answers immediately, even if those answers are wrong. Moreover, when the full story finally appears, newspapers will no longer consider it news, so they won't report it. You will have to search for the official report. In the United States, the National Transportation Safety Board (NTSB) can be trusted. NTSB conducts careful investigations of all major aviation, automobile and truck, train, ship, and pipeline incidents. (Pipelines? Sure: pipelines transport coal, gas, and oil.)

Designing for Error

It is relatively easy to design for the situation where everything goes well, where people use the device in the way that was intended, and no unforeseen events occur. The tricky part is to design for when things go wrong.

Consider a conversation between two people. Are errors made? Yes, but they are not treated as such. If a person says something that is not understandable, we ask for clarification. If a person says something that we believe to be false, we question and debate. We don't issue a warning signal. We don't beep. We don't give error messages. We ask for more information and engage in mutual dialogue to reach an understanding. In normal conversations between two friends, misstatements are taken as normal, as approximations to what was really meant. Grammatical errors, self-corrections, and restarted phrases are ignored. In fact, they are usually not even detected because we concentrate upon the intended meaning, not the surface features.

Machines are not intelligent enough to determine the meaning of our actions, but even so, they are far less intelligent than they could be. With our products, if we do something inappropriate, if the action fits the proper format for a command, the product does it, even if it is outrageously dangerous. This has led to tragic accidents, especially in health care, where inappropriate design of infusion pumps and X-ray machines allowed extreme overdoses of medication or radiation to be administered to patients, leading to their deaths. In financial institutions, simple keyboard errors have led to huge financial transactions, far beyond normal limits.

Even simple checks for reasonableness would have stopped all of these errors. (This is discussed at the end of the chapter under the heading "Sensibility Checks.")

Many systems compound the problem by making it easy to err but difficult or impossible to discover error or to recover from it. It should not be possible for one simple error to cause widespread damage. Here is what should be done:

- Understand the causes of error and design to minimize those causes.
- Do sensibility checks. Does the action pass the "common sense" test?
- Make it possible to reverse actions—to "undo" them—or make it harder to do what cannot be reversed.
- Make it easier for people to discover the errors that do occur, and make them easier to correct.
- Don't treat the action as an error; rather, try to help the person complete the action properly. Think of the action as an approximation to what is desired.

As this chapter demonstrates, we know a lot about errors. Thus, novices are more likely to make mistakes than slips, whereas experts are more likely to make slips. Mistakes often arise from ambiguous or unclear information about the current state of a system, the lack of a good conceptual model, and inappropriate procedures. Recall that most mistakes result from erroneous choice of goal or plan or erroneous evaluation and interpretation. All of these come about through poor information provided by the system about the choice of goals and the means to accomplish them (plans), and poor-quality feedback about what has actually happened.

A major source of error, especially memory-lapse errors, is interruption. When an activity is interrupted by some other event, the cost of the interruption is far greater than the loss of the time required to deal with the interruption: it is also the cost of resuming the interrupted activity. To resume, it is necessary to remember precisely the previous state of the activity: what the goal was, where one was in the action cycle, and the relevant state of the system. Most systems make it difficult to resume after an interruption.

Most discard critical information that is needed by the user to remember the numerous small decisions that had been made, the things that were in the person's short-term memory, to say nothing of the current state of the system. What still needs to be done? Maybe I was finished? It is no wonder that many slips and mistakes are the result of interruptions.

Multitasking, whereby we deliberately do several tasks simultaneously, erroneously appears to be an efficient way of getting a lot done. It is much beloved by teenagers and busy workers, but in fact, all the evidence points to severe degradation of performance, increased errors, and a general lack of both quality and efficiency. Doing two tasks at once takes longer than the sum of the times it would take to do each alone. Even as simple and common a task as talking on a hands-free cell phone while driving leads to serious degradation of driving skills. One study even showed that cell phone usage during walking led to serious deficits: "Cell phone users walked more slowly, changed directions more frequently, and were less likely to acknowledge other people than individuals in the other conditions. In the second study, we found that cell phone users were less likely to notice an unusual activity along their walking route (a unicycling clown)" (Hyman, Boss, Wise, McKenzie, & Caggiano, 2010).

A large percentage of medical errors are due to interruptions. In aviation, where interruptions were also determined to be a major problem during the critical phases of flying—landing and takeoff—the US Federal Aviation Authority (FAA) requires what it calls a "Sterile Cockpit Configuration," whereby pilots are not allowed to discuss any topic not directly related to the control of the airplane during these critical periods. In addition, the flight attendants are not permitted to talk to the pilots during these phases (which has at times led to the opposite error—failure to inform the pilots of emergency situations).

Establishing similar sterile periods would be of great benefit to many professions, including medicine and other safety-critical operations. My wife and I follow this convention in driving: when the driver is entering or leaving a high-speed highway, conversation ceases until the transition has been completed. Interruptions and distractions lead to errors, both mistakes and slips.

Warning signals are usually not the answer. Consider the control room of a nuclear power plant, the cockpit of a commercial aircraft, or the operating room of a hospital. Each has a large number of different instruments, gauges, and controls, all with signals that tend to sound similar because they all use simple tone generators to beep their warnings. There is no coordination among the instruments, which means that in major emergencies, they all sound at once. Most can be ignored anyway because they tell the operator about something that is already known. Each competes with the others to be heard, interfering with efforts to address the problem.

Unnecessary, annoying alarms occur in numerous situations. How do people cope? By disconnecting warning signals, taping over warning lights (or removing the bulbs), silencing bells, and basically getting rid of all the safety warnings. The problem comes after such alarms are disabled, either when people forget to restore the warning systems (there are those memory-lapse slips again), or if a different incident happens while the alarms are disconnected. At that point, nobody notices. Warnings and safety methods must be used with care and intelligence, taking into account the tradeoffs for the people who are affected.

The design of warning signals is surprisingly complex. They have to be loud or bright enough to be noticed, but not so loud or bright that they become annoying distractions. The signal has to both attract attention (act as a signifier of critical information) and also deliver information about the nature of the event that is being signified. The various instruments need to have a coordinated response, which means that there must be international standards and collaboration among the many design teams from different, often competing, companies. Although considerable research has been directed toward this problem, including the development of national standards for alarm management systems, the problem still remains in many situations.

More and more of our machines present information through speech. But like all approaches, this has both strengths and weaknesses. It allows for precise information to be conveyed, especially when the person's visual attention is directed elsewhere. But if several speech warnings operate at the same time, or if the environment is noisy, speech warnings may not be understood. Or if conversations among the users or operators are necessary, speech warnings will interfere. Speech warning signals can be effective, but only if used intelligently.

DESIGN LESSONS FROM THE STUDY OF ERRORS

Several design lessons can be drawn from the study of errors, one for preventing errors before they occur and one for detecting and correcting them when they do occur. In general, the solutions follow directly from the preceding analyses.

ADDING CONSTRAINTS TO BLOCK ERRORS

Prevention often involves adding specific constraints to actions. In the physical world, this can be done through clever use of shape and size. For example, in automobiles, a variety of fluids are required for safe operation and maintenance: engine oil, transmission oil, brake fluid, windshield washer solution, radiator coolant, battery water, and gasoline. Putting the wrong fluid into a reservoir could lead to serious damage or even an accident. Automobile manufacturers try to minimize these errors by segregating the filling points, thereby reducing description-similarity errors. When the filling points for fluids that should be added only occasionally or by qualified mechanics are located separately from those for fluids used more frequently, the average motorist is unlikely to use the incorrect filling points. Errors in adding fluids to the wrong container can be minimized by making the openings have different sizes and shapes, providing physical constraints against inappropriate filling. Different fluids often have different colors so that they can be distinguished. All these are excellent ways to minimize errors. Similar techniques are in widespread use in hospitals and industry. All of these are intelligent applications of constraints, forcing functions, and poka-yoke.

Electronic systems have a wide range of methods that could be used to reduce error. One is to segregate controls, so that easily confused controls are located far from one another. Another is to use separate modules, so that any control not directly relevant to the current operation is not visible on the screen, but requires extra effort to get to.

UNDO

Perhaps the most powerful tool to minimize the impact of errors is the Undo command in modern electronic systems, reversing the operations performed by the previous command, wherever possible. The best systems have multiple levels of undoing, so it is possible to undo an entire sequence of actions.

Obviously, undoing is not always possible. Sometimes, it is only effective if done immediately after the action. Still, it is a powerful tool to minimize the impact of error. It is still amazing to me that many electronic and computer-based systems fail to provide a means to undo even where it is clearly possible and desirable.

CONFIRMATION AND ERROR MESSAGES

Many systems try to prevent errors by requiring confirmation before a command will be executed, especially when the action will destroy something of importance. But these requests are usually ill-timed because after requesting an operation, people are usually certain they want it done. Hence the standard joke about such warnings:

Person: Delete "my most important file."
System: Do you want to delete "my most important file"?
Person: Yes.
System: Are you certain?
Person: Yes!
System "My most favorite file" has been deleted.
Person: Oh. Damn.

The request for confirmation seems like an irritant rather than an essential safety check because the person tends to focus upon the action rather than the object that is being acted upon. A better check would be a prominent display of both the action to be taken and the object, perhaps with the choice of "cancel" or "do it." The important point is making salient what the implications of the action are. Of course, it is because of errors of this sort that the Undo command is so important. With traditional graphical user interfaces on computers, not only is Undo a standard command, but when files are "deleted," they are actually simply moved from sight and stored in the file folder named "Trash," so that in the above example, the person could open the Trash and retrieve the erroneously deleted file.

Confirmations have different implications for slips and mistakes. When I am writing, I use two very large displays and a powerful computer. I might have seven to ten applications running simultaneously. I have sometimes had as many as forty open windows. Suppose I activate the command that closes one of the windows, which triggers a confirmatory message: did I wish to close the window? How I deal with this depends upon why I requested that the window be closed. If it was a slip, the confirmation required will be useful. If it was by mistake, I am apt to ignore it. Consider these two examples:

A slip leads me to close the wrong window.

Suppose I intended to type the word *We,* but instead of typing Shift + W for the first character, I typed Command + W (or Control + W), the keyboard command for closing a window. Because I expected the screen to display an uppercase W, when a dialog box appeared, asking whether I really wanted to delete the file, I would be surprised, which would immediately alert me to the slip. I would cancel the action (an alternative thoughtfully provided by the dialog box) and retype the Shift + W, carefully this time.

A mistake leads me to close the wrong window.

Now suppose I really intended to close a window. I often use a temporary file in a window to keep notes about the chapter I am working on. When I am finished with it, I close it without saving its contents—after all, I am finished. But because I usually have multiple windows open, it is very easy to close the wrong one. The computer assumes that all commands apply to the active window—the one where the last actions had been performed (and which contains the text cursor). But if I reviewed the temporary window prior to closing it, my visual attention is focused upon that window, and when I decide to close it, I forget that it is not the active window from the computer's point of view. So I issue the command to shut the window, the computer presents me with a dialog box, asking for confirmation, and I accept it, choosing the option not to save my work. Because the dialog box was expected, I didn't bother to read it. As a result, I closed the wrong window and worse, did not save any of the typing, possibly losing considerable work. Warning messages are surprisingly ineffective against mistakes (even nice requests, such as the one shown in Chapter 4, Figure 4.6, page 143).

Was this a mistake or a slip? Both. Issuing the "close" command while the wrong window was active is a memory-lapse slip. But deciding not to read the dialog box and accepting it without saving the contents is a mistake (two mistakes, actually).

What can a designer do? Several things:

- **Make the item being acted upon more prominent.** That is, change the appearance of the actual object being acted upon to be more visible: enlarge it, or perhaps change its color.
- **Make the operation reversible.** If the person saves the content, no harm is done except the annoyance of having to reopen the file. If the person elects Don't Save, the system could secretly save the contents, and the next time the person opened the file, it could ask whether it should restore it to the latest condition.

SENSIBILITY CHECKS

Electronic systems have another advantage over mechanical ones: they can check to make sure that the requested operation is sensible.

It is amazing that in today's world, medical personnel can accidentally request a radiation dose a thousand times larger than normal and have the equipment meekly comply. In some cases, it isn't even possible for the operator to notice the error.

Similarly, errors in stating monetary sums can lead to disastrous results, even though a quick glance at the amount would indicate that something was badly off. For example, there are roughly 1,000 Korean won to the US dollar. Suppose I wanted to transfer $1,000 into a Korean bank account in *won* ($1,000 is roughly ₩1,000,000). But suppose I enter the Korean number into the dollar field. Oops—I'm trying to transfer a million dollars. Intelligent systems would take note of the normal size of my transactions, querying if the amount was considerably larger than normal. For me, it would query the million-dollar request. Less intelligent systems would blindly follow instructions, even though I did not have a million dollars in my account (in fact, I would probably be charged a fee for overdrawing my account).

Sensibility checks, of course, are also the answer to the serious errors caused when inappropriate values are entered into hospital medication and X-ray systems or in financial transactions, as discussed earlier in this chapter.

MINIMIZING SLIPS

Slips most frequently occur when the conscious mind is distracted, either by some other event or simply because the action being performed is so well learned that it can be done automatically, without conscious attention. As a result, the person does not pay sufficient attention to the action or its consequences. It might therefore seem that one way to minimize slips is to ensure that people always pay close, conscious attention to the acts being done.

Bad idea. Skilled behavior is subconscious, which means it is fast, effortless, and usually accurate. Because it is so automatic, we can type at high speeds even while the conscious mind is occupied composing the words. This is why we can walk and talk while navigating traffic and obstacles. If we had to pay conscious attention to every little thing we did, we would accomplish far less in our lives. The information processing structures of the brain automatically regulate how much conscious attention is being paid to a task: conversations automatically pause when crossing the street amid busy traffic. Don't count on it, though: if too much attention is focused on something else, the fact that the traffic is getting dangerous might not be noted.

Many slips can be minimized by ensuring that the actions and their controls are as dissimilar as possible, or at least, as physically far apart as possible. Mode errors can be eliminated by the simple expedient of eliminating most modes and, if this is not possible, by making the modes very visible and distinct from one another.

The best way of mitigating slips is to provide perceptible feedback about the nature of the action being performed, then very perceptible feedback describing the new resulting state, coupled with a mechanism that allows the error to be undone. For example, the use of machine-readable codes has led to a dramatic reduction in the delivery of wrong medications to patients. Prescriptions sent to the pharmacy are given electronic codes, so the pharmacist can scan both the prescription and the resulting medication to ensure they are the same. Then, the nursing staff at the hospital scans both the label of the medication and the tag worn around the patient's wrist to ensure that the medication is being given to the correct individual. Moreover, the computer system can flag repeated administration of the same medication. These scans do increase the workload, but only slightly. Other kinds of errors are still possible, but these simple steps have already been proven worthwhile.

Common engineering and design practices seem as if they are deliberately intended to cause slips. Rows of identical controls or meters is a sure recipe for description-similarity errors. Internal modes that are not very conspicuously marked are a clear driver of mode errors. Situations with numerous interruptions, yet where the design assumes undivided attention, are a clear enabler of memory lapses—and almost no equipment today is designed to support the numerous interruptions that so many situations entail. And failure to provide assistance and visible reminders for performing infrequent procedures that are similar to much more frequent ones leads to capture errors, where the more frequent actions are performed rather than the correct ones for the situation. Procedures should be designed so that the initial steps are as dissimilar as possible.

The important message is that good design can prevent slips and mistakes. Design can save lives.

THE SWISS CHEESE MODEL OF HOW ERRORS LEAD TO ACCIDENTS

Fortunately, most errors do not lead to accidents. Accidents often have numerous contributing causes, no single one of which is the root cause of the incident.

James Reason likes to explain this by invoking the metaphor of multiple slices of Swiss cheese, the cheese famous for being riddled with holes (Figure 5.3). If each slice of cheese represents a condition in the task being done, an accident can happen only if holes in all four slices of cheese are lined up just right. In well-designed systems, there can be many equipment failures, many errors, but they will not lead to an accident unless they all line up precisely. Any leakage—passageway through a hole—is most likely blocked at the next level. Well-designed systems are resilient against failure.

This is why the attempt to find "the" cause of an accident is usually doomed to fail. Accident investigators, the press, government officials, and the everyday citizen like to find simple explanations for the cause of an accident. "See, if the hole in slice A

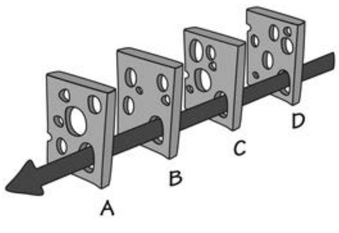

FIGURE 5.3. **Reason's Swiss Cheese Model of Accidents.** Accidents usually have multiple causes, whereby had any single one of those causes not happened, the accident would not have occurred. The British accident researcher James Reason describes this through the metaphor of slices of Swiss cheese: unless the holes all line up perfectly, there will be no accident. This metaphor provides two lessons: First, do not try to find "the" cause of an accident; Second, we can decrease accidents and make systems more resilient by designing them to have extra precautions against error (more slices of cheese), less opportunities for slips, mistakes, or equipment failure (less holes), and very different mechanisms in the different subparts of the system (trying to ensure that the holes do not line up). (Drawing based upon one by Reason, 1990.)

had been slightly higher, we would not have had the accident. So throw away slice A and replace it." Of course, the same can be said for slices B, C, and D (and in real accidents, the number of cheese slices would sometimes measure in the tens or hundreds). It is relatively easy to find some action or decision that, had it been different, would have prevented the accident. But that does not mean that this was the cause of the accident. It is only one of the many causes: all the items have to line up.

You can see this in most accidents by the "if only" statements. "If only I hadn't decided to take a shortcut, I wouldn't have had the accident." "If only it hadn't been raining, my brakes would have worked." "If only I had looked to the left, I would have seen the car sooner." Yes, all those statements are true, but none of them is "the" cause of the accident. Usually, there is no single cause. Yes, journalists and lawyers, as well as the public, like to know the cause so someone can be blamed and punished. But reputable investigating agencies know that there is not a single cause, which is why their investigations take so long. Their responsibility is to understand the system and make changes that would reduce the chance of the same sequence of events leading to a future accident.

The Swiss cheese metaphor suggests several ways to reduce accidents:

- Add more slices of cheese.
- Reduce the number of holes (or make the existing holes smaller).
- Alert the human operators when several holes have lined up.

Each of these has operational implications. More slices of cheese means mores lines of defense, such as the requirement in aviation and other industries for checklists, where one person reads the items, another does the operation, and the first person checks the operation to confirm it was done appropriately.

Reducing the number of critical safety points where error can occur is like reducing the number or size of the holes in the Swiss cheese. Properly designed equipment will reduce the opportunity for slips and mistakes, which is like reducing the number of holes and making the ones that remain smaller. This is precisely how the safety level of commercial aviation has been dramatically improved. Deborah Hersman, chair of the National Transportation Safety Board, described the design philosophy as:

U.S. airlines carry about two million people through the skies safely every day, which has been achieved in large part through design redundancy and layers of defense.

Design redundancy and layers of defense: that's Swiss cheese. The metaphor illustrates the futility of trying to find the one underlying cause of an accident (usually some person) and punishing the culprit. Instead, we need to think about systems, about all the interacting factors that lead to human error and then to accidents, and devise ways to make the systems, as a whole, more reliable.

When Good Design Isn't Enough

WHEN PEOPLE REALLY ARE AT FAULT

I am sometimes asked whether it is really right to say that people are never at fault, that it is always bad design. That's a sensible question. And yes, of course, sometimes it is the person who is at fault.

Even competent people can lose competency if sleep deprived, fatigued, or under the influence of drugs. This is why we have laws banning pilots from flying if they have been drinking within some specified period and why we limit the number of hours they can fly without rest. Most professions that involve the risk of death or injury have similar regulations about drinking, sleep, and drugs. But everyday jobs do not have these restrictions. Hospitals often require their staff to go without sleep for durations that far exceed the safety requirements of airlines. Why? Would you be happy having a sleep-deprived physician operating on you? Why is sleep deprivation considered dangerous in one situation and ignored in another?

Some activities have height, age, or strength requirements. Others require considerable skills or technical knowledge: people not trained or not competent should not be doing them. That is why many activities require government-approved training and licensing. Some examples are automobile driving, airplane piloting, and medical practice. All require instructional courses and tests. In aviation, it isn't sufficient to be trained: pilots must also keep in practice by flying some minimum number of hours per month.

Drunk driving is still a major cause of automobile accidents: this is clearly the fault of the drinker. Lack of sleep is another major culprit in vehicle accidents. But because people occasionally are at fault does not justify the attitude that assumes they are always at fault. The far greater percentage of accidents is the result of poor design, either of equipment or, as is often the case in industrial accidents, of the procedures to be followed.

As noted in the discussion of deliberate violations earlier in this chapter (page 169), people will sometimes deliberately violate procedures and rules, perhaps because they cannot get their jobs done otherwise, perhaps because they believe there are extenuating circumstances, and sometimes because they are taking the gamble that the relatively low probability of failure does not apply to them. Unfortunately, if someone does a dangerous activity that only results in injury or death one time in a million, that can lead to hundreds of deaths annually across the world, with its 7 billion people. One of my favorite examples in aviation is of a pilot who, after experiencing low oil-pressure readings in all three of his engines, stated that it must be an instrument failure because it was a one-in-a-million chance that the readings were true. He was right in his assessment, but unfortunately, he was the one. In the United States alone there were roughly 9 million flights in 2012. So, a one-in-a-million chance could translate into nine incidents.

Sometimes, people really are at fault.

Resilience Engineering

In industrial applications, accidents in large, complex systems such as oil wells, oil refineries, chemical processing plants, electrical power systems, transportation, and medical services can have major impacts on the company and the surrounding community.

Sometimes the problems do not arise in the organization but outside it, such as when fierce storms, earthquakes, or tidal waves demolish large parts of the existing infrastructure. In either case, the question is how to design and manage these systems so that they can restore services with a minimum of disruption and damage. An important approach is *resilience engineering,* with the goal of designing systems, procedures, management, and the training of people so they are able to respond to problems as they arise. It strives to ensure that the design of all these things—the equipment, procedures, and communication both among workers and also externally to management and the public—are continually being assessed, tested, and improved.

Thus, major computer providers can deliberately cause errors in their systems to test how well the company can respond. This is done by deliberately shutting down critical facilities to ensure that the backup systems and redundancies actually work. Although it might seem dangerous to do this while the systems are online, serving real customers, the only way to test these large, complex systems is by doing so. Small tests and simulations do not carry the complexity, stress levels, and unexpected events that characterize real system failures.

As Erik Hollnagel, David Woods, and Nancy Leveson, the authors of an early influential series of books on the topic, have skillfully summarized:

> *Resilience engineering is a paradigm for safety management that focuses on how to help people cope with complexity under pressure to achieve success. It strongly contrasts with what is typical today—a paradigm of tabulating error as if it were a thing, followed by interventions to reduce this count. A resilient organisation treats safety as a core value, not a commodity that can be counted. Indeed, safety shows itself only by the events that do not happen! Rather than view past success as a reason to ramp down investments, such organisations continue to invest in anticipating the changing potential for failure because they appreciate that their knowledge of the gaps is imperfect and that their environment constantly changes. One measure of resilience is therefore the ability to create foresight—to anticipate the changing shape of risk,*

before failure and harm occurs. (Reprinted by permission of the publishers. Hollnagel, Woods, & Leveson, 2006, p. 6.)

The Paradox of Automation

Machines are getting smarter. More and more tasks are becoming fully automated. As this happens, there is a tendency to believe that many of the difficulties involved with human control will go away. Across the world, automobile accidents kill and injure tens of millions of people every year. When we finally have widespread adoption of self-driving cars, the accident and casualty rate will probably be dramatically reduced, just as automation in factories and aviation have increased efficiency while lowering both error and the rate of injury.

When automation works, it is wonderful, but when it fails, the resulting impact is usually unexpected and, as a result, dangerous. Today, automation and networked electrical generation systems have dramatically reduced the amount of time that electrical power is not available to homes and businesses. But when the electrical power grid goes down, it can affect huge sections of a country and take many days to recover. With self-driving cars, I predict that we will have fewer accidents and injuries, but that when there is an accident, it will be huge.

Automation keeps getting more and more capable. Automatic systems can take over tasks that used to be done by people, whether it is maintaining the proper temperature, automatically keeping an automobile within its assigned lane at the correct distance from the car in front, enabling airplanes to fly by themselves from takeoff to landing, or allowing ships to navigate by themselves. When the automation works, the tasks are usually done as well as or better than by people. Moreover, it saves people from the dull, dreary routine tasks, allowing more useful, productive use of time, reducing fatigue and error. But when the task gets too complex, automation tends to give up. This, of course, is precisely when it is needed the most. The paradox is that automation can take over the dull, dreary tasks, but fail with the complex ones.

When automation fails, it often does so without warning. This is a situation I have documented very thoroughly in my other books and many of my papers, as have many other people in the field of safety and automation. When the failure occurs, the human is "out of the loop." This means that the person has not been paying much attention to the operation, and it takes time for the failure to be noticed and evaluated, and then to decide how to respond.

In an airplane, when the automation fails, there is usually considerable time for the pilots to understand the situation and respond. Airplanes fly quite high: over 10 km (6 miles) above the earth, so even if the plane were to start falling, the pilots might have several minutes to respond. Moreover, pilots are extremely well trained. When automation fails in an automobile, the person might have only a fraction of a second to avoid an accident. This would be extremely difficult even for the most expert driver, and most drivers are not well trained.

In other circumstances, such as ships, there may be more time to respond, but only if the failure of the automation is noticed. In one dramatic case, the grounding of the cruise ship *Royal Majesty* in 1997, the failure lasted for several days and was only detected in the postaccident investigation, after the ship had run aground, causing several million dollars in damage. What happened? The ship's location was normally determined by the Global Positioning System (GPS), but the cable that connected the satellite antenna to the navigation system somehow had become disconnected (nobody ever discovered how). As a result, the navigation system had switched from using GPS signals to "dead reckoning," approximating the ship's location by estimating speed and direction of travel, but the design of the navigation system didn't make this apparent. As a result, as the ship traveled from Bermuda to its destination of Boston, it went too far south and went aground on Cape Cod, a peninsula jutting out of the water south of Boston. The automation had performed flawlessly for years, which increased people's trust and reliance upon it, so the normal manual checking of location or careful perusal of the display (to see the tiny letters "dr" indicating "dead reckoning" mode) were not done. This was a huge mode error failure.

Design Principles for Dealing with Error

People are flexible, versatile, and creative. Machines are rigid, precise, and relatively fixed in their operations. There is a mismatch between the two, one that can lead to enhanced capability if used properly. Think of an electronic calculator. It doesn't do mathematics like a person, but can solve problems people can't. Moreover, calculators do not make errors. So the human plus calculator is a perfect collaboration: we humans figure out what the important problems are and how to state them. Then we use calculators to compute the solutions.

Difficulties arise when we do not think of people and machines as collaborative systems, but assign whatever tasks can be automated to the machines and leave the rest to people. This ends up requiring people to behave in machine like fashion, in ways that differ from human capabilities. We expect people to monitor machines, which means keeping alert for long periods, something we are bad at. We require people to do repeated operations with the extreme precision and accuracy required by machines, again something we are not good at. When we divide up the machine and human components of a task in this way, we fail to take advantage of human strengths and capabilities but instead rely upon areas where we are genetically, biologically unsuited. Yet, when people fail, they are blamed.

What we call "human error" is often simply a human action that is inappropriate for the needs of technology. As a result, it flags a deficit in our technology. It should not be thought of as error. We should eliminate the concept of error: instead, we should realize that people can use assistance in translating their goals and plans into the appropriate form for technology.

Given the mismatch between human competencies and technological requirements, errors are inevitable. Therefore, the best designs take that fact as given and seek to minimize the opportunities for errors while also mitigating the consequences. Assume that every possible mishap will happen, so protect against them. Make actions reversible; make errors less costly. Here are key design principles:

- Put the knowledge required to operate the technology in the world. Don't require that all the knowledge must be in the head. Allow for efficient operation when people have learned all the requirements, when they are experts who can perform without the knowledge in the world, but make it possible for non-experts to use the knowledge in the world. This will also help experts who need to perform a rare, infrequently performed operation or return to the technology after a prolonged absence.
- Use the power of natural and artificial constraints: physical, logical, semantic, and cultural. Exploit the power of forcing functions and natural mappings.
- Bridge the two gulfs, the Gulf of Execution and the Gulf of Evaluation. Make things visible, both for execution and evaluation. On the execution side, provide feedforward information: make the options readily available. On the evaluation side, provide feedback: make the results of each action apparent. Make it possible to determine the system's status readily, easily, accurately, and in a form consistent with the person's goals, plans, and expectations.

We should deal with error by embracing it, by seeking to understand the causes and ensuring they do not happen again. We need to assist rather than punish or scold.