Human Error, Interaction and the Development of Safety-Critical Systems

This chapter summarises the theoretical and practical consequences of human error for the design, operation and maintenance of interactive systems. The focus is on safety-critical applications, in industries ranging from aviation to healthcare. However, human error has an impact on all systems that require human intervention. Individual failures have enormous financial consequences, although most cost less than the 40.5 billion yen that was lost in 2005 when a Japanese trader agreed to sell 610,000 shares for as little as 1 yen each, rather than one share for 610,000 yen (New York Times, 2005). Human error also has a cumulative effect. Less dramatic errors contribute to the frustration that many computer users experience as they work to complete deadlines and make deliveries in everyday working environments (Reason, Human Error, 1990). If designers, managers and regulators can identify those situations in which users are likely to make errors then we can help to minimise their impact (Johnson, Handbook of Accidents and Incidents, 2003). This may save lives, avoid financial disaster or simply increase the sense of satisfaction that users experience when they operate complex systems.

Part 1: What is Error?

This opening section summarises different perspectives on human error. The aim is to distinguish between different types of failure and also to identify the situations or contexts in which they are likely to occur. There is an increasing recognition that part of the responsibility for human error lies with designers and operators and not just with the end users of interactive systems (Hollnagel E. , The ETTO Principle: Efficiency-Thoroughness Trade-Off, 2009). This joint responsibility can be illustrated by a simple example. If an end user selects the wrong item from a menu then we might say that it was 'their fault'. If the user had read the manual then they would not have made this mistake. Equally, it can be argued that the menu item should have been given a more appropriate label so that the user never had to refer to the manual in the first place. This joint responsibility for error between the user and designer extends beyond the human-machine interface into the working environment. Many regulatory organisations take an increasingly broad view of the contexts in which errors are likely to occur (Kirwan, 1994). Even if end-users are provided with excellent user interfaces, errors will still happen if people are expected to work in noisy environments or cramped conditions against tight deadlines.

Resilience and the Myth of Expert Performance

It is difficult to understand human error unless we first consider the strengths that justify user involvement in complex tasks (Hollnagel, Woods, & Leveson, 2006). It is important to recognise that people are particularly good at coping with failure. We constantly adapt to problems in our working lives. We learn to operate poorly designed human-machine interfaces. The limitations of previous systems have been well documented in the other chapters of this book. These coping mechanisms help to explain why we find so many legacy applications that continue to be used even though they show no evidence of even the most basic human factors involvement in their design.

Over time users learn through their mistakes (Reason, Human Error, 1990). They find new ways of doing frequent tasks so that they avoid problems in the user interface. Hence, failure is a necessary component of learning. By making mistakes, users develop expertise and so paradoxically, error is a necessary component of expert performance (Rasmussen, 1983). Over time, frequent computer users will develop higher-level coping strategies that help them to reduce the number of errors that they make and hence to learn to use new systems faster. For instance, many graphical user interfaces deliberately encourage users to experiment with different aspects of their functionality. Desktop publishing systems will automatically update font selections as the user scrolls through a list of options. Frequent users quickly learn how to apply 'undo' commands to these selections so that they can get back to the previous version of document if they make a mistake. This supports further experimentation that, in turn, increases familiarity with the system and reduced the likelihood of further errors. Hence, it is possible to identify a 'virtuous circle' in which experimentation and error lead to learning and the accumulation of expertise. Equally, one can identify a 'vicious circle' in which novice users inadvertently change the font of their document but do not know enough about the user interface to be able to undo the change. This may lead them into further problems as they get further and further away from the original version of their document with every subsequent interaction.

A great deal of attention has recently been devoted to the topic of 'resilience engineering' (Hollnagel, Woods, & Leveson, 2006). This assumes that we should focus less on the causes of human error and more on the promotion of recovery actions, such as the application of undo in the previous desktop publishing example. Resilience engineering starts from the assumption that humans are not simply the cause of error, they act as a key means of mitigating failure in complex systems. This is a critical observation (Reason, 2008). For many years, developers have responded to the problem of human error in safety-critical systems by attempting to engineer-out the human involvement in these systems (Dekker, 2006). The argument is made that because even experts make mistakes we should minimise the opportunity for operator error to undermine safety. For this reason, engineers have worked hard to develop autonomous spacecraft, such as NASA's DART or the European Space Agency's Autonomous Transfer Vehicle, they have also developed automated systems that intervene for instance to apply automatic braking equipment in rail applications. However, these systems have had very mixed success. For instance, accidents and collisions have been caused when automated braking systems have been inadvertently triggered. In other words, removing or restricting operator intervention tends to move the opportunity for error to other areas of the development lifecycle (Johnson, Handbook of Accidents and Incidents, 2003). The end user may not be responsible for particular failures; however, errors tend to occur in the design and maintenance of the automated systems that are assuming new levels of control. It seems likely, therefore, that 'human error' will remain a significant concern for the development of complex systems.

Slips, Lapses, Mistakes and Violations

In this chapter we are mainly concerned with 'errors' as opposed to deliberate violations. A violation occurs when users knowingly break a rule (Reason, Human Error, 1990). These rules may

take the form of Standard Operating Procedures (SOPs) that govern the interaction with complex systems in the military, aviation or maritime environments. Other rules stem from the operating requirements placed on companies and individuals by regulatory agencies, for instance within the nuclear industry. Violations can also occur in more 'every day' settings – for instance when users ignore security policies by sending unencrypted emails or by accessing social networking sites in company time.

The distinction between errors and violations is not always as clear as it might seem. For instance, users can unwittingly violate rules if they are unaware of them or the rule is not clearly expressed. In such circumstances, it can be argues that an error has occurred rather than a deliberate violation. In other words, it is difficult to distinguish between errors and violations because these two forms of failure stem from different intentions even though the observable actions can be identical.

One way of identifying errors is to break down a task into its component steps (Kirwan, 1994). Any departure from those steps can be seen as a potential failure on the part of the user. Unfortunately, this does not really help designers very much. Many recent styles of interface enable the same task to be accomplished in many different ways. For example, this document is being composed on a desktop publishing system that offers at least seven different ways of changing the font associated with a word in the text. This can be done directly by altering the font associated with a selected word through a scroll down menu or through a text input box. It can be done as part of a more complex dialogue by right clicking over the word. It can also be done indirectly by editing the style associated with the word and so on. This leads to immense complexity even over a relatively simple and frequent task. It seems rather harsh to argue that the user committed an error if they inadvertently fail to select precisely the best means of changing the font at any single point during the editing of a document providing that they eventually achieve the desired goal. Very few users ever perform complex tasks in exactly the same way that a designer might envisage. There are also tremendous variations in performance between different users. Hence, what might seem to be an optimal trace of interaction for a novice might be seen as an 'error' for more expert users.

Instead of defining errors to be a departure from optimal performance, they might instead be defined as a departure from 'normal' operation. This definition acknowledges that we might normally expect tremendous variation between different users. It, therefore, addresses one of the problems that arose in comparing performance with a 'perfect approach'. What might be a 'normal' error to expect from a novice might not be expected for an expert. This definition also recognises that we might be able to recognise unusual or erroneous performance that differs from expected interaction. However, a number of theoretical and practical problems complicate this approach. For instance, we tend to commit hundreds of small errors every day. This is a consequence of the key role that errors play in learning. As we have seen, most people deploy 'coping strategies' so that they can recover from tiny failures within minimum disruption to their higher level tasks (Tversky & Kahneman, 1974). Errors often occur in successful interactions in which the user managed to achieve their goal. In many cases, we may not even realise that an error has occurred.

James Reason has approached the problem of defining error by looking at differences between intentions, actions and consequences (Reason, Human Error, 1990). If a user has an intention then they act in a manner that is intended to achieve a particular goal. For instance, I might plan to print

a double-sided version of this document using my desktop publishing application. Errors of omission can occur when users forget to perform an action – these are known as lapses. This could occur if I select the print dialogue but forget to click on the check-box for double sided printing. Errors of commission occur when users perform an action that does help them to achieve their goal- these are known as slips. For instance, I might inadvertently close the document while scrolling down the File menu. Finally, users may not be able to identify a plan of action that is an appropriate means of achieving their goal – these are known as mistakes. It would be a mistake to try and find the print option in the Edit menu of most publishing applications.

Situation Awareness and Team-Based Interaction

The discussion up to this point has focused narrowly on individual users interacting with particular applications. However, the topic of human error is often associated with the loss of situation awareness that occurs when users fail to identify the consequences of changes in their environment (Endsley, 2004). This is most apparent in safety-critical domains. For instance, if an Air Traffic Controller becomes preoccupied in other tasks then they may fail to hear the Short Term Conflict Alerts which are issued when two aircraft are in a dangerous proximity to each other. This oversight can lead to a loss of situation awareness that, in turn, can lead a controller to divert one aircraft into the path of another. This loss of awareness not only applies to the user's failure to monitor changes in software systems; it also arises when individuals fail to observe the actions of their colleagues. Hence it is an important aspect of team-based error. For instance, one person might send an email without realising that their co-worker had just sent exactly the same request moments earlier. In this context, a loss of situation awareness is closely associated with 'distributed cognition' – this is the process by which teams work together to achieve collective goals.

Endsley (2004)has developed a number of conceptual models that help us to understand the ways in which a loss of situation awareness can contribute to team-based errors. He argues that in order to interact with complex systems we must first be able to perceive key changes in our environment. If we miss warnings, such as the Short Term Conflict Alert, then we are likely to make mistakes. At a second level, we may perceive key information but fail to comprehend what those signals mean. For instance, it is unlikely that a lay person would be able to recognise the meaning of an STCA even if they heard the warning. Finally, a loss of situation awareness may contribute to errors if we cannot use the information that we obtain to make accurate predictions about future states. In other words, even if we recognise that a warning has been issues we must be able to identify those aircraft that are in danger.

A number of other researchers have extended this work in different directions. For instance, Klein has developed the ideas of Recognition Primed Decision Making – this suggests that we do not conduct detailed explicit analysis of the changes in our environment prior to making a plan for action (Klein, 1998). Instead we attempt to approximate a 'best fit' between what we see around us and things that we have met in previous interactions. Hence the user of a desktop publishing system does not spend hours reading manuals and experimenting with tutorials before starting to use an application. Instead, they are more likely to have a go by applying analogies and expertise gained in other similar systems. Because these analogies may be misleading, this 'best fit' process will inevitably lead to errors that call upon the coping strategies identified within resilience engineering.

Workload and Performance Shaping Factors

A range of factors can affect the likelihood that a user will make an error. These factors also determine whether, having committed an error, we can detect the problem and intervene to resolve or mitigate any adverse consequences. For instance, increasing workload can make a user more likely to make mistakes. Distractions and a number competing priorities can combine to undermine situation awareness (Endsley, 2004). If teams are working to the limit then individuals may not have enough time to update their colleagues on their actions and this can degrade distributed cognition in complex multi-user tasks. It is, however, notoriously difficult to predict the impact of workload on user performance. Some people commit errors in situations where their co-workers continue without any problems. There may be gender differences in the ability to juggle multiple tasks or to gain an overview across many competing priorities. These variations in performance are compounded by the difficulty in defining or measuring workload. One approach is to measure physical effort, for instance in terms of oxygen consumption over time using techniques developed within sports medicine. However, these metrics hardly apply to the mental workload that characterises most human computer interaction. Subjective approaches have also been used, for example, within NASA's Task Load Index (Hart, 2006). However, great care is required to calibrate these approaches and also to ensure that external influences do not bias the users' answers about their subjective impressions of an interactive system. A further problem is it can be difficult to extrapolate from subjective impressions as a means of predicting future errors under carrying levels of workload.

Alternatively, secondary tasks can assess the impact of workload on errors. It can be difficult to force users to make errors if their primary task is to focus on interaction with a particular user interface. As we have seen, users quickly become skilled at overcoming minor set-backs to achieve their overall task. From this it follows that designers can introduce additional secondary tasks to determine whether increasing workload will have an impact on the frequency and consequences of errors during interaction with a primary system. For example, users might be requested to perform a number of simple mental calculations as they use an interface. Overall performance can be judged both in terms of the number of errors made using the primary system and the number of errors made in the secondary task over time. Workload can be varied by adjusting the difficulty of the calculations or by reducing the time to complete each mental task.

Users react to increasing levels of workload in many different ways. Encysting occurs when individuals become so preoccupied with the details of a particular task that they ignore the bigger picture (Reason, Human Error, 1990). For example, a user may be so busy trying to alter the font in their title that they no longer have enough time to complete the rest of the report. In contrast, thematic vagabonding occurs when users move from one task to the next without devoting sufficient time to make real progress on any of them. It is important also to see these responses as part of a wider set of psychological mechanisms that users adopt when faced with a broader range of 'performance shaping factors' (Hollnagel E. , 1998). As mentioned previously, users are remarkably resilient to errors. They learn from them and work around them. However, these processes can be undermined by a range of issues including fatigue, distraction, heat, noise, work related stress, domestic stress, alcohol or drug consumption and so on. These negative

performance shaping factors can have such a profound impact on users that they undermine resilience to poor interface design. For instance, traders may continue to use a securities trading system without error until they are forced to make a series of complex trades before the close of business on a Friday afternoon. The additional pressure created by the deadline and the complexity of the transactions may combine to provide the performance shaping factors that lead to error.

Balanced against PSF's such as fatigue or stress, are a number of positive performance shaping factors – these include explicit training that is intended to help individuals and teams of operators understand the causes and consequences of human error. For instance, Crew Resource Management (CRM) techniques have been widely applied in aviation and in healthcare to encourage good practice in communication between groups of co-workers (Civil Aviation Authority, , 2006). The aim is to encourage mutual monitoring to support distributed cognition and mutual situation awareness. CRM is also intended to encourage a flexible allocation of tasks between groups as a means of responding to increasing levels of workload. In the previous example, teams of traders might work together to make final checks before a transaction is completed. This cooperation increases the level of mutual resilience against errors that continue to be made even after designers have 'optimised' interfaces to reduce the likelihood of erroneous transactions.

Context and Systemic Failures

Very often performance shaping factors are independent of the human computer interface. Stress, heat, noise, distraction and fatigue are all issues that emerge from the environment or context in which a system is being used (Johnson, Handbook of Accidents and Incidents, 2003). In safety-critical systems this has led to an escalation in the scope of accident investigations. In previous generations, the focus would have been on the system operator as the source of a potential error. Problems might then have been acknowledged in the human factors of complex systems – looking at the design and layout of critical controls as well as the format of information being displayed to the end user. Increasingly, however, the focus has moved to management and regulation to identify the ways in which higher level decisions create the working environment in which an error is more likely to occur. The user may not have been trained properly, they may have been asked to complete tasks with insufficient time to check for potential errors, they may not have access to adequate communications infrastructure to communicate with their colleagues and so on. In such circumstances, we must look at the management of an organisation which is responsible for creating the context in which users will make mistakes (Reason, 1997). Many Western countries have acknowledged this change in perspective by enacting legislation dealing with Corporate Killing or Corporate Manslaughter in addition to the Health and Safety Legislation that considers individual violations and errors (Johnson, 2008).

If we trace the links of influence back from the 'sharp end' at which an operator makes a mistake we can follow responsibility through middle and senior management. Ultimately, however, we can reach the regulator or government organisation that is responsible for supervising the market in which a company can operate. It is for this reason that many regulatory agencies have organised initiatives to address the human factors of error. Crew Resource Management is a requirement for all commercial aircrew within European and North American air space. Individual agencies have gone further – for instance EU-OPS and JAR-OPS 3 Subpart N contain requirements for anyone

delivering CRM training. The UK Civil Aviation Authority has issued specific requirements within a document known as CAP 737- "Crew Resource Management (CRM) Training" and a CAA Standards Document 29.

Emotional Aspects of Error

The previous paragraphs might have given the impression that human error was a subject for abstract academic research and of concern to engineers in a narrow range of safety or security related industries. However, it is important to reiterate that error is part of the human condition. The ability to learn from mistakes is a key component of resilience in every form of interaction ranging from the use of spreadsheets through to mobile phones and missile systems. It is also important to stress that there is an emotional aspect to error (Dekker, 2006). The process of learning from previous incidents or accidents can be undermined by the feelings of guilt and blame that are often felt by the individuals who are involved in adverse events. Fear of retribution or sanction can encourage users to destroy logs or other forms of evidence that might be used to avoid future errors – for example by improved training or through redesign. These emotions can also create a situation in which one error can trigger further failures. For instance, users may become so preoccupied with replaying the previous failure 'in their head' that they commit further errors. It is for this reason that Air Traffic Control Officers will, typically, be removed from further duties if they have been involved in a near miss incident. In the same way, the users of more general application can experience increasing levels of frustration as time pressure or stress lead from one error to the next.

Part 2: Consequences of Error for the Design of Complex User Interfaces

The first part of this chapter has provided an overview of human error – for example by distinguishing between violations, slips, lapses and mistakes (Reason, 1990). We have also pointed out the remarkable resilience that enables users to overcome and learn from the hundreds of small errors that characterise everyday interaction. In contrast, this second part of the chapter looks at the consequences of error for the design and operation of complex user interfaces. In particular, we look at attempts to model human error and thereby to minimize the likelihood of slips, lapses and mistakes. We also look at the problems that arise when trying to validate predictions of particular errors rates in a given context of use (Bainbridge, 1987).

Simulation and Verification?

One of the problems that complicate the development of interactive systems is that it is hard for designers to anticipate the problems that users will experience. Simply being involved in the development of a complex application will give developers insights that would often never occur to everyday users. These insights can be derived from hours of work on dialogue design, as well as a direct knowledge of implementation mechanisms or the probable error modes that could complicate the use of software systems. It is difficult, therefore, to convey the sense of surprise that

development teams often experience when they see users struggling to operate the systems that they have delivered (Norman, 1990).

One reason for the difficulty in anticipating errors is that it can be difficult to reproduce errors under experimental or laboratory conditions. People behave differently if they know they are being watched – they read instructions more diligently and often will obey warnings that would simply elicit an automatic cancel in the workplace. Other problems relate more directly to the practical difficulty of testing for errors. It can be difficult to recreate the full range of performance shaping factors, including stress and fatigue that might be encountered in the workplace – often there are ethical concerns about running these types of evaluations prior to deployment. Serious errors will hopefully be relatively rare events; hence tests may have to run for weeks or months before one is observed. Even if such events are never detected during a test then this may provide developers with little confidence about the eventual reliability of their system. Dijkstra noted that testing only ever established the presence of a design problem but not its absence. If the tests were continued over a longer period, with different input data or with different users then further evidence may be obtained about mistakes, slips and lapses.

A number of alternate techniques might be recruited to help identify problems of situation awareness, workload or distributed cognition during the development of complex, interactive systems. For example, participatory design techniques recruit potential end-users to development teams. Part of their role can be to alert designers to potential errors that might complicate the operation of the system in the eventual working environment. However, this can be difficult to manage. Over time it can be increasingly hard for end users to disassociate themselves from the development team. By working with a design for a prolonged period of time, it can be increasingly difficult for individuals to place themselves back in the position of one of their co-workers using an interactive system for the first time (Beyer & Holtzblatt, 1998).

User modelling techniques a provide alternative for the development of interactive systems. These try to model some of the cognitive mechanisms that both support resilience but which also lead to errors involving software applications. For instance, they can be used to model the limited capacity of short term memory or the interference effects from high workload that have been observed as triggers for slips and lapses (Johnson, 1999). This approach can be used not only to predict potential error mechanisms but also to provide an explanation for why those errors might occur. This is important if development teams are to identify potential design solutions. Unfortunately, the application of user modelling creates new challenges. Rather than simply establishing that error predictions are observed in the use of an interactive system, it also becomes important to ensure that the models also provide a valid explanation of the underlying error mechanisms.

Human Reliability Analysis

Risk analysis continues to play a key role in the design of safety-critical systems. This process provides numeric estimates of the likelihood and consequences of the different hazards that can lead to an adverse event. Quantified risk assessments works best for hardware systems for which it is possible to derive statistical estimates of the probability of random failures over time. For example, the US military publish handbooks that contain the probability of failure for a range of

components over time (US Department of Defense, 1995). Designers can use this data to calculate the probability of failure for a system that is built from these various individual components. In the same way, Swain and Guttman (1983) have sought to publish handbooks of human reliability that provide high-level estimates for the likelihood of particular types of error. For example, the probability that a user might incorrectly read back a series of figures from a display might be 1 in 200 attempts. This approach has numerous advantages for the engineering of complex systems – the same risk-based approaches can be applied throughout all aspects of the development process.

A number of concerns limit the practical application of human reliability analysis. Firstly, critics of the approach have argued that the probabilities can be difficult to validate (Reason, 1990). If users know that their actions are being observed then they may be less likely to make an error; they will exploit a range of self-monitoring techniques to catch and rectify any mistakes that might jeopardise their tasks. Secondly, even if accurate data is available for previous errors in similar systems then gross estimates do not take into account a host of more complex cognitive and social factors. Hollnagel (1998) voices this criticism when he describes Human Reliability Analysis as 'psychologically vacuous'. In other words, this approach often neglects the impact of performance shaping factors on the likelihood of user errors. More recent methodologies have sought to address these criticisms by helping designers to first calculate the base probability for particular errors and then apply deltas or modifying terms to equations that account for the impact of performance shaping factors. If a user is likely to be acting under time pressure then the probability of incorrectly reading back some data is increased. If they have prior training in techniques like Crew Resource Management then a delta may be applied to reduce the probability of such errors. Although these developments help to address caveats about the application of Human Reliability Analysis, they also raise further concerns about the validation of both the base probabilities and also the 'fudge factors' that are introduced to account for performance variations.

In less critical domains, it can be difficult to make an appropriate business case to justify the investments that are needed to support human reliability analysis as a means of predicting potential errors with mass-market interactive systems. These techniques have been used in financial software and in other forms of secure applications. The longevity of the approach is certainly surprising given the sustained theoretical objections, mentioned in previous paragraphs.

Incident and Accident Reporting

Incident and accident reports can be used to identify the base probabilities that are used in Human Reliability Analysis (Johnson, 2003). They provide useful evidence about those human errors that have the potential to threaten safe and successful operation. Unlike more heuristic forms of design, they provide a clear and tangible link with operational experience and hence usually provide clear insights into the impact of performance shaping factors, including fatigue, interruptions and stress. It is important not only to focus on the role of human error in major adverse events. By focusing on the individual errors that led to the loss of billions of Yen, as in the example cited in the opening sections, we might waste resources trying to protect against the last failure rather than preparing for the next. Similarly, we might also neglect the many less significant errors that have a greater cumulative impact on successful interaction over the life time of a system. The Heinrich ratio is widely used to relate the number of accidents to serious and to minor incidents. Typically, this is

written as 1 accident to 30 major incidents to 300 minor incidents (Johnson, 2003). The precise figures in the Heinrich ratio remain a focus for considerable debate and they vary between industries. For example, studies of railway maintenance have shown that there is a bi-polar distinction between accidents which tend to be fatal and near-miss incidents that were a long way from resulting in an accident.

In everyday environments, system logs and critical incident diaries provide equivalents of accident and incident reports in safety-related applications. For example, web server logs can be instrumented to monitor for situations in which users decide to terminate a transaction before payment is confirmed. These can then form the focus for other types of usability study based on scenarios that mimic the trajectory of failure that is illustrated in the logs. Critical incident diaries require users to note down any adverse events that arise during their interaction with interactive applications. They may also be prompted to provide evidence about any performance shaping factors that may have influenced the problems that they experienced or which increased the likelihood of an error even if they managed to spot the problem in time.

In spite of these different approaches to eliciting evidence about previous failure, considerable uncertainty remains over the reliability of any information obtained in this way. For example, incident diaries depend upon users being motivated enough to remember to pause after an error which will already have delayed their primary task in order to complete an entry in their diary. In the initial phase after an incident reporting system has been installed, companies can quickly become overwhelmed by a mass of relatively low priority incidents as users learn to report their concerns. After this 'confessional phase', the number of incident reports often dries up. Under-reporting remains a problem even in safety-critical industries where staff may be concerned that they will be punished if they confess to particular errors. Management may be worried about subsequent litigation or the loss of operating licenses if they are informed about near-miss incidents. Although automated logs can provide a trace of the interactions that lead to potential failures, they provide relatively few insights into the cognitive mechanisms and performance shaping factors that might have led users to make particular errors (Dekker, 2006).

As one might expect, accident and incident analysis have been profoundly affected by changes in wider research into human error. For example, many previous studies have been conducted to derive estimates for the number of accidents that are caused by mistakes, slips and lapses (Reason, 1997). These studies have typically been used in one of two ways – either to increase spending on human factors and interface design to reduce operator error or to justify increased spending on automation to entirely remove user input. There is, however, a growing recognition that human error only acts as a catalyst or trigger for underlying problems in the management and operation of complex systems. In this view, we can think of interactive systems being protected by different barriers. If an industry is well regulated, the company is well managed and the operator follows standard operating procedures then an accident is unlikely to occur. If, however, there are ambiguities in the regulatory regime and the management have not provided a suitable working environment or equipment and the user switches off elements of the protection system then we can see how holes appear in each layer of defence. This model is sometimes referred to as Reason's Swiss cheese model, for obvious reasons (Reason, 1990).

It can be difficult to ensure that organisations and individuals act on information about previous errors, even if such insights can be obtained in a reliable manner. For instance, attribution bias is one of several cognitive phenomena that affect the way we determine who was responsible for an adverse event (Johnson, 2003). Salience is a particularly important aspect of this bias because issues that were critical for the person involved in an adverse event may not be apparent to outside observers. This makes it difficult for designers to understand the reasons why an individual might have made a mistake, slip or lapse. Further problems arise when we are quick to condemn the errors of others while we are eager to identify the impact of performance shaping factors in explaining our own failures.

Lifecycle issues: Design, Operation, Maintenance and Decommissioning

Incident and accident reporting can be used to form a feedback look that is intended to ensure that organisations learn from those failures that do occur. In systems engineering, prospective risk assessments that were used to guide the early stages of development can be validated against the data that is derived about real hazards during the operation of complex, interactive systems. As we have seen, however, these feedback loops will only work if users are encouraged to provide information about the errors that they commit or observe. In addition there also needs to be appropriate mechanisms that enable end user concerns to be communicated back to development teams. This is not as simple as it might sound. Very often the groups who are involved in the design of an interactive system will move to subsequent projects as soon as an application has been delivered. From this it follows that they may never hear about the problems that end users experience in the everyday operation of complex systems (Beyer & Holtzblatt, 1998). Other problems arise when applications are developed for one market place and are then transferred to new user groups with minimal changes in their user interface. For instance, many GPS units that were designed to help private pilots are now being sold into the maritime market. A new crop of navigation errors and accidents have occurred with marine systems because some designers have failed to learn the lessons provided by the initial application of these units within the aviation community.

Further problems arise when the end users of safety-critical systems have to learn to cope not only with design problems in their user interfaces but also with problems in underlying applications. Many accidents have occurred during what are termed ‘degraded modes of operation’ (Johnson, Kirwan, & Licu, 2009). These arise when users struggle to maintain levels of service even though they may have lost key components in their underlying systems infrastructure. Examples include situations in which Air Traffic Controllers have tried to keep aircraft moving in low visibility during a failure of their ground movement radar system or when train drivers have proceeded through red lights without being able to confirm that the track ahead is clear because of faulty communications with signalling staff or systems. Such situations represent particular challenges for the development of complex, interactive systems because designers must consider not simply the errors that can occur with a ‘perfect application’ but also what could happen weeks and months into the future when imperfect maintenance activities lead to degraded modes of operation.

Not only do designers have to consider the problems that arise during the development and maintenance of complex systems, there is also a requirement to consider decommissioning

activities. Increasing concerns over the environmental impact of safety-related applications has persuaded regulators to extend a lifecycle approach to the mitigation of operator error. Lessons learned from the decommissioning of nuclear and military installations have been used to persuade the operators of complex systems that they must consider the hazards that can arise from mistakes, slips and lapses at the end of the working lives of many applications.

Safety Cases and Safety Management Systems

Safety Management Systems have been developed to formalise the processes of organisational learning that are intended to ensure we learn the lessons from previous errors. They, typically, include guidance on the documentation both of risk assessments and of incident reports. They also help to formalise responsibility for ensuring that the concerns of end users are considered and, where necessary, are acted upon. Safety cases are an increasingly important component of safety management systems. These can be thought of as arguments about why a complex system is 'acceptably safe' (Johnson, 2003). Safety cases are significant from the perspective of this handbook because they often include assumptions and evidence about operator intervention. For instance, it might be argued that a control system is acceptably safe, in part, because the user can respond to a critical warning by shutting down operations within a fixed time period. This argument might, in turn, be supported by evidence derived from user testing in simulators. The safety case might later be amended to reinforce this argument with further data derived from operational experience with the application. As we have seen, however, the theoretical and practical problems in predicting potential errors makes it difficult to place high levels of confidence in arguments that rest on the ability of operators to meet such requirements. Safety cases must, therefore, often extend arguments about operator intervention to demonstrate that the system will still be acceptably safe even when a user fails to intervene in the manner that might initially be expected.

The processes and products that help to define Safety Management Systems are important from another perspective. They illustrate the extent to which developers must go to mitigate the consequences of human error not simply in the operation of complex systems but in the design, implementation, operation, maintenance and decommissioning of safety-related applications. Mistakes, slips and lapses complicate every stage of the lifecycle in human machine systems. This makes it difficult to identify the boundaries of safety-critical applications. For instance, it is obvious that the systems used by clinicians, pilots or nuclear engineers have a direct impact on safety. However, it can also be argued that the user interfaces to the tools that designers employ in the development of aerospace, healthcare and nuclear systems will also indirectly have an impact on the safety of the final applications. The care and thought that goes into the development of safety-critical systems is often lacking in the design of software and interface development tools. Similarly, the user interfaces of the reporting systems that are often intended to elicit information about previous errors with other systems are often poorly designed. In such circumstances, it can hardly be a surprise that so little is known about the mistakes, slips and lapses that end users experience in their everyday work.

Part 3: Conclusions and Looking to the Future

This chapter has provided a broad overview of human error in the context of both safety-critical and mass market applications. It is impossible to provide a complete survey of such a vast and active area of research. The continued prevalence of human error as a contributory cause in accidents across many industries also demonstrates that much remains to be learned in this area. However, we have identified key insights that emerge from the application of research ideas in a range of different industries. The distinctions between violations, mistakes, slips and lapses, the interaction between performance shaping factors and the quantitative predictions of human reliability analysis, the importance of workload and situation awareness, as well as the Swiss cheese model of accident causation have all informed the development and operation of complex interactive systems in domains ranging from healthcare to power distribution, from aviation to the military.

Looking to the future, it seems likely that we will see increasing levels of automation as a potential means of reducing the impact of human error. This is already apparent, for example in the development of autonomous space craft for future manned missions to Mars and the Moon or in proposals to extend the role of Unmanned Autonomous Vehicles into controlled airspace. However, these developments will not reduce the importance of human error. Instead the focus may shift away from the immediate operator of a safety critical system onto the teams that must develop and maintain increasing levels of complexity.

Bibliography

Bainbridge, L. (1987). Ironies of Automation. In J. Rasmussen, K. Duncan, & J. Leplat, *New technology and human error.* New York: Wiley.

Beyer, H., & Holtzblatt, K. (1998). *Contextual design: Defining customer-centered systems.* San Francisco: Morgan Kaufmann.

Civil Aviation Authority, . (2006). *CAP 737: Crew Resource Management Training.* Farnham, UK: UK Civil Aviation Authority.

Dekker, S. (2006). *THE FIELD GUIDE TO UNDERSTANDING HUMAN ERROR.* Aldershot, UK: Ashgate.

Endsley, M. (2004). Situation awareness: Progress and directions. In S. Banbury, & S. Tremblay, *A cognitive approach to situation awareness: Theory, measurement and application* (pp. 317–341). Aldershot, UK: Ashgate.

Hart, G. (2006). NASA-Task Load Index (NASA-TLX); 20 years later. *Proceedings of the Human Factors and Ergonomics Society 50th Annual Meeting* (pp. 904-908). Santa Monica, CA: Human Factors & Ergonomics Society.

Hollnagel, E. (1998). *CREAM (Cognitive Reliability and Error Analysis Method).* North Holland: Elsevier.

Hollnagel, E. (2009). *The ETTO Principle: Efficiency-Thoroughness Trade-Off.* Farnham, UK: Ashgate Publishing.

Hollnagel, E., Woods, D., & Leveson, N. (2006). *Resilience Engineering: Concepts and Precepts.* Aldershot, UK: Ashgate.

Johnson, C. (2003). *Handbook of Accidents and Incidents.* Glasgow, Scotland.: Glasgow University Press.

Johnson, C. (2008). Ten contentions of corporate manslaughter legislation: Public policy and the legal response to workplace accidents. *Safety Science* , 46:3(349-370).

Johnson, C. (1999). The Application of User Modeling Techniques to Reason about the Human Contribution to Major Accidents. *Proceedings of the 7th International Conference on User Modelling, Banff Canada,* (pp. 13-22). New York, USA: Springer Verlag.

Johnson, C., Kirwan, B., & Licu, T. (2009). The Interation Between Safety Culture and Degraded Modes: A Survey of National Infrastructures for Air Traffic Management. *Risk Management* , 11:3(241-284).

Kirwan, B. (1994). *A Guide to Practical Human Reliability Assessment.* London, UK: Taylor and Francis.

Klein, G. (1998). *Sources of Power: How People Make Decisions.* Boston, USA: MIT Press.

New York Times. (2005, December Friday 9th). Japan rebukes exchange for costly trading error.

Norman, D. (1990). The "problem" of automation: Inappropriate feedback and interaction, not "over-automation. In D. Broadbent, A. Baddeley, & J. Reason, *Human factors in hazardous situations* (p. 585). Oxford: Oxford University Press.

Rasmussen, J. (1983). Skills, rules, and knowledge: Signals, signs, and symbols and other distinctions in human performance models. *IEEE Transactions on Systems, Man, and Cybernetics* , SMC-13, 257-267.

Reason, J. (1990). *Human Error.* Cambridge: Cambridge University Press.

Reason, J. (1997). *Managing the Risks of Organisational Accidents.* Aldershot, UK: Ashgate.

Reason, J. (2008). *The Human Contribution: Unsafe Acts, Accidents and Heroic Recoveries.* Aldershot, UK: Ashgate.

Swain, A. D., & Guttman, H. E. (1983). *Handbook of human reliability analysis with emphasis on nuclear power plant applications.* Washington DC, USA.: NUREG/CR-1278.

Tversky, A., & Kahneman, D. (1974). Judgment under uncertainty: Heuristics and biases. *Science* , 185, 1124-1131.

US Department of Defense. (1995). *MIL-HDBK-217: Reliability Prediction of Electronic Equipment.* Washington DC, USA.: US Department of Defense.