Three laws good: Technology is a dangerous master

Dr Malcolm James Cook[a], Terry Simpson[a], David Garrett[a] & Martin Thody[a1]

[a] *BAE SYSTEMS (Maritime), Barrow-in-Furness, Cumbria, LA14 1AF*

Corresponding Author: e-mail:Malcolm.Cook2@baesystems.com

"Technology is a useful servant but a dangerous master" Christian Lous Lange

Synopsis

A philosophy of technology use has developed in many safety-critical industries that is based upon the view that human operators are feckless and unreliable system operators, so wherever possible should not be trusted to execute safety-critical tasks. The implicit view of automation is that it invariably improves system performance and increases reliability. After many decades or even centuries of machine and automation development human error remains one of the dominant features in failures of modern systems. The drive towards introducing automation has claimed a larger performance envelope, lower operating costs with fewer people, less risk of hazard realisation, and a more economical path in development. One of the aims of introducing automation is to introduce higher reliability in the belief that this implicitly brings with it increases in safety. As Leveson (2011) points out high reliability can be misleading because interactions between elements that are working as expected may trigger the system failure because of transverse consequences. The propagation of the view that human operators are the weakest operational link and the pervasive myths about the reliability of automated solutions, which affords automation the easier scenarios of task execution, need to be re-visited (Cook, Thody and Garrett, 2017). This should ensure that the best capability and optimal safety case is developed for future systems based upon operator and system in synergy. This may be especially true if the claims for automation are treated more aggressively in terms of liability.

1 Introduction

It is proposed that approaches to design shaped by safety engineering thinking may not be effectively reconciled with issues of usability and operability, safety may tend to dominate design thinking by demanding the greater share of available resource. This bias is based upon a priori logic which seems credible, plausible and justifiable in manner that is simple to argue for safety engineering but proves more problematic for human factors issues to influence. The arguments in favour of safer engineering are tied together with claims about system reliability, the use of automation and now involves embedded software management of functions, the real-world evidence suggests that reliability and safety are overlapping but distinct system qualities that are easily confused in the arguments about safety.

SIL(Safety Integrity Level) ratings are a method of avoiding rare but potentially catastrophic accident in systems (Beugin, Renaux and Cauffriez, 2007). It has been argued that current approaches do not assist developers or assessors of safety critical software (Fenton and Neil, 1998). However, safety analysts advocating SIL rated systems and adherence to IEC 61508, can create untenable costs in verifying complex systems that grow exponentially, with the combinatorial explosion of real-world challenges. Functional Safety based approaches refer to IEC 61508 and related standards like IEC 61511, which are intended to assist in applying SIL levels to systems under development (Smith and Simpson, 2004, 2016). In process industries (Janainian, 2017), SIL assessment has been used to identify safety instrumented functions (SIFs) which protect the systems against intrinsic hazards. The quantitative approach to risk management has limits and it is accepted that part of the limits relate to the cost of developing the higher level integrity systems, at SIl 3 or 4 (Kemp, 2016). However, in many cases system designers opt for a more manageable closed system approach, excluding certain inconvenient and costly variations in environmental and design challenges. This process of "approximation" of a safety case based on partially automated systems generates a process of residualization, whereby the more difficult and unpredictable safety challenges are left for the human operator, when the automation/software fails.

In allocating functions between human operator and machine intelligence designs may push more problematic tasks onto the human who can only be associated with very weak claims for performance success on demand, via techniques like Human Reliability Analysis (HRA). Case studies are used to explain the consequence of current approaches to function allocation. A better way of exploring requirements in the early stages of design is proposed by the authors (Jacobson and Ng, 2004), which has been linked to scenario-based modelling (Mcleod, 2008) to enhance human factors integration, that should reduce the likelihood of

[1] No biographies are provided for these authors, for security reasons.

http://doi.org/10.24868/issn.2631-8741.2018.016

residualisation, based on the principle of encapsulation taken from software development. Late discovery of human factors requirements frequently results in designs that are functional but not usable. The consequence of this proposed approach should be enabling technology insertions as a servant, and avoiding the human operator becoming both a slave and a scapegoat for poor design.

Automation has continued to develop and evolve, necessitating experts in human factors to extend the capabilities of their taxonomies (Cabrall, Sheridan, Prevot, de Winter & Happee, 2018) but even this may not protect system design from becoming problematic. It may be that evaluating the design in context remains the best way of guaranteeing the correctness of the design (Feigh and Pritchett, 2014; Pritchett, Kim and Feigh, 2014). The endpoint for the introduction of automation and assurance of safe operation may be the single lone operator in a system where reducing the operators from three to two was radical, and now automation may be used to bring that down to a single operator (Harris, Stanton and Starr, 2015) or an absent operator in some cases, with autonomous systems. In estimating the value of an automated function replacing a human operator the implicit rewards appear very attractive, in a superficial analysis. Introducing automation can be associated with doing things faster, executing tasks more accurately when they are repetitive, and processing a wide array of data, but when automation fails in this expanded performance envelope the human operators is faced with an impossible task. Changing capability always involves additional risk and the argument put forward in the analysis is that the management and even the assessment of that risk have a cost which needs to be absorbed into the overall project budget but it is rarely addressed. Reversionary modes of operation are more about safety than capability, and the reversionary modes receive attention proportionate to their perceived need. This may part of the reason why human error persists as a significant contribution in more advanced systems because it is simply not addressed in sufficient detail, with enough resource.

Software and hardware combinations can be awarded very high performance on demand estimates, providing very reliable performance and software intensive systems are pervasive, employed in a large number of safety critical systems (Firesmith, 2004). Once programmed software and hardware systems have error producing conditions which stimulate automation failure and these are assumed to be known in as far as is reasonably practicable. Spectacular failures attributed to software, such as the Ariane V spacecraft launch which had to be terminated (Dowson, 1997; Lions, 1996; Nuseibeh, 1997) indicate that errors do emerge in use which have not been trapped in development. Although in the case of Ariane V, it was the re-use of software that had previously worked in the wrong context, with a new launch vehicle with different ballistic characteristics that gave rise to the failure[2]. This attempted re-use which produced a costly error and consequences is perhaps an indication of the pressure from costs to avoid re-inventing the proverbial wheel. As will be noted later cost becomes a major driver for high integrity software systems in safety critical applications because of the costs of verification and validation. The importance of V&V is captured in the challenge,

> "If you don't test your software, how do you know it works" Pham (2000) Software Reliability.

Some have suggested that exhaustive testing of computer software is potentially intractable (Kuhn, Wallace and Gallo, 2004). The plethora of books on software reliability methods, software reliability testing and software safety reliability underscore the importance of software failures in driving the scrutiny applied to software especially in safety critical systems. What is perhaps surprising is that very few of the developers involved in such work have considered the history, as Leveson (2013) has, to interpret the way that the business of producing safety critical software has evolved and the nature of the problems have changed.

It is argued that one needs to consider under what conditions automation is applied to system management and operation before one accepts the value of automating processes, implicitly excluding the human operator. Leveson (2013) has suggested that "empirical evidence seems to validate the commonly stated hypothesis that the majority of safety problems arise from software requirements and not coding errors". As Firesmith (2004) suggests rare combinations of circumstance and hazard can result in system failures which were not foreseen in development. Rarely are requirements statements complete, comprehensive or universal in their application, so there are often cases which are not examined. If one accepts that safety systems are about methods for applying constraints to system operation (Marais, Dulac and Leveson, 2004) then incomplete requirements are represented as gaps in safety systems or they may be realised as errors in the software, like that

[2] Some have claimed that large-scale IT failures will become more than an expensive inconvenience (Charette, 2005), whatever the ultimate causes recent events suggest that software–based systems are vulnerable and very rarely immune to intrinsic errors. Often it is a mismatch between the necessary design and the expressed requirement which leads to failure.

which occurred in Ariane V software which was a floating point to integer conversion, as a result of larger horizontal forces in the V compared to the IV rocket. Failure of software projects is not new (Flowers, 1996) and failures to consistently deliver required functionality are not rare either (Hughes & Dwivedi, 2015). Some of the same issues that contribute to failure of projects in some cases are latent in delivered designs, where they later may appear as a part of an unexpected accident sequence.

This paper considers the use of automation in flight control systems on moving platforms and as such it can be applied to planes, spacecraft, and maritime platforms. As Leveson (2013) has suggested for spacecraft, it may prove difficult to eliminate errors in flight control software and for that reason a human may be present to essentially trap the errors that the software makes. However, there are reasons to believe that there is a significant amount of optimism in this and the ability of the human operator to provide an adequate recovery response. The operator is often given less effective situation awareness with automated systems and experiences surprises when their expectations are not met by software-based systems.

2 The Law of Big Numbers

Automation is often, but not always, designed for use on fully operational systems where there are no faulted systems in the control and service pathway. Often, where systems can be faulty, diverse and independent function is used to cascade through functional systems to an available and operational system. Thus, automation is often configured for the required functional availability. The need for diverse and independent functional abilities is rarely associated with the necessities of automation and is more often justified in terms of providing barriers to catastrophic system loss.

The reason why this is more likely is because of the complexity of modern systems. Equally it is the complexity of developing automated solutions for such systems that presents a problem. Consider the Apollo 13 mission, where the Lunar Excursion Module (LEM) became the life raft that brought Jim Lovell, Jack Swigert and Fred Haise back from the abortive landing. The ingenuity of the crew and those at Mission Control re-engineered a solution from what they had available. The United Airlines Flight 232, Souix City crash, is another example of a failure where the systems were not configured to land the plane after loss of control of many flight control surfaces. Even if automated systems could be creative it would difficult to verify and validate that ability.

If multi-purpose functionality is required for safety-critical systems and that functionality must be maintained in the face of partial functional unavailability, loss of systems, the potential scenarios of use are a product of combinatorial explosion, where the costs of validation and verification increase exponentially. Consider for example steering control software which uses multiple planes to manage the manoeuvring of the platform under control, the standard mode of operation may rely upon all control surfaces being operable. Or, it is possible that control software can be used when a single control surface has failed. As the number of control surfaces that could fail increase it adds to costs of the verification and validation process to demonstrate that the software control is demonstrable. In many systems which are automated the easiest method of cost reduction is to transfer control to the human operator, often with the view that the operator controlled the vehicle before and no problems should arise.

However, more human operators probably managed the task in legacy systems and as a result the manning reductions afforded by introducing automation, there are fewer operators, which results in overload. In addition, the operator who was previously engaged in using the system on a regular basis is no longer involved to the same degree or with the same frequency, so they have skill fade to contend with. Finally, the task that is offered to the operator is often the more difficult one when the challenge is too complex for the automation to adapt to. These observations are not new but it is surprising to find them constantly being re-discovered in accident sequences and by engineers developing complex control systems for safety critical tasks.

Concerns about the impact of introducing automated systems have a long history in the aviation sector and the management of highly dynamic and non-routine situations were among the first characteristics identified as problematic (Sarter and Woods, 1997). It was clear that managing unusual situations revealed other latent problems with the mental models of the operator and with low system observability (Sarter and Woods, 1997). The operator's mental model of the situation starts to drift because of the low observability of the commanded processes and when a poor understanding of system operation is layered on top of this one has the potential for automation surprises, where the expectations of the operator are not met by the behaviour of the system. Indeed, automation and its claimed simplicity of use may be used to reduce training further, exacerbating the predicament faced by the operator in those safety critical failures of the automation.

Thus, even though automation is intended to reduce the operator's workload and the need for situation awareness, automation has been shown to impose high attentional demands on the operator and increase the level of knowledge demands on the operators. Operators paradoxically need to sustain high levels of attention and cognitive resource usage because they may be required to intervene without foreknowledge or indication of the failure. As Leveson (2013) has pointed out it has proved almost impossible to exclude the possibility of software errors in flight control systems, so one needs to plan for the possibility of transitioning from automated control to some form of manual control. The introduction of automation has through this transitional process, between automation and manual control, created a new opportunity for human error, which ironically it was intended to eliminate.

The human operator's contribution can seem to be minimised by the argument that the automation manages the greater percentage of the operational use, captured as a time at risk argument. This time at risk argument nominally equates the greater proportion of operational time under automation with the much smaller time when the human operator must contribute under challenging conditions after unexpected failures. The failure to correctly assess the impact of the remainder and to address usability issues results in marginalisation of operator's contribution, with not unexpected consequences.

3 Human Error

Human errors can be estimated and a number of characteristics are used to predict the frequency of error. It is expected that when human operators are required to respond quickly in an unprepared and unplanned way, their performance will be poorer and more error prone. Faster responses are usually associated with higher error rates and this is described as a speed-accuracy trade-off. When automation fails without warning, requiring a transition to manual control it is almost the perfect stimulus for increased error rates.

Models of human cognition are predicated on the idea that there is a limited working memory capacity and integrating and analysing a lot of data is a significant challenge to the human operator. Automated systems are often used in situations in which they integrate a lot of information and develop a response to that, so that when the automation fails the equivalent task relies upon the human operator. Given the limited working memory of the human operator a sudden demand to integrate and analyse large amounts of data overwhelms the operator, creating opportunities for influential human errors leading to accident sequences.

It is impossible to avoid the issue of behaviour and information processing management motivation, in the presence of automation. As noted above automation is used in situations where the information is changing at a reasonable pace and the human operator would need to expend a significant amount of effort to follow and supervise the automation. If the operator actually fulfils the same role as the automation, then one has to consider why so much money and design effort has been devoted to the automation. In practice, the human operator is often given additional tasks to complete when automation exists because the operator is perceived as less engaged in the task which has been automated. Pejorative terms such as dependence and reliance have often been applied to the operator's tendency to be dependent on automated system management. Whether one views the operator's tendency to depend on automation as complacency or not is a moot point, it happens and as a result in many automated systems the operator has reduced situation awareness.

It is interesting to hear how frequently professional groups deny the influence of automation on the effort devoted by operators to monitoring automatic tasks. There is often an insistence that professionalism, training and culture can overcome energetic management of task demands. There is no doubt that operators can in the short-term demonstrate significant capability but the argument in terms of safety is about the longer term. Once familiarity with the system operation has increased and the operator trusts the system to manage the process with minimal errors, the natural tendency is to relax. The difficulty is that automation is traded for operators on one basis and the reality of automation influence and capability is somewhat different.

4 The Best of the Worst

Automation has frequently been promoted on the basis that it will reduce human error by eliminating the human, as much as possible, and provide regular dependable system operation. In practice, automation is largely used to control the easier elements of system operation when the required system elements are working. Thus, the more difficult and problematic system failures are devolved to the human operator who is expected to monitor the automation in case is enters a state where it cannot resolve the problems. Not only is the operator expected to respond to these exceptions they are expected to be ready to do this quickly and often on a right first time basis. As is noted above this expectation of success is frequently falsely held and often meets with a disappointing outcome. The operators are often disengaged from the process and their situation awareness is poor, so they need to use time post-event to re-build it. The demands from the tasks, for which automation is frequently used, are often at the limit of what human operators can cope with. Thus, it is at these unexpected transitions between automated system management and human operator control that errors and failures to regain control will occur. Consider how many times in a designed system, can the operator return a difficult task to the automation or ultimately where the safety critical systems execute a final and irreversible action. Leveson (2017) asserts that engineers have ignored humans in hazard analysis of systems or treated them as if they fail randomly like hardware but, as she notes, human operators are affected by context and behaviour-shaping mechanisms, which make human error more systematic than some engineering models imply.

The models applied to design and to development of system requirements are rarely complete and comprehensive, so unsurprisingly the systems are not universal in application or capability. In this analysis there are echoes of the views expressed by Firesmith (2004) and Leveson (2013), in that the requirements are rarely comprehensive and as a consequence there are always difficult cases which remain for the human operator to manage in system operation. It has been this metaphorical Achilles Heel that has proved to be a significant obstacle to the application of automation in many moving platforms, preventing the affordance of increased autonomy and authority desired by many system designers promoting automation.

5 Broken Promises

" A mythology has arisen about the Shuttle software with claims being made about it being “perfect software” and “bug-free” or having “zero-defects,” Leveson (2013).

The promise of automation was to provide a larger performance envelope because the faster information processing speed of the automation (hardware and software) would allow expansion of the performance envelope towards areas where human operators could not previously manage the hazard realisation and protection. Interposing the automation between the human operator and the process appears to open up a possibility. It also opens up a risk, when failure occurs anywhere within the stretch zone where humans can no longer step-in. Potential failure within the stretch zone reduces the value of the automated solution within the safety case and may compromise its value because the use of automation in providing that is shrunk within a time at risk argument.

Lower manning may be afforded by software and automated solutions. Lower manning may appear attractive but lower manning may increase the likelihood of operators being overwhelmed in situations of high demand. Concurrent failures in systems, although unlikely, are possible. The reliance on automation when systems are functioning normally de-skills the operators and concurrent failures may overwhelm the operator's inherent cognitive capacity and acquired ability to cope. Indeed, the need for fast, right first time responses in the event of automation failures are likely to be magnified by reduced numbers of operators after the introduction of automation. Consider the change to automation, with the same number of concurrent processes, a smaller crew complement and a similar frequency of significant and complex errors, for the probabilistic reasons identified above. It is possible that the co-occurrence of demands on the now reduced crew will potentially exceed their capacity, noting that frequency calculations usually consider frequency of failure event occurrence and often assume independence of events, with the exception of common cause and single points of failure.

Although automated solutions are likely to reduce the risk of hazard realisation in simple incidents it is likely that for the foreseeable future that automation will not be applied to faulted conditions because the verifying the automation for variant models of control may prove too costly. Thus, while automation reduces the opportunities for time to practice skills to the highest level in low jeopardy circumstances it leaves the most difficult situations for human operator to manage, with a lower level of preparation.

Whilst the recurrent costs of maintaining a crew complement at the required skill level is expensive the up-front cost of verifying and validating automation is equally expensive and the more diverse the events it is designed to cope with, the greater the cost. Whilst there is still a requirements gap, between the ideal automated system, and the delivered system it is arguable that crews need to be more skilled and greater effort is required to maintain them as proficient, which begs the question. Where are the economic and safety benefits of automated safety critical systems?

6 Pruning the Heart of the Fault Tree

A time at risk argument can be applied to the evidence above. Consider a system operating in two modes, a simple mode 90% of the time, which has a risk rate of failure 1% of the time. The second mode is more complex and occurs when the automation cannot be applied, occurring 10% of the time with a risk rate of failure of 10%, as a result of the reliance on human operators. Multiplying these figures, for simple and complex modes, gives a frequency of absolute failure of 0.9% and 1%. One can see that the introduction of automation has only halved the potential accident rate, assuming the automation works in the 90% of the cases that are simple. Automation has eliminated accident occurrences but it may actually have increased the likelihood of failures in the more complex cases where lack of opportunity to gain experience (train), speed of the response required, impact on situation awareness and the other factors simply transfer the error rates onto the transitions between automation and operator intervention, they become human error. It is rare to see this analysis within a safety case and perhaps it is something that should be considered.

The changes brought about by some designs which use automation can be more illusory than real even when they appeal to the standard function allocation methodologies. Sheridan (2000) has suggested that computers, automation and robotics offer greater capability but they do not come without risks, such as greater system complexity and designer bewilderment, which might be perceived as Leveson's (2013) requirements issue. Whatever the interpretation, the fundamental relationship is between risk and capability, expanding capability amplifies or increases risk, and that may be what is captured in Sheridan's assertion about designer bewilderment. Sheridan's (2000) assertion that intermediate complexity is best for automation implementation appears to be almost superstitious, in that it suggests that retaining human involvement is important in assuring function but equally lowering the bar on capability ensures that human operators can recover when software fails, the latter perhaps representing an optimistic point of view. There have been many accidents where the unexpected transition back from automatic to human operation has been less than optimal. For example, the human operator's assumption that controlled operation under automatic systems will present no problems when the automation is disengaged has been proven to be challenging. Operators have missed a failure masked by automation function and unreported, a failure which is revealed when the automation is disengaged. This again represents a use case where the automation has not been programmed to detect errors and report them, and this kind of situation is fundamentally the reason why automation is rarely a good team player (Klein, Woods, Bradshaw, Hoffman, and Feltovich, 2004).

Examination of the interaction issues has demonstrated that automation rarely supports effective joint human-machine interaction even though human operators appear to respond to systems in a manner that suggests they apply expectations which are more compatible with human-human interaction (Joe et. al., 2014). This hypothesis about human system interaction is supported theoretically by other established groups (Reeves and Nass, 1996).

In conclusion, it seems likely that future application of technology faces many challenges, many of which are resonant with the current academic literature. First, the verification and validation of software correctness and reliability need to be established, and even then one may need to accept that there are failure modes that require the human operator to be partly engaged in the task. Second, the graceful degradation of automated support needs consideration to develop intermediate modes of operation that do not support the same level of functionality, but can support return of platforms to base. The improvements in software requirements engineering, and software reliability may never completely exclude the possibility of failures but they help to make them more manageable. As suggested in the introduction evaluating design in the appropriate range of operational contexts is the correct response to Pham's challenge, to know the software works.

Consideration of this inherent cost at the start of the software development may make estimates of likely cost, for development, more realistic. The operating costs for safety critical systems, which automation is intended to reduce, may actually be much smaller than the cost of developing and assuring the software at the heart of the new system design. As noted above, the realistic appreciation of the capability afforded by such advanced automation should be used to temper the expectations fostered by new designs. Failure in software projects is always a real possibility (Charette, 2005) and one of the best defences is planning at design onset (Leveson, 2013).

It can be argued that a significant amount of effort has been made in terms of system engineering to demonstrate how the introduction of automation will reduce the failure rates and safety critical events, for the easier accident sequences which start from a known point where all systems are available. Numerically, the contribution to safety from these simple cases, can be significant but not extraordinary, in addition, it may take effort away from effectively engaging the human operator in the management of the system operation. Human operators are still the critical adaptive element in systems, and by neglecting them in the design process because they have minimum PFD of 1E-2 in time and safety critical tasks, designers are potentially creating opportunities for human error in accident sequences, particularly at transitions from automation to operator control. It would be better if designers focussed on a joint model of system operation, examining the human operator's requirements and those of the system simultaneously (Hollnagel and Bye, 2000). This joint modelling approach is aligned to functional models with a consideration of how the operator and the system provide timely inputs to prevent hazard realisation.

Concerns about the way in which design teams allocate functions are not new (Hancock and Scallen, 1996; Fuld, 1993) and they run to the core of the concerns expressed above. The naivety of the function allocation process is probably well supported by Leveson's (2017) view that " Safety is an emergent property. Looking only at one part of a complex system, it is not possible to tell whether an accident will occur. The property of “safety” only makes sense when all the interactions among the system components are considered together. "

In particular the somewhat parochial view held by a body of engineers that "I am a human, and (perhaps) also an engineer, therefore I am a human factors engineer." Is a passive obstruction to progress because it dismissively degrades the significance of professional accreditation awarded to authentic human factor engineers. Indeed, one might argue that if the knowledge were so widely and easily available we would not continue to repeat the mistakes of the past. There are those who have suggested that the application of the function allocation rules is justified because Fitt's List fulfils important criteria associated with the philosophy of science (de Winter and Dodou, 2011). However, implicit in the allocation of function is the encapsulation of responsibility and authority, as historical evidence demonstrates confidence in the abilities associated with automation are not as robust as Fitt's List would assume or imply. Machines can respond quickly but there lack of context sensitivity means that sometimes they may respond incorrectly. Machines can execute concurrent tasks with large bodies of data that are partly integrated using a given rule set but they demonstrate no flexibility or meta-processing abilities, so that when they fail, they return a complex and challenging task to the human operator. Fundamentally, the boundary between automation (machine operation) and human operation is more permeable than Fitt's List assumes or requires.

7 Conclusion

The failure to understand the way that the task changes, in the presence of automation, and the manner in which the human operator engages at the critical times (in the event of software failure) is the crux of the current problems with automation. This interface is largely designed or determined by those without the required training or understanding. The importance of establishing the role of humans and automation, and the exploration of the task-related requirements have been identified as a critical part of requirements development, which need to be addressed as early as possible within projects where human-automation is a primary element of delivering functionality (Feigh and Pritchett, 2014). Pritchett, Kim and Feigh (2014a, 2014b) stressed the importance of detailed analyses where the outcomes were a result of tightly coupled interactions between human and automation, importantly both static analysis and dynamic simulations were used, indirectly responding to the criticism of static modelling of function allocation was ineffective. Dynamic analysis normally assumes that the barrier in task execution, management and functional availability is more fluid.

The concerns about static function allocation, implied by the MABA-MABA approach, are echoed by Dekker and Woods (2002) who suggest that automation is almost always justified on the basis of its presumed benefits for system performance. This nudges the system designers and design teams towards the implicit view that automation is always better than human operator performance, even though this may not be true in all cases, as indicated above. Automation rarely deals well with all exceptional and unexpected events (Cook, Thody and Garrett, 2017), even though humans may not fare better they are perceived as more responsible for the failure to manage unexpected events, human error persists in accident reports where other causes are reduced. Human error even persists where the design of the automation or the interface may have induced the error, consider Leveson's (2013) observations about interface limitations resulting from technology limitations. The improvements in system performance as a result of automation, which may be largely delineated within a limited region, may be used to expand the performance envelope to a point where human operators could not accomplish the required recovery in the event of automation failure. And yet, these failures to step-in as required may be recorded as human errors and failures.

There is a critical legal issue. If those developing systems make the claim that their automation can manage all circumstances and contexts they may be obliged to prove it, the exhaustive proof which has been represented as almost impossible by Leveson (2013) and very expensive to the point of unaffordable. In addition, any failure events which can be traced to the automation may then associate liability with the responsible company. Before making a claim and argument about automated systems it has become critical that this is supported by evidence, and not opinion. As Leveson (2013) suggests, accidents with the same or similar causes keep occurring which indicates that practitioners are failing to learn from experience, as they should. Or, perhaps the practitioners who have learned are not the same practitioners who make critical design decisions. Or, as Leveson (2013) suggests the assumptions and techniques applied to accident analysis are of limited utility. The assumptions may be implicitly held and applied by practitioners, perpetuating the same erroneous views about accident causation. Almost 16 years after concerns about the impact of technology were identified (Leveson, 2002) and a new accident model was developed it seems that we have potentially missed a solution as to the continuing cause of human error. Space exploration and space craft probably represent the most transparent and clear demonstration of the limits to software engineered solutions, in that unmanned missions have no human component and yet the rates of failure are still relatively high (Leveson, 2013). Arguing that the human designer is the cause of failures, like some ghost in the machine, is in some sense undermining the utility of automation because systems do not generate themselves, and are unlikely to do so for the foreseeable future.

Acknowledgements

The development of this work was supported by BAE Systems (Maritime) but the views expressed in this paper are those of the authors and not of any other party, the authors are grateful for the support afforded. However, the authors take full responsibility for the content of the paper and the views expressed. This paper has been security cleared and is deemed available for general publication in the open literature.

References

Beugin, J., Renaux, D., & Cauffriez, L. (2007). A SIL quantification approach based on an operating situation model for safety evaluation in complex guided transportation systems. Reliability Engineering & System Safety, 92(12), 1686-1700.

Cabrall, C. D., Sheridan, T. B., Prevot, T., de Winter, J. C., & Happee, R. (2018, January). The 4D LINT Model of Function Allocation: Spatial-Temporal Arrangement and Levels of Automation. In International Conference on Intelligent Human Systems Integration (pp. 29-34). Springer, Cham.

Charette, R. N. (2005). Why software fails [software failure]. IEEE Spectrum, 42(9), 42-49.

Cook, M., Thody, M., & Garrett, D. (2017). I didn't see that coming: The perils of underwater automation. Warship 2017: Naval Submarines & UUVs, 14-15 June 2017, Bath, UK.

Dekker, S. W., & Woods, D. D. (2002). MABA-MABA or abracadabra? Progress on human–automation co-ordination. Cognition, Technology & Work, 4(4), 240-244.

de Winter, J. C., & Dodou, D. (2014). Why the Fitts list has persisted throughout the history of function allocation. Cognition, Technology & Work, 16(1), 1-11.

Dowson, M. (1997). The Ariane 5 software failure. ACM SIGSOFT Software Engineering Notes, 22(2), 84.

Feigh, K. M., & Pritchett, A. R. (2014). Requirements for effective function allocation: A critical review. Journal of Cognitive Engineering and Decision Making, 8(1), 23-32.

Firesmith, D. (2004). Engineering safety requirements, safety constraints, and safety-critical requirements. Journal of Object technology, 3(3), 27-42.

Fuld, R. B. (1993). The fiction of function allocation. Ergonomics in Design, 1(1), 20-24.

Flowers, S. (1996). Software failure: Management failure. J. Wiley and Sons.

Hancock, P. A., & Scallen, S. F. (1996). The future of function allocation. Ergonomics in design, 4(4), 24-29.

Harris, D., Stanton, N. A., & Starr, A. (2015). Spot the difference: Operational event sequence diagrams as a formal method for work allocation in the development of single-pilot operations for commercial aircraft. Ergonomics, 58(11), 1773-1791.

Hughes, D.L., & Dwivedi, V.K. (2015). Success and failure of IS/IT projects: A state of the art analysis and future directions. Springer.

Jacobson, I., & Ng, P. W. (2004). Aspect-oriented software development with use cases (Addison-Wesley object technology series). Addison-Wesley Professional.

Jahanian, H. (2017). Optimization, a rational approach to SIL determination. Process Safety and Environmental Protection, 109, 452-464.

Joe, J. C., O'Hara, J., Medema, H. D., & Oxstrand, J. H. (2014). Identifying requirements for effective human-automation teamwork (No. INL/CON-14-31340). Idaho National Laboratory (INL).

Kemp, R. (2016). Quantitative risk management and its limits. In A. Burgess, A. Alemanno & J.O. Zinn (Eds.) Routledge Handbook of Risk Studies, p.p. 164-178.

Klien, G., Woods, D. D., Bradshaw, J. M., Hoffman, R. R., & Feltovich, P. J. (2004). Ten challenges for making automation a" team player" in joint human-agent activity. IEEE Intelligent Systems, 19(6), 91-95.

Kuhn, D. R., Wallace, D. R., & Gallo, A. M. (2004). Software fault interactions and implications for software testing. IEEE transactions on software engineering, 30(6), 418-421.

Leveson, N.G. (2004). A new accident model for engineering safer systems. Safety science, 42(4), 237-270.

Leveson, N.G. (2009). Software challenges In achieving space safety. Journal of the British Interplanetary Society 62, July/August (2009).

Leveson, N. G. (2011). Applying systems thinking to analyze and learn from events. Safety Science, 49(1), 55-64.

Leveson, N.G. (2013). Software and the Challenge of Flight Control. In, R. Launius, J Craig & J. Krige (Eds.) Space shuttle legacy: How we did it/What we learned. American Institute of Aeronautics & Astronautics.

Leveson, N. G. (2017). Rasmussen's legacy: A paradigm change in engineering for safety. Applied ergonomics, 59, 581-591.

Lions, J.L. (1996) http://www.di.unito.it/~damiani/ariane5rep.html Accessed 3/6/18.

Marais, K., Dulac, N., & Leveson, N. (2004, March). Beyond normal accidents and high reliability organizations: The need for an alternative approach to safety in complex systems. In Engineering Systems Division Symposium (pp. 1-16). MIT Cambridge, MA.

MacLeod, I. S. (2008). Scenario-based requirements capture for human factors integration. *Cognition, Technology & Work*, *10*(3), 191-198.

Nuseibeh, B. (1997). Ariane 5: who dunnit?. IEEE Software, 14(3), 15.

Pham, H. (2000). Software reliability. Singapore: Springer-Verlag

Pritchett, A. R., Kim, S. Y., & Feigh, K. M. (2014a). Measuring human-automation function allocation. Journal of Cognitive Engineering and Decision Making, 8(1), 52-77.

Pritchett, A. R., Kim, S. Y., & Feigh, K. M. (2014b). Modeling human–automation function allocation. Journal of Cognitive Engineering and Decision Making, 8(1), 33-51.

Reeves, B., & Nass, C. I. (1996). The media equation: How people treat computers, television, and new media like real people and places. Cambridge university press.

Sheridan, T. B. (2000). Function allocation: algorithm, alchemy or apostasy? International Journal of Human-Computer Studies, 52(2), 203-216.

Smith, D. J., & Simpson, K. G. (2004). Functional Safety: A straightforward guide to applying IEC 61508 and related standards. Routledge.

Smith, D. J., & Simpson, K. G. (2016). The Safety Critical Systems Handbook: A Straightforward Guide to Functional Safety: IEC 61508 (2010 Edition), IEC 61511 (2015 Edition) and Related Guidance. Butterworth-Heinemann.