or profit, and implies that of all the forces acting upon and from within an organisation, for example, there are four prominent categories: (a) commercial market forces such as FBC, (b) an operational self-stabilised production space, (c) a "safety net" zone designed to catch outliers of hazardous events, and (d) a zone where accidents are more than likely to occur. Resilience, if present, is to be found centred and balanced in a "Three Bears" zone between these forces. Opposing

Table 1
Survey of software quality techniques used.

Software quality technique	% of engineers who failed to use it
Regression testing	41
Regular peer code reviews	43
Compliance with formal safety standard	38
Static analysis of code	33
Formal written coding standard	17

Source: [4].

resilience and in the absence of perfection then, to a greater or lesser degree, brittleness permeates throughout.

The measurement of resilience is challenging, and difficulties in directly measuring resilience are partly offset by some useful indicators or markers of both it and its absence, brittleness by indirectly conveying a subjective measure of aspects of resilience. These include technological, organisational, logistical, and financial factors. For example, having a level of preparedness in anticipating negative events, the ability to recover production, or resume the supply of critical services to some previous level. Similarly, the capability of having transparency of critical information or documentation when needed – in the absence of which complexity increases. Also, the ability to substitute resources or component parts in place of the originals or in the case of failures caused by random events to apply redundancy to systems [31]. In the case of potential faults caused by sentient attackers then many are aware of avoiding identical redundancy at all costs, which would otherwise only promote failure totally [14] and to concertedly seek diversity of mechanism or engineered differences instead [10].

8. Integration of control, safety, and security

It is widely viewed that designing-in functionality early on in design is not only easier and often produces a "good" product, but also results in lower total cost of ownership. This path is seductively attractive to vendors and customers alike but there can be surprises waiting. Designing industrial processes follows a rigorous, mandatory standards-orientated procedure involving many disciplines, including design engineering, control engineering, automation engineering, safety engineering, security engineering, and resilience assurance, as well as technical and trade disciplines. It is a strategic, long-term, economically irreversible, and holistic commitment to considerable capital expenditure.

Processes are controlled by a BPCS that is subordinate to a theoretically independent SIS, ensuring the safety of the controlled process. BPCSs and SISs employ some form of I/O controller such as a programmable logic controller (PLC), first introduced in 1969, and often incorporates a digital microprocessor increasingly combined into some level of mutual integration, sometimes applying the resources of a single device to fulfil the functional requirements in two separate control and safety functions. While PLCs are usually of a general-purpose PLC standard using a full-variability application software language such as SCL, SISs too are a type of PLC yet differ in at least three different ways: a) level of self-diagnostics, b) implementation of redundancy and voting, and c) independent certification of performance integrity. Further, unlike general-purpose PLCs, if a SIS employs software at all, it will frequently employ only fixed or, at worse, a limited-variability software such as ladder logic. To a greater or lesser degree, separation or integration of the two functional units varies greatly. Owing to the levels of integration used combined with digital route-ability of attached, networked equipment, then the number of potential interactions and their levels of complexity rise.

9. Reasons to separate

Reasons to separate the BPCS and SIS include (a) reducing common cause, common-mode, and systemic failures and minimising the impact of BPCS failures on the SIS, (b) retaining flexibility for changes, maintenance (such as process re-calibrating or tuning), testing and documentation of the BPCS, (c) to alleviate the validation and functional safety assessment of the SIS, (d) to promote the security of access and cybersecurity for the SIS such that the impact of relatively frequent revisions to the BPCS have minimum or no impact on the SIS, (e) to minimise the volume of analysis work required to ensure the SIS and BPCS are properly designed, verified and managed [22]. Any level of functional integration, on the other hand, is also motivated by differing rationales.

Often driven at higher levels of integration by FBC motives, at the basic or separated level of no integration – some say the safest level, demands that the BPCS and the SIS work completely independently from each other. i.e., they have independent power supplies, sense inputs via different and independent sensors such as pressure sensors, they process information using independent and preferably using distinctly different means, and they produce independent outputs to independent final control elements such as valves of differing types and manufacturers. Complete separation is self-evident, and the approach is adhered to by certain organisations.

10. Reasons to integrate

There is, however, considerable diversity of the "less separate" approach and, in the face of the aforementioned reasons to separate, a range of motives support the development of products less "separate".

A first level of "integration", referred to as "interfaced", uses a combination of BPCS and SIS still separate, sometimes from distinct vendors, but for reasons of ergonomics, practicality, economics, or even cultural reasons and rarely for security reasons, some connection is enabled to exchange data so that information can be converged or consolidated into a single common HMI display. This approach constitutes a growing portion of the entire current US market [23] and likely also elsewhere, a worrying development.

The second stage of "integration" sees the BPCS and SIS still as separate processors but being typically "made" by the same manufacturer with much closer integration between the two systems. They may, for example, have similar architectures and software common to both, even written by the same team. The HMIs may look very similar, and the configuration tools may be similar. From a hardware perspective, they may be very similar so that, again in pursuit of FBC, only one set of replacement parts is needed and one training course required. Also, there will be only one "TTT" or just one financial "throat to throttle" should things go wrong.

The third form of integration sees "common" systems, i.e., one single or "TTT" box comprising both the BPCS (control) and the SIS (safety). Numerous vendors offer these systems, including a number of high-profile names.

Vendors sometimes counter concerns of security or safety by either "flat-plating the bird" or citing formal certification issued by international certification authorities. For example, it is said that memory partitioning, separation of execution contexts, stack management, and use of firewalls ensure that security, safety, and non-safety programmes running in the same processing environment are actually separate and non-interfacing. However, for vendor-motivated integration to exist, there must be some level of coupling between the BPCS and the SIS. In the case of memory partitioning, for example, the memory itself is shared. Indeed, even if only minimally, there is likely to exist a level of upstream-downstream coupling similar to concepts described in detail by [19] which at some point may lead to what is referred to by him as uncontrollable variation in performance or functional resonance. Further, with increasing acceptance of how software per se can be a serious source of risk in ICSs as promoted by [16] and [17] one may therefore ask integrated product vendors what exactly their integrated BPCS-SIS systems do have in common and then closely examine both the security and safety implications for example along the lines of Consequence-driven Cyber-informed Engineering (CCE) promoted by the Idaho National Laboratory in [5]. Further, in view of the shortcomings, indeed even frequent inability, of engineering and IT disciplines to communicate by means of a commonly agreed standardised set of diagrammatic symbols applying Automation-ML (markup-language) techniques [13], then that area may be an opportune task with which to start before the level of integration has crossed some yet-to-be-identified irreversible Rubicon [15]. As complexity can be the enemy of security, then countering this integration mindset, some critics suggest the "complexity" of an "integrated" BPCS-SIS is of too great a magnitude to

ensure a required level of security and safety in some CNI environments [23]. This may also be the case in many non-CNI arenas of lower security levels yet where safety and complexity nevertheless pose fatal hazards.

The phenomenon of complexity is found along a continuum between structured regularity at the one extreme and pure randomness at the other [30], though in the case of software, it has been proposed that complexity can also be defined in terms of the minimum space a mathematician needs to describe it [30]. Products as complex as SISs harbour complexities of other varieties such as mathematical, technological, interaction, and complexity of decomposability as described by [29].

In a noticeably mild warning against the use of increased complexity, while promoting security and safety by means of independent BPCS-SIS system segregation, a statement can be found buried deep in the international process industry standard IEC 61,511. Clause 11.2.4. It specifies that "separation of control systems from safety systems is mandatory". Despite addressing the typical four physical areas of separation involving field sensors, final elements, logic solver and wiring however, apart from stating that neither failures of, nor work carried on the BPCS such as maintenance, operations, or modifications should impact the SIS (in section A.11.2.4), it fails to explicitly say why it should be done nor how it should be done. Importantly, it also fails to specify what is meant by the term "separation". In the pursuit of FBC these gaps are exploited by some vendors who "rationalise" system design and production methods to portray an attractively packaged product to sometimes insufficiently sceptical or overly hasty customers. Component commonality, particularly software or logic solver based, therefore can be seen as a source of cyber vulnerability, incremental risk or change in level of security. In light of some concerns of supply-chain risks as well as loss of control of offshored software development by some security systems vendors then there appears to exist warranted concern for the drift to a form of pseudo-security and pseudo-safety manifested by integrated BPCS-SIS systems, an overlap which in some cases may prove to be more than worrying. This highlights the issue of risk for which there appear to be only two classes of cyber risk worthy of the title of "risk to critical national infrastructure" [15], a) those which offer one single point of failure such as the global DNS system, b) that of cascade failure of CNI systems upon which we depend most. This second area is where concepts of complexity and interdependence intersect [30] and [15]; one such place is where integrated BPCS and SIS systems meet. Safety, security, or resilience or their absence are found there. As software is often used in both BPCS and SIS design, it can be enlightening to scrutinise the quality of real-life safety systems' software.

A short discussion now follows of reasons for caution and potential measures to consider when selecting SIS vendors and products.

11. Reasons for caution

Best practice development of quality, safety-critical software involves organising human effort in a way that produces a product that performs reliably and always in the desired way. It includes cultural factors as well as management ability and technical infrastructure and quality tools [12]. Central to this will be an appropriate process to follow. It may include test-driven development, regression testing, peer code reviews, compliance with formal safety standards, static code analysis, adhesion to written coding standards and include secure coding techniques. Reasons for using, for example, a programming language characteristic exhibiting fixed, limited, or full language variability may not be well-differentiated, but rather a matter of "taste". A four-year- old study highlights a range of closely related measures.

Encompassing 1703 professional embedded software systems designers from across the globe with an average of 16 years of paid experience, a 2018 survey [4] revealed some disturbing indicators. 2/3 of potentially injurious products under development by them targeted the four sectors of: industrial, medical, automotive, defence/aero. Of the quality techniques listed below, their usage profiles yielded:

Likely a cause for concern, one must, however, give recognition to quality. Few software development teams are awarded the US government's Level 5 ranking of the Software Engineering Institute (SEI). For example, as awarded to Lockheed-Martin's onboard Shuttle software group [12] or, more recently, the Boeing seL4 ("L" as in Jochen Liedtke, a major contributor to the field) microkernel orientated work in High-Assurance Cyber Military Systems (HACMS) [1].

The pursuit of FBC drives the adoption of new development techniques, with the range of "Agile" approaches being just some. Yet, while the Agile manifesto defines four development values and 12 development principles, most of them, if not all, appear to be at least in some opposition to the requirements of safety-critical software development [37]. Given the growth rate of safety-critical, software-driven devices, this could be a cause for growing serious concern. In the first instance, the focus should be on safety and security, not features and productivity.

Functional safety-engineering related to hardware, involving extensive context-sensitive and probability-based reliability and risk calculations, forms the overriding requirement for the operation of complex socio-technical assets. The calculations used and the prudent judgement applied in the form of good engineering practice (GEP), used to ensure safety and avoidance of hazards, can be lengthy and complex. Yet, international standards IEC 61,508 (functional safety), IEC 61511 (process industries) recognise only two categories of system failure: safe or dangerous; there is no allowance for a grey zone.

Despite the topic attracting strong debate, some vendors have adopted categories of faults, which may accumulate to emerge as failures not recognised by the international standards, and detract, subvert or undermine the calculation of the Safe Failure Fraction (SFF) which impacts the calculation of the Safety Integrity Level (SIL) of required safety measures. Such categories include, for example, "no effect", "no part", or "annunciation undetected", and there are others. There is also, it seems, insufficient consideration given for the interaction of the software with the hardware in complex yet critical circumstances.

12. Uncertain assumptions and over-simplifications

Reliability requirements for the safety instrumented functions (SIF) performed by a SIS are derived from a hazard and risk analysis task in accordance, for example, with IEC 61,508. The analysis frequently rests on a number of assumptions and simplifications about the nature of the SIS and its context. Of the numerous assumptions often made [25], a significant issue is that of the SIS not being influenced by any factors outside (rather limited) physical boundaries of the SIS. In the case of an integrated BPCS-SIS, that will likely be networked or interfaced in some way, the boundaries will be blurred and may be dynamic in nature, and it might be impossible to absolutely define those physical boundaries. Digital exposure may be a real yet unknown uncertainty and significantly underestimated. There may be other sources of unknown uncertainties but the issues of network intrusion or inherent vulnerabilities will, to some unknowable degree, be real. The resulting calculated probability of failure on demand PFD_{avg} of the SIS may be well below that which exists under real operational conditions and hence hide a dangerous undetectable hazard which, in the not infrequent pursuit of FBC, may be completely overlooked.

In pursuing FBC, the temptation is to evaluate and deploy at a lower cost. Coupled, however, with the uncertainty of possible unknowable vulnerabilities introduced by functional integration, and with proliferation of cyberattacks, then it appears prudent to avoid risk of emulating the complacency of the ancient Roman aqueduct designers with myopic admiration for Imperial architecture only. One does, however, hope any further drift to BPCS-SIS integration will not, in the longer term, result in our only remaining "aqueducts" being those found exclusively below ground.

13. Evidence-based vendor selection

Lockheed-Martin's former software development team used a 4 stage development process for its efforts in the NASA space programme [12]. The concept centres on a four-element code of practice (here, vastly oversimplified): 1. The product is only as good as the plan for the product, 2. The best teamwork is a healthy rivalry, 3. The database is the software database, 4. Don't keep fixing the mistakes – fix whatever permitted the mistake in the first place. This oversimplified description of their code of practice moved them forward and set new world standards in software quality. In search of evidence to help users select a vendor and product, I propose some indicators of such quality properties and combine a number of indicators of quality from both their world and from the world of business.

Given the costly consequences of mid-stream (or later) process design change, of a serious incident involving causality due to any aspect of a chosen system of BPCS or SIS is considerable, then choice of vendor must be made based on a strategic mindset; the decision has long-term impact, will be holistic in its influence and be economically irreversible. Approaching any CNI safety-critical system design should be regarded as a corporate existential issue and as such the UK's Defence and Security Public Contract Regulations, Chapter 15 (DSPCR) [35] offers much solid advice – directly applicable to the non-CNI, SME commercial world too.

It constructively addresses supplier selection and intelligence regarding: a legal framework, the mandatory exclusion criteria of suppliers, their economic and financial standing, technical or professional ability, the use of a pre-qualifying questionnaire for potential suppliers, and much more. The reconnaissance involves detailed research, some knowledge of the vendor's vendors (who supplies the vendor) and basic background checks on senior staff (at least) and essentially, the past, current, and potential ownership structure of the organisation (who is waiting to buy it or likely acquire its assets). The exercise can be an enlightening experience in both due diligence and strategic investment protection. The discovery of certain countries' depletion of the capability to develop such products may give rise to anxiety in investment, liability, and litigation. Approaches used may include (the purchasing orientated) simple Targeted Account Selling technique and others. e.g., discreetly identifying the vendor's reference customers, if any, and surreptitiously verifying any claims the vendor may have publicly made early on. Normal business services ratings agencies should not be ignored.

13.1. Product selection

With the predicted use of SISs in many devices set to rise significantly, the generic advice is to support major vendors to invest in the mass-production of solid-state, non-programmable SIS systems to reduce the cost to levels enabling wider use, currently limited to major projects only [4].

AI-based software control systems deployed in critical devices such as autonomous vehicles will demand fall-back systems i.e., SIS solid-state systems which cannot be "hacked", those deployed in plant and factories even more so. In following advice by [16], where possible, select only devices where all SIS software usage has been avoided – totally, and select only devices where all levels of BPCS-SIS integration have been avoided. Where use of solid-state, non-programmable SISs is not possible then end-users should strive to deploy SISs that employ only fixed or limited variability programming languages such as ladder logic or function block design.

13.2. Evidence of quality assurance of the software safety case

Numerous disciplines demand a convincing safety case be provided by vendors to include at least three principal elements of: 1) the safety requirements, 2) evidence used as a basis for a safety argument 3) an argument supporting the evidence in meeting the requirements. "*A safety case provides an argument as to why the system is believed to be safe to deploy in its intended operational context,*" where the safety argument binds and relates the safety evidence in a clear, comprehensible, and defensible way to the safety requirements [28]. Though this concept is not pursued here, it is recognised as a potent weapon in exerting pressure on BPCS-SIS vendors to demonstrate quality assurance of systems' claimed safety levels.

Evidence of three aspects of a safety case, including quality management, safety management, and evidence of functional and technical safety, are, for example, requisite elements of the EN 50129 railways safety standard [6]. In the case of motor vehicles, vendors are respectful of decisions made by the US NHTSA as, for example, in the case of TESLA Inc., insisting on the sole use of Artificial Intelligence for passenger safety resulting in having its safety certification revoked [11]. Moreover, the safety assurance of software and its quality assurance is seminally demonstrated in the pattern and anti-pattern approaches pursued by [42], who offers solid, fundamental material on constructing and assuring arguments in developing safety cases of safety-critical software. As the certified safety of critical software systems relies extensively on statistical extrapolation techniques that [8] and [34] address and which cannot be verified by proof testing, the role of statistics in reliably proving software safety is addressed. As a consequence of the sometimes flawed nature of software quality, and in contrast to the historic type-orientated generic certification of vehicles, yet complementing periodic roadworthiness checks, it is proposed that fully autonomous vehicles be subject to similar independent, periodic and rigorous certification focusing on software.

13.3. Evidence of product quality

When eliciting rudimentary information in 2020/2021 from network firewall vendors on their normal business practices, suspicions were aroused about the quality of several such products. The exercise revealed some black sheep using some sharp practices. As evidenced in [4] some SIS vendors are likely to be little different, and hence a critical approach is aspired to. Where software is developed using either limited variability or full variability programming languages such as Structured Control Language (SCL), or the most widely used SIS (or a BPCS) development language "C" then evidence of product quality may be sought in the responses to a range of searching questions.

13.4. Evidence of relevance

Establishing the degree of relevance to a customer's needs may include gathering knowledge of a vendor's capabilities, including the ability to demonstrate a history in a customer's specific industry or the extent of exposure to a customer's specific environment, application, and system of SIS.

13.5. Evidence of quality: software

Exploratory vendor discussions eliciting indicators of product quality may cover software requirements management and issues centred on segregation of design from development teams. Other focuses may be on the selection and use of development tools for certifiable software, or levels of developer experience in similar EUC environments. Closer attention may be given to the design of the software development process itself and its improvement cycle, regression testing consistency, how peer code reviews are conducted, or how static code analysis is performed. Secure coding techniques and related formal safety standards compliance will likely feature highly. Once the software is released, attention may focus on gathering operational field performance data that eases inspection of the maturity of version control infrastructure as well as Management of Change (MOC) procedures.

13.6. Evidence of quality: management

Defence, safety-critical or sensitive business sectors routinely apply screening and structured procurement procedures across departments, and additional full financial disclosure is not rare on both a corporate and personal level. Application of international standards such as BS7858 of staff vetting/screening is applied widely. In financial terms, verifiable caps may be imposed on maximum permitted margins and may be applied as specific contractual terms. Scrutiny may extend to the close comparison and analysis of working practices as designed versus work as actually performed, for example, the managed design of the software life cycle versus how teams interact in reality.

14. Suggestion for commissioning and compliance authorities

Finding and developing credible methods with which to produce ultra-reliable software for use in complex safety and security-related settings is challenging, and what constitutes "credible" must be approached cautiously. Nevertheless, the quantification of the reliability of safety-and security-critical software, as discussed by [8], strongly suggests that the application of probabilistic and statistical methods in that context is unfeasible. They show how it is not possible to demonstrate how research experiments can ensure ultra-reliable software, and hence it seems that, given today's probabilistic and statistical fine focus on proving PFD compliance via SFF and the implied admission that some element of residual risk, both in safety and security terms, will always remain, then it should be desirable to help decision-makers aware of that residual risk. Those risks will range from "known uncertainties" to "unknown uncertainties" with some residual yet permanent element of complete surprise. A significant proposal of the paper is the explicit use of an Uncertainty Assessment Report.

It is [38] who address this concern in detail by drawing attention, with applications in many spheres of transferable usage, to the need for Commissioning and Compliance Authorities to be furnished with "Uncertainty Assessments Reports" – the very opposite of, yet in addition to, a proof of compliance document currently used in industry reporting.

In the absence of perfect knowledge of a BPCS-SIS, an "Uncertainty Assessment" report would highlight both the extent and depth of uncertainty about its safety and security. A minimum of two aspects would be addressed in considerable detail: faithfulness and completeness covering the accurate and full extent of current uncertainty and all significant sources of uncertainty. The two most common pitfalls of uncertainty assessment, namely a) adopting a one-size-fits-all approach representing uncertainty b) ignoring the risk of surprise, are comprehensively covered areas which, with today's approach, appear to be, to some degree, ignored. Classes of the uncertainty of probability of failure on demand (PFD) include completeness uncertainty, model uncertainty, and parameter uncertainty as are detailed by [25] and [27]. Respecting the inherent and comprehensive value of the entire concept of uncertainty assessment reporting, five key risk factors directly applicable to BPCS-SIS safety and security are at the core of the approach. These relate to 1) the nature of the system studied, 2) what is known about it, 3) what is thought to be known about it, 4) what is known about any past genuine surprises it, or similar systems may have exhibited, and 5) if it is being subjected to any new or novel operating conditions [27]. Some or all of these aspects will apply to both new and existing installations.

System complexity, limited system knowledge, overconfidence in oneself and system, past surprise events, and novel operational conditions are addressed. The approach offers the most useful advice on how such factors can be more easily recognised as a warning to the less experienced. Though not explicitly mentioned by (Parker and Risbey) there is strong evidence to suggest their recommendations for certification bodies in other spheres are equally applicable to the certification of BPCS and SIS systems controlling CNI.

15. Conclusion

The inability to exhaustively identify and analyse the failure modes and effects of software-based BPCS-SIS systems results in some quantity of uncertainty of failure modes, some of which will be insecure and dangerous. Realities of data network security concerns of integrated software-based BPCS-SIS systems in substituting software for other software-free forms of protection systems undermine the ability to offer security and functional safety protection or continuity of service or resilience by failing exhaustively to identify all possible failure modes. The concern extends beyond the bounded industrial SIS arena, and into areas where enhanced, complex interdependence between security, data networks, and safety-critical systems resides.

Evidence is presented of quality issues of safety-related embedded software systems development, the inappropriate application of certain software development methods, the neglect of specific coding techniques, and in bowing to commercial pressures, attempts to embellish safety/failure probability calculations in order to promote the sale of allegedly high SIL level devices. In contrast, note is made of exemplary work in constructing software safety case argument assurance. Supporting those concerns are the results of a survey investigating quality dimensions of safety-critical, embedded systems development. Yet more concerning are aspects of hardware and software interaction testing, which the integration of BPCS-SIS demands, yet does not appear to sufficiently receive – if at all. It is conceded, however, that justification for those concerns remains unproven. As a non-ideal comparison, the stellar performance of software quality levels achieved by one team is made, though in the wider-world reality, given how most SIS vendors will have neither the capability nor the resources to achieve that level of quality performance and how some strive to cut quality corners, then serious, if not hazardous, outcomes may be anticipated.

In the absence of perceived strong legislation, yet given the uniquely strategic character of SIS systems, then organisational and procedural suggestions, both overt and covert in nature, for users of SIS systems are proposed ranging from social engineering, intelligence reconnaissance, and vendor profiling techniques and safety case assurance. The international standard IEC 61511 clause 11.2.4, relating to BPCS-SIS segregation, is respectfully questioned. At the very least, here, it is regarded as being open to interpretation such that the promised synergistic effects of Marketing may yet, instead, become severely antagonistic.

In the face of growing complexity-interdependency-uncertainty (CIU), the paper offers an evidence-based approach to kneading vendors of BPCS-SIS integrated systems. This amounts to detailed research into the background of those offering a uniquely strategic component that impacts the very existence of an organisation and its assets, the welfare of its employees, their families, society, and our environment. The paper points to a novel alternative application of Uncertainty Assessment Reporting and promotes its mandatory use in addition to a SIF/SIL verification assessment report – as an indispensable component of a security and safety assessment.

Funding

This work was kindly supported by the Welsh European Funding Office ("WEFO"), an Executive Agency of the Welsh Government, UK and has made available funding through the European Social Fund for East Wales, UK and partly funded by Thales Defence UK [project number: 21,436].

Declaration of Competing Interest

The author whose name is listed immediately below report the following details of affiliation or involvement in an organisation or entity with a financial interest in the subject matter or materials discussed in this manuscript.

References

[1] (ACM)-Association-for-Computing-Machinery, Formally Verified Software in the Real World, (ACM)-Association-for-Computing-Machinery, 2018. Available at: https://cacm.acm.org/magazines/2018/10/231372-formally-verified-software-in-the-real-world/fulltext (Accessed: 1 February 2022).

[2] M.J. Assante, Infrastructure protection in the ancient world, in: Proceedings of the 42nd Annual Hawaii International Conference on System Sciences, HICSS, 2009, https://doi.org/10.1109/HICSS.2009.260.

[3] Australia, G. of (2021) The GAP & IIER-A National Resilience Project.

[4] Barr-Group (2018) Barr Group 2018 Report The State of Embedded Systems Safety.

[5] A.A. Bochman, S. Freeman, Countering Cyber Sabotage: introducing Consequence-driven, Cyber-informed Engineering (CCE), 1st ed, CRC Press, 2021, https://doi.org/10.4324/9780367491161.

[6] British Standards Institute, Railway Applications - Communication, signalling and Processing Systems - Safety related Electronic Systems For Signalling, British Standards Institute, 2007, p. 42.

[7] British Standards Institute (2010) *BS EN61508-1: Functional safety of Electrical/Electronic/Programmable Electronic Safety-Related Systems*. United Kingdom.

[8] R.W. Butler, G.B. Finelli, The infeasibility of experimental quantification of life-critical software reliability, in: Proceedings of the Conference on Software for Citical Systems, SIGSOFT 1991, 1991, pp. 66–76, https://doi.org/10.1145/125083.123054.

[9] David Slatera, Erik Hollnagel, Ralph MacKinnon, A.C.-S. Mark Sujan, P.B. Alistair Ross, A systems analysis of the COVID-19 pandemic response in the United Kingdom–part 1-the overall context, Saf. Sci. (2021).

[10] P. Davies, The automotive challenge: cyber cognisant, legally sustainable best practice in a complex socio-technical system, Thales (2019).

[11] Digital, O. (2021) Tesla loses safety certifications after eliminating radar sensors to bet on Autopilot. Available at: https://olhardigital.com.br/en/2021/05/28/carros-e-tecnologia/tesla-perde-certificacoes-de-seguranca/(Accessed: 6 September 2021).

[12] C. Fishman, Lockheed Martin, They Write the Right Stuff, FastCompany, 1996. Available at: https://www.fastcompany.com/28121/they-write-right-stuff (Accessed: 19 August 2021).

[13] S. Fluchs, Making OT Security Engineering Deserve Its Name A Thought Model and a Data Model, S4 × 22 Dale Peterson, Langenfeld, Germany, 2020.

[14] D.E. Geer Jr, The Future of Security: criticality, Rejectionists, Risk Tolerance, in: Proceedings of the Rocky Mountain Information Security Conference, 2012.

[15] Geer Jr, D.E. (2018) 'A Rubicon', *A Hoover Institution Essay*, (Aegis Series Paper No. 1801), pp. 1–20. Available at: https://www.hoover.org/research/rubicon%0Apapers3://publication/uuid/5BB794C3-0DCB-4E6E-A19F-9E647A9B8A77.

[16] A. Ginter, SCADA Security: What's broken and How to Fix it, Abterra Technologies Inc, 2016.

[17] A. Ginter, Secure Operations Technology, Abterra Technologies Inc, 2018.

[18] P. Gruhn, Safety Instrumented Systems A Life-Cycle Approach, International Society of Automation (ISA), 2018.

[19] E. Hollnagel, FRAM: the functional resonance analysis method: modelling complex socio-technical systems. FRAM: The Functional Resonance Analysis Method: Modelling Complex Socio-technical Systems, CRC Press, 2012, https://doi.org/10.3357/asem.3712.2013.

[20] E. Hollnagel, Managing for security, Adv. Sci. Technol. Sec. Appl. (2021) 43–53, https://doi.org/10.1007/978-3-030-42523-4_4.

[21] E. Hollnagel, D.D. Woods, N. Leveson, Resilience engineering: concepts and precepts, Resilience Eng. (2012), https://doi.org/10.1136/qshc.2006.018390.

[22] ISA, ISA Guidance on Compliance Integrated and Separate? The 61508 Association, 2017.

[23] W. Iversen, The Great Safety Debate, Automation World, 2007. Available at, https://www.automationworld.com/products/control/article/13301802/the-great-safety-debate (Accessed: 18 March 2021).

[24] Janbu, A.F. (2009) Treatment of Uncertainties in Reliability Assessment of Safety Instrumented Systems.

[25] H. Jin, M.A. Lundteigen, M. Rausand, Uncertainty assessment of reliability estimates for safety-instrumented systems, in: Proceedings of the Institution of Mechanical Engineers, Part O: Journal of Risk and Reliability 226, 2012, pp. 646–655, https://doi.org/10.1177/1748006x12462780.

[26] H. Jin, A. Summers, Dependent, independent, and pseudo-independent protection layers in risk analysis, Process Saf. Prog. (2016), https://doi.org/10.1002/prs.11796.

[27] I.L. Johansen, M. Rausand, Defining complexity for risk assessment of sociotechnical systems: a conceptual framework, in: Proceedings of the Institution of Mechanical Engineers, Part O: Journal of Risk and Reliability 228, 2014, pp. 272–290, https://doi.org/10.1177/1748006x13517378.

[28] T.P. Kelly, Arguing safety – a systematic approach to managing safety cases, Science (1998). Available at: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.58.8163&rep=rep1&type=pdf.

[29] A. Khurana, Managing complex production processes, Sloane Manage. Rev. 40 (2) (1999) 85–97 (Winter 1999).

[30] D. Krakauer, Complexity & Stupidity, Sam Harris (2016). Available at, https://samharris.org/complexity-stupidity/ (Accessed: 18 March 2021).

[31] S. Lenz, Vulnerabilitaet Kritischer Infrastrukturen (German), Vulnerability of Critical Infrastructures, German Federal Ministry of Population Protection and Catastrophe Aid (Bundesamt fuer Bevoelkerungsschutz und Katastrophenhilfe), Berlin, 2009.

[32] N.G. Leveson, Safeware System Safety and Computers, 4th edn, Addison Wesley Longman Inc, 2000.

[33] Leveson, N.G. (2017) 'Engineering a Safer and More Secure World', Presentation.

[34] J. McDermid, T. Kelly, Software in safety critical systems: achievement and prediction, Nuclear Future 2 (3) (2006) 140–145, https://doi.org/10.1680/nuen.2006.2.3.140.

[35] MoD, U.K. (2021) Defence and Security Public Contract Regulations (DSPCR) Chapter 15. Available at: https://www.gov.uk/government/publications/the-european-union-defence-and-security-public-contracts-regulations-dspcr-2011/dspcr-chapter-15-supplier-selection (Accessed: 1 September 2021).

[36] National Aeronautics and Space Administration, *NASA Software Safety Guidebook*. NASA TECHN, *Nasa Technical Standard*, NASA Techn., 2004. Available at: http://www.hq.nasa.gov/office/codeq/doctree/871913.pdf.

[37] Osepchuk, B. (2020) 'Safety-Critical Software: 15 things every developer should know - Small Business Programming'. Available at: https://smallbusinessprogramming.com/safety-critical-software-15-things-every-developer-should-know/ (Accessed: 20 August 2021).

[38] W.S. Parker, J.S. Risbey, False precision, surprise and improved uncertainty assessment, Philos. Trans. Royal Soc. A 373 (2055) (2015), https://doi.org/10.1098/rsta.2014.0453.

[39] J. Rasmussen, I. Svedung, Proactive Risk Management in a Dynamic Society, Swedish Rescue Services, Karlstad, 2000. http://rib.msb.se/Filer/pdf%5C16252.pdf.

[40] J.P. Scoblic, Strategic Foresight As Dynamic capability: A new Lens On Knightian uncertainty Strategic Foresight As Dynamic Capability, Harvard Business School. #20-093, United States of America, 2020.

[41] SPRIN-D (2021) Federal German agency for disruptive innovation. Available at: https://www.sprind.org/en/ (Accessed: 26 October 2021).

[42] R.A. Weaver, The Safety of Software – Constructing and Assuring Arguments, University of York, 2003.