A New Accident Model for Engineering Safer Systems *

Nancy Leveson

Software Engineering Research Laboratory
Aeronautics and Astronautics Dept.
Massachusetts Institute of Technology

Abstract: New technology is making fundamental changes in the etiology of accidents and is creating a need for changes in the explanatory mechanisms used. We need better and less subjective understanding of *why* accidents occur and how to prevent future ones. The most effective models will go beyond assigning blame and instead help engineers to learn as much as possible about *all* the factors involved, including those related to social and organizational structures. This paper presents a new accident model founded on basic systems theory concepts. The use of such a model provides a theoretical foundation for the introduction of unique new types of accident analysis, hazard analysis, accident prevention strategies including new approaches to designing for safety, risk assessment techniques, and approaches to designing performance monitoring and safety metrics.

1 Introduction

Accident models form the basis for investigating and analyzing accidents, preventing future ones, and determining whether systems are suitable for use (risk assessment). In accident investigation they impose patterns on the accidents and influence both the data collected and the factors identified as causative. They also underlie all hazard analysis and risk assessment techniques. Because they influence the factors considered in any of these activities, they may either act as a filter and bias toward considering only certain events and conditions or they may expand activities by forcing consideration of factors that are often omitted.

Most accident models view accidents as resulting from a chain or sequence of events. Such models work well for losses caused by failures of physical components and for relatively simple systems. But since World War II, the types of systems we are attempting to build and the context in which they are being built has been changing. This paper argues that these changes are stretching the limits of current accident models and safety engineering techniques and that new approaches are needed. The changes include:

- **Fast pace of technological change**: Technology is changing faster than the engineering techniques to cope with the new technology are being created. Lessons learned over centuries about designing to prevent accidents may be lost or become ineffective when older technologies

*This paper is abstracted from a book in preparation by the author. The paper is contained in the proceedings of the MIT Engineering Systems Division Internal Symposium, May, 2002. The research involved is partially supported by NSF ITR Grant CCR-0085829 and by a grant from the NASA Intelligent Systems (Human-Centered Computing) Program NCC2-1223

are replaced with new ones. New technology introduces unknowns into our systems and even *unk-unks* (unknown unknowns).

At the same time as the development of new technology has sprinted forward, the time to market for new products has significantly decreased and strong pressures exist to decrease this time even further. The average time to translate a basic technical discovery into a commercial product in the early part of this century was 30 years. Today our technologies get to market in 2-3 years and may be obsolete in 5. We no longer have the luxury of carefully testing systems and designs to understand all the potential behaviors and risks before commercial or scientific use.

- **Changing Nature of Accidents**: Digital technology has created a quiet revolution in most fields of engineering, but system engineering and system safety engineering techniques have not kept pace. Digital systems introduce new "failure modes" that are changing the nature of accidents. Many of the approaches that worked on electromechanical components—such as replication of components to protect against individual component failure (i.e., redundancy)—are ineffective in controlling accidents that arise from the use of digital systems and software. Redundancy may even increase risk by adding complexity. All software problems detected during the flight testing of a NASA experimental aircraft using two versions of the computer-based control system resulted from errors in the redundancy management system added to protect against software errors—the much simpler control software itself worked perfectly [18]. Overconfidence in redundancy and misunderstanding of the failure modes of software-implemented components has played an important role in recent aerospace accidents, such as the loss of the Ariane 5 on its first flight [17].

- **New types of hazards**: The most common accident models are based on an underlying assumption that accidents are the result of an uncontrolled and undesired release of energy or interference in the normal flow of energy. Our increasing dependence on information systems are, however, creating the potential for loss of information or incorrect information that can lead to unacceptable physical, scientific, or financial losses. The "head in the sand" approach of simply denying that software is safety-critical when it only provides information and does not directly release energy (a common attitude in some applications, such as air traffic control) is becoming less and less acceptable as software plays an increasingly important role in accidents.

- **Decreasing tolerance for single accidents**: The losses stemming from accidents is increasing with the cost and potential destructiveness of the systems we build. Our new scientific and technological discoveries have not only created new or increased hazards (such as radiation exposure and chemical pollution) but have provided the means to harm increasing numbers of people as the scale of our systems increases and to impact future generations through environmental pollution and genetic damage. Financial losses and lost potential for scientific advances are also increasing in an age where, for example, a spacecraft may take 10 years and up to a billion dollars to build. Learning from accidents needs to be supplemented with increasing emphasis on preventing the first one.

- **Increasing complexity and coupling**: Complexity has many facets, most of which are increasing in the systems we are building, particularly *interactive* complexity. We are designing systems with potential interactions among the components that cannot be thoroughly planned, understood, anticipated, or guarded against. The operation of some systems is so complex that it defies the understanding of all but a few experts, and sometimes even they

have incomplete information about its potential behavior. Software is an important factor here: it has allowed us to implement more integrated, multi-loop control in systems containing large numbers of dynamically interacting components where tight coupling allows disruptions or dysfunctional interactions in one part of the system to have far-ranging rippling effects. The problem is that we are attempting to build systems that are beyond our ability to intellectually manage: Increased interactive complexity and coupling make it difficult for the designers to consider all the potential system states or for operators to handle all normal and abnormal situations and disturbances safely and effectively. This situation is not new: throughout history, inventions and new technology have often gotten ahead of their scientific underpinnings and engineering knowledge, but the result has always been increased risk and accidents until science and engineering caught up.[1]

- **More complex relationships between humans and automation**: Humans are increasingly sharing control of systems with automation and moving into positions of higher-level decision making with automation implementing the decisions. These changes are leading to new types of human error (such as new types of mode confusion) and a new distribution of human errors (for example, increasing errors of omission versus commission [27, 28]). All human behavior is influenced by the context in which it occurs, and operators in high-tech systems are often at the mercy of the design of the automation they use. Many recent accidents blamed on operator error could more accurately be labeled as resulting from flawed system and interface design. Inadequacies in communication between humans and machines is becoming an increasingly important factor in accidents.

- **Changing regulatory and public views of safety**: In our increasingly complex and interrelated societal structure, responsibility for safety is shifting from the individual to government. Individuals no longer have the ability to control the risks around them and are demanding that government assume greater responsibility for controlling behavior through laws and various forms of oversight and regulation. As companies come under increasing pressure to satisfy time-to-market and budgetary pressures, government will have to step in to provide the protection the public demands. The alternative is individuals and groups turning to the courts for protection, which could have much worse potential effects, such as unnecessarily stifling innovation through fear of law suits.

These changes are challenging both our accident models and the accident prevention and risk assessment techniques based on them. New paradigms are needed.

The next section discusses the limitations of current event-based models and presents the goals for an improved model. Then the new model is presented along with a classification of accident causal factors derived from the model. The final section discusses the implications of the new model for accident analysis, accident prevention, risk assessment, and performance monitoring.

[1] As an example, consider the introduction of high-pressure steam engines in the first half of the nineteenth century, which transformed industry and transportation but resulted in frequent and disastrous explosions. While engineers quickly amassed scientific information about thermodynamics, the action of steam in the cylinder, the strength of materials in the engine and many other aspects of steam engine operation, there was little scientific understanding about the buildup of steam pressure in the boiler, the effect of corrosion and decay, and the causes of boiler explosions. High-pressure steam had made the current boiler design obsolete by producing excessive strain on the boilers and exposing weaknesses in the materials and construction of the boilers. Attempts to add technological safety fixes were unsuccessful because engineers did not fully understand what went on in steam boilers: It was not until well after the mid-century that the dynamics of steam generation was understood. For an examination of the parallels between the early development of high-pressure steam engines and software engineering, see [15].

2 Limitations of Event Chain Models

Event-based accident models explain accidents in terms of multiple events sequenced as a chain over time. The events considered almost always involve some type of component failure, human error, or energy-related event. The chains may be branching or there may be multiple chains synchronized using time or common events [3]. Forward sequences (as in FMEA or Events Trees) or backward ones (as in Fault Trees) may be used. Other relationships may be represented by the chain in addition to a chronological one, but any such relationship is almost always a direct, linear one. As such, event-based models encourage limited notions of causality—usually linear causality relationships are emphasized—and it is difficult to incorporate non-linear relationships, including feedback. In addition, some important causal factors are difficult to fit into simple event models. For example, studies have found that the most important factor in the occurrence of accidents is management commitment to safety and the basic safety culture in the organization or industry.

In event-based models, the causal factors identified depend on the events that are considered and the selection of the conditions related to those events. However, other than the physical events immediately preceding or directly involved in the loss, the choice of events to include is subjective and the selection of conditions to explain the events is even more so.

Although the first event in the chain is often labeled the "initiating event," the selection of an initiating event is arbitrary and previous events and conditions could always be added. This subjectivity in selection of a stopping point in a backward event chain means that the assignment of a "root cause" for an accident is a purely pragmatic question regarding the stopping rule applied for analysis after the fact—there is no well-defined "start" of the causal chain involved in accidents.

Why does the lack of a well-defined stopping point matter? There are two basic reasons for conducting an accident investigation: (1) to assign blame for the accident and (2) to understand why it happened so that future accidents can be prevented. When the goal is to assign blame, the backward chain of events considered often stops when someone or something appropriate to blame is found. As a result, an analysis based on such a model may provide too superficial an explanation of why the accident occurred to prevent similar losses.

When learning how to engineer safer systems is the goal rather than identifying who to punish, then the emphasis in accident analysis needs to shift from "cause" (which has a limiting, blame orientation) to understanding accidents in terms of reasons, i.e., why the events and errors occurred. In an analysis by the author of recent aerospace accidents involving software in some way, most of the reports stopped after assigning blame (usually to the operators) and never got to the root of why the accident occurred—for example, why the operators made the errors they did and how to prevent such errors in the future or why the software requirements error was made and why it was not detected and fixed before the software was used [17].

Event chains developed to explain the accident usually concentrate on the proximate events immediately preceding the loss. But the foundation for an accident is often laid years before. One event simply triggers the loss, but if that event had not happened, another one would have. In the Bhopal disaster, for example, cost cutting and political pressures by Union Carbide and its Indian subsidiary resulted in eliminating refrigeration, putting off maintenance, reducing the workforce, changing worker shift replacement policies, etc., all of which led to the worst industrial accident in history. This degradation in the safety margin occurred over time and without any particular single decision to do so but simply as a series of decisions that moved the plant slowly toward a situation where any slight error would lead to a major accident. Given the overall state of the Bhopal Union Carbide plant and its operation, if the slip disk had not been left out of the pipe washing operation that December day in 1984, something else would have triggered an accident. In fact, a similar leak had occurred the year before, but did not have the same catastrophic consequences. To identify

one event (such as a maintenance worker leaving out the slip disk) or even several events as the root cause or the start of an event chain leading to this accident would be misleading at best.

Rasmussen writes:

> The stage for an accidental course of events very likely is prepared through time by the normal efforts of many actors in their respective daily work context, responding to the standing request to be more productive and less costly. Ultimately, a quite normal variation in somebody's behavior can then release an accident. Had this 'root cause' been avoided by some additional safety measure, the accident would very likely be released by another cause at another point in time. In other words, an explanation of the accident in terms of events, acts, and errors is not very useful for design of improved systems.

In addition to subjectivity in selecting the events and stopping point, the links between the events that are chosen to explain the "cause" are subjective and subject to bias. In the loss of an American Airlines B-757 near Cali, Colombia, in 1995, two significant events were (1) *Pilot asks for clearance to take the Rozo approach* followed later by (2) *Pilot types R into the FMS.*[2] In fact, the pilot should have typed the four letters *ROZO* instead of *R*—the latter was the symbol for a different radio beacon (called Romeo) near Bogota—and as a result the aircraft incorrectly turned toward mountainous terrain. While these events are noncontroversial, the link between the two events could be explained by any of the following:

- *Crew Procedure Error*: In the rush to start the descent, the captain entered the name of the waypoint without normal verification from the other pilot.
- *Pilot Error*: In the rush to start the descent, the pilot executed a change of course without verifying its effect on the flight path.
- *Approach Chart and FMS Inconsistencies*: The identifier used to identify Rozo on the approach chart (*R*) did not match the identifier used to call up Rozo in the FMS.
- *FMS Design Deficiency*: The FMS did not provide the pilot with feedback that choosing the first identifier listed on the display was not the closest beacon with that identifier.
- *American Airlines Training Deficiency*: The pilots flying into South America were not warned about duplicate beacon identifiers or adequately trained on the logic and priorities used in the FMS on the aircraft.
- *Manufacturers' Deficiencies*: Jeppesen-Sanderson did not inform airlines operating FMS-equipped aircraft of the differences between navigation information provided by Jeppesen-Sanderson FMS navigation databases and Jeppesen-Sanderson approach charts or the logic and priorities employed in the display of electronic FMS navigation information.
- *International Standards Deficiency*: No single worldwide standard provides unified criteria for the providers of electronic navigation databases used in flight management systems.

The selection of one of these linking conditions will greatly influence the "cause" ascribed to the accident yet all are plausible and each fully explains (according to formal logic) the event sequence. Choosing only one may reflect more on the person or group making the selection than on the

[2] An FMS is an automated Flight Management System, which assists the pilots in various ways. In this case, it was being used to provide navigation information.

accident itself. In fact, understanding this accident and learning enough from it to prevent future accidents requires identifying all these factors: The accident model used should encourage and guide such a comprehensive analysis. Operators, managers, engineers, and regulatory agencies may all have different views of the flawed processes underlying an accident, depending on their perspective and the role they play in the overall socio-technical system. All of these views should be represented in the accident analysis; at the same time, the factual data should be separated from the interpretation of that data.

The countermeasures to prevent accidents considered as chains of events usually involve either removing events or conditions or adding enough AND gates (required simultaneous conditions or events) that the likelihood of the chaining factors being realized is very low, that is, the accident sequence is broken. Because the most common events considered in event-based models are component failures, engineering design has focused to a great extent on preventing such failures, i.e., increasing component integrity, and on adding redundancy (AND gates) to reduce their likelihood of leading to a loss.

This focus on failure events and the reliability engineering techniques to prevent them, however, does not account for (1) social and organizational factors in accidents, (2) system accidents and software errors, (3) human error, and (4) adaptation over time.

2.1 Social and Organizational Factors

Event-based models are poor at representing systemic accident factors such as structural deficiencies in the organization, management deficiencies, and flaws in the safety culture of the company or industry. An accident model should encourage a broad view of accident mechanisms that expands the investigation from beyond the proximate events.

Ralph Miles Jr., in describing the basic concepts of systems theory, noted that:

> Underlying every technology is at least one basic science, although the technology may be well developed long before the science emerges. Overlying every technical or civil system is a social system that provides purpose, goals, and decision criteria. [19, p.1]

Effectively preventing accidents in complex systems requires using accident models that include that social system as well as the technology and its underlying science. Without understanding the purpose, goals, and decision criteria used to construct and operate systems, it is not possible to completely understand and most effectively prevent accidents.

2.2 System Accidents and Software Errors

Since World War II, we are increasingly experiencing a new type of accident that arises in the interactions *among* components (electromechanical, digital, and human) rather than in the failure of individual components. Perrow coined the term *system accident* to describe it [21]. In contrast, accidents arising from component failures, including the possibility of multiple and cascading failures, might be termed *component failure accidents*.

Accidents resulting from *dysfunctional interactions* among system components (system accidents) have received less attention than component failure accidents. This lack of concern may stem partly from the fact that in the simpler systems of the past, analysis and testing allowed exercising the system to detect all potential undesired interactions and changing the system design to eliminate them. Increasing complexity and the introduction of software control is reducing this ability and increasing the incidence of system accidents. System accidents can be explained in terms of inadequate control over component interactions, and prevention requires reducing or

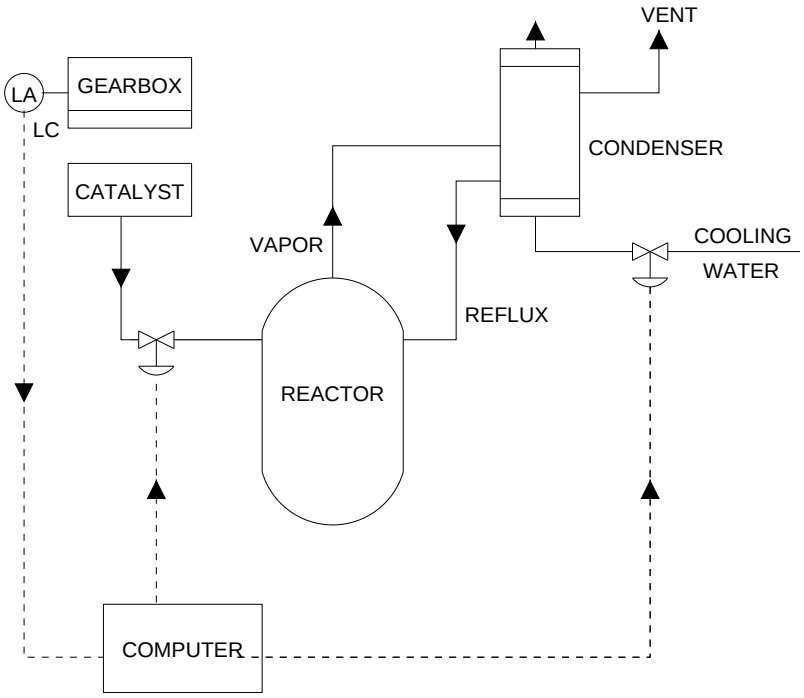

Figure 1: A chemical reactor design (adapted from Kletz [13, p.6]).

eliminating dysfunctional interactions, i.e., interactions that can lead to hazardous states in the controlled process. A taxonomy and classification of the types of dysfunctional interactions leading to accidents is presented below (see Section 3.4).

The Ariane 5 and Mars Polar Lander losses are examples of system accidents. In both of these accidents, the components did not fail in terms of not satisfying their specified requirements. The individual components operated exactly the way the designers had planned—the problems arose in the unplanned or misunderstood effects of these component behaviors on the system as a whole, that is, errors in the system design rather than the component design, including errors in allocating and tracing the system functions to the individual components. The solution, therefore, lies in system engineering.

Consider an example of a system accident that occurred in a batch chemical reactor in England [12]. The design of this system is shown in Figure 1. The computer was responsible for controlling the flow of catalyst into the reactor and also the flow of water into the reflux condenser to cool off the reaction. Additionally, sensor inputs to the computer were supposed to warn of any problems in various parts of the plant. The programmers were told that if a fault occurred in the plant, they were to leave all controlled variables as they were and to sound an alarm. On one occasion, the computer received a signal indicating a low oil level in a gearbox. The computer reacted as its requirements specified: It sounded an alarm and left the controls as they were. By coincidence, a catalyst had been added to the reactor, but the computer had just started to increase the cooling-water flow to the reflux condenser; the flow was therefore kept at a low rate. The reactor overheated, the relief valve lifted, and the contents of the reactor were discharged into the atmosphere.

Note that there were no component failures involved in this accident: the individual components, including the software, worked as specified but together they created a hazardous system state.

Merely increasing the reliability of the individual components or protecting against their failure would not have prevented the loss. Prevention required identifying and eliminating or mitigating unsafe interactions among the system components.

Most software-related accidents have been system accidents—they stem from the operation of the software, not from its *lack* of operation and usually that operation is exactly what the software engineers intended. Thus event models as well as system design and analysis methods that focus on classic types of failure events will not apply to software. Confusion about this point is reflected in the many fault trees containing useless (and misleading) boxes that say *"Software Fails."* Software is the design of a machine abstracted from its physical realization, for example, the logical design of an autopilot separated from any physical design to implement that logic in hardware. What does it mean to talk about an abstraction or a design failing? A better way to understand the role of software in accidents is described later in this paper (see Section 3.1).

2.3 Human Error

Human error is usually defined as any deviation from the performance of a specified or prescribed sequence of actions. However, instructions and written procedures are almost never followed exactly as operators strive to become more efficient and productive and to deal with time pressures. In fact, a common way for workers to apply pressure to management without actually going out on strike is to "work to rule," which can lead to a breakdown in productivity and even chaos.

In studies of operators, even in such highly constrained and high-risk environments as nuclear power plants, modification of instructions is repeatedly found and the violation of rules appears to be quite rational, given the actual workload and timing constraints under which the operators must do their job [10, 30, 32]. In these situations, a basic conflict exists between error as seen as a deviation from the *normative procedure* and error as seen as a deviation from the rational and normally used *effective procedure* [24].

One implication is that following an accident, it will be easy to find someone involved in the dynamic flow of events that has violated a formal rule by following *established practice* rather than *specified practice*. Given the frequent deviation of established practice from normative work instructions and rules, it is not surprising that operator "error" is found to be the cause of 70-80% of accidents.

Most decisions are sound using a local judgement criterion and given the time and budget pressures and short-term incentives that shape behavior. Experts do their best to meet local conditions and in the busy daily flow of activities are unaware of any potentially dangerous side effects. Each individual decision may appear safe and rational within the context of the individual work environments and local pressures, but may be unsafe when considering the larger socio-technical system as a whole: It is difficult if not impossible for any individual to judge the safety of their decisions when it is dependent on the decisions made by other people in other departments and organizations.

Traditional decision theory research perceives decisions as discrete processes that can be separated from the context and studied as an isolated phenomenon. More recent research has taken a very different approach: Instead of thinking of operations as predefined sequences of actions, human interaction with a system is increasingly being considered to be a continuous control task in which separate "decisions" or errors are difficult to identify. Edwards, back in 1962, was one of the first to argue that decisions can only be understood as part of an ongoing process [9]. The state of the system is perceived in terms of possible actions, one of these actions is chosen, and the resulting response from the controlled system acts as a background for the next action. Errors then are difficult to localize in the stream of behavior; *the effects of less successful actions are a natural part*

of the search on the part of the operator for optimal performance. Not only are separate decisions difficult to identify in this model of human control, but the study of decision making then cannot be separated from a simultaneous study of the social context, the value system in which it takes place, and the dynamic work process it is intended to control [22, 23, 31]. This view is the foundation of *dynamic decision making* [4] and the new field of *naturalistic decision making* [34, 20].

As argued by Rasmussen and many others, devising more effective accident models will require shifting the emphasis in explaining the role of humans in accidents from error (deviations from normative procedures) to focus on the mechanisms and factors that shape human behavior, i.e., the performance-shaping mechanisms and context in which human actions take place and decisions are made. Effective approaches to understanding the role of humans in safety must look at the goals and motives behind human behavior. Models are needed that account for the complex role that human decisions and behavior are playing in the accidents occurring in high-tech systems and that handle not simply individual decisions or even sequences of decisions, but the overall decision-makig process and the interactions among decisions by multiple, interacting decision makers.

2.4 Adaptation

Any accident model that includes the social system and human error must account for adaptation. To paraphrase a familiar saying, the only constant is that nothing ever remains constant. Systems and organizations continually experience change as adaptations are made in response to local pressures and short-term productivity and cost goals. People adapt to their environment or they change their environment to better suit their purposes. A corollary of this propensity for systems and people to adapt over time is that safety defenses are likely to degenerate systematically through time, particularly when pressure toward cost-effectiveness and increased productivity is the dominant element in decision making. Thus, the redundancy and other precautions added to protect against human error often degenerate over time as work practices adapt to increase efficiency within the local environment. The critical factor here is that such adaptation is not a random process—it is an optimization process depending on search strategies—and thus should be predictable and potentially controllable [23].

Woods has stressed the importance of adaptation in accidents. He describes organizational and human failures as breakdowns in adaptations directed at coping with complexity, and accidents as involving a "drift toward failure as planned defenses erode in the face of production pressures and change" [33]. Similarly, Rasmussen has argued that major accidents are often caused not by a coincidence of independent failures but instead reflect a systematic migration of organizational behavior to the boundaries of safe behavior under pressure toward cost-effectiveness in an aggressive, competitive environment [23]. The Bhopal accident, described earlier, is one example of this type of systematic migration toward an accident, but most accidents in complex socio-technical systems exhibit this same behavior. One implication of this viewpoint is that the struggle for a good safety culture will never end because it must fight against the functional pressures of the environment. Improvement of the safety culture will therefore require an analytical approach directed toward the behavior-shaping factors in the environment [23].

Humans and organizations can adapt and still maintain safety as long as they stay within the area bounded by safety constraints. But in the search for optimal operations, humans and organizations will usually close in on and explore the boundaries of established practice, and such exploration implies the risk of occasionally crossing the limits of safe practice unless the constraints on safe behavior are enforced.

For an accident model to handle system adaptation over time, it must consider the processes involved in accidents and not simply events and conditions: Processes control a sequence of events

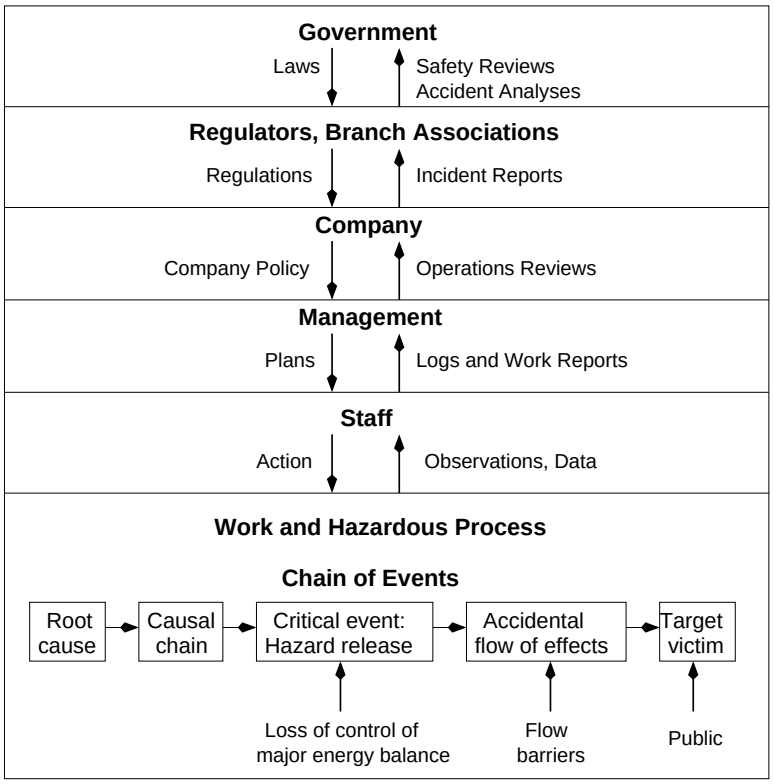

Figure 2: Rasmussen and Svedung socio-technical model of system operations. Adapted from [23, 25]

and describe system and human behavior over time rather than considering events and human actions individually. As Rasmussen argues, accident causation must be viewed as a complex *process* involving the entire socio-technical system including legislators, government agencies, industry associations and insurance companies, company management, technical and engineering personnel, operations, etc.

The idea of modeling socio-technical systems using process-control concepts is not a new one. Jay Forrester in the 1960s, for example, created *system dynamics* using such an approach. Industrial engineering models often include both the management and technical aspects of systems. As one example, Johansson [29] describes a production system as four subsystems: physical, human, information, and management. The physical subsystem includes the inanimate objects—equipment, facilities, and materials. The human subsystem controls the physical subsystem. The information subsystem provides flow and exchange of information that authorizes activity, guides effort, evaluates performance, and provides overall direction. The organizational and management subsystem establishes goals and objectives for the organization and its functional components, allocates authority and responsibility, and generally guides activities for the entire organization and its parts.

Rasmussen and Svedung have described a hierarchical model of the socio-technical system involved in risk management (see Figure 2) [25]. At the social and organizational levels of their model, Rasmussen and Svedung use a pure control-based model, and at all levels they focus on information flow. At the lowest, technical level, however, they revert from a control theory model back to a chain-of-events model. In addition, they focus on the downstream part of the chain following the

occurrence of the hazard. This downstream emphasis is common in the process industry, where Rasmussen has done most of his work.

The new model introduced in the next section builds on the ideas used in the upper levels of the Rasmussen-Svedung model, but it continues the control-theoretic approach down through and including the technical system and its development and operations. In addition, more emphasis is placed (1) on the upstream process, i.e., in preventing the occurrence of the hazardous state, (2) on the system development process, and (3) on the components of control beyond information flow. Perhaps most important, the model in this paper allows classifying the specific factors involved in accidents. This classification can be used in accident analysis, accident prevention, and risk assessment.

3 An Accident Model Based on Systems Theory

The hypothesis underlying the new model, called STAMP (Systems Theory Accident Modeling and Processes) is that system theory is a useful way to analyze accidents, particularly system accidents. In this conception of safety, accidents occur when external disturbances, component failures, or dysfunctional interactions among system components are not adequately handled by the control system, that is, they result from inadequate control or enforcement of safety-related constraints on the development, design, and operation of the system.

Safety then can be viewed as a control problem, and safety is managed by a control structure embedded in an adaptive socio-technical system. The goal of the control structure is to enforce constraints on system development (including both the development process itself and the resulting system design) and on system operation that result in safe behavior. In this framework, understanding why an accident occurred requires determining why the control structure was ineffective. Preventing future accidents requires designing a control structure that will enforce the necessary constraints.

In STAMP, systems are viewed as interrelated components that are kept in a state of dynamic equilibrium by feedback loops of information and control. A system in this conceptualization is not a static design—it is a dynamic process that is continually adapting to achieve its ends and to react to changes in itself and its environment. The original design must not only enforce appropriate constraints on behavior to ensure safe operation, but the system must continue to operate safely as changes occur. The process leading up to an accident (loss event) can be described in terms of an adaptive feedback function that fails to maintain safety as performance changes over time to meet a complex set of goals and values.

Instead of defining safety management in terms of preventing component failure events, it is defined as a continuous control task to impose the constraints necessary to limit system behavior to safe changes and adaptations. Accidents can be understood, using this model, in terms of why the controls that were in place did not prevent or detect maladaptive changes, that is, by identifying the safety constraints that were violated and determining why the controls were inadequate in enforcing them.

The basic concepts in STAMP are constraints, control loops and process models, and levels of control. Each of these is now described followed by a classification of accident factors based on the new model and on basic systems theory concepts.

3.1 The Central Role of Constraints in System Safety

The most basic concept in the new model is not an event, but a *constraint*. In systems theory, control is always associated with the imposition of constraints. The cause of an accident, instead

of being understood in terms of a series of events, is viewed as the result of a lack of constraints imposed on the system design and on operations, that is, by inadequate enforcement of constraints on behavior at each level of a socio-technical system. In systems theory terminology, safety is an emergent property that arises when the system components interact within an environment. Emergent properties are controlled or enforced by a set of constraints (control laws) related to the behavior of the system components. Accidents result from a lack of appropriate constraints on the interactions.

As an example, the unsafe behavior (hazard) in the Challenger loss was the release of hot propellant gases from the field joint. The miscreant O-ring was used to control the hazard, i.e., its role was to seal a tiny gap in the field joint created by pressure at ignition. The design, in this case, did not effectively impose the required constraint on the propellant gas release. Starting from here, there are then several questions that need to be answered to understand why the accident occurred. Why was this particular design unsuccessful in imposing the constraint, why was it chosen (what was the decision process), why was the flaw not found during development, and was there a different design that might have been more successful? These questions and others consider the original design process.

Understanding the accident also requires examining the contribution of the operations process. One constraint that was violated during operations was the requirement to correctly handle feedback about any potential violation of the safety design constraints, in this case, feedback during operations that the control by the O-rings of the release of hot propellant gases from the field joints was not being adequately enforced by the design. There were several instances of feedback that was not adequately handled, such as data about O-ring blowby and erosion during previous shuttle launches and feedback by engineers who were concerned about the behavior of the O-rings in cold weather. In addition, there was missing feedback about changes in the design and testing procedures during operations, such as the use of a new type of putty and the introduction of new O-ring leak checks without adequate verification that they satisfied system safety constraints on the field joints. As a final example, the control processes were flawed that ensured unresolved safety concerns were adequately considered before each flight, i.e., flight readiness reviews and other feedback channels to project management making flight decisions.

Why do design constraints play such an important role in complex systems, particularly software-intensive systems? The computer is so powerful and so useful because it has eliminated many of the physical constraints of electromechanical devices. This is both its blessing and its curse: We do not have to worry about the physical realization of our software designs, but we also no longer have physical laws the limit the complexity of these designs. I call this the *curse of flexibility*. Physical constraints enforce discipline on the design, construction, and modification of our design artifacts. Physical constraints also control the complexity of what we build. With software, the limits of what is *possible* to accomplish are different than the limits of what can be accomplished *successfully* and *safely*—the limiting factors change from the structural integrity and physical constraints of our materials to limits on our intellectual capabilities. It is possible and even quite easy to build software that we cannot understand in terms of being able to determine how it will behave under all conditions: We can construct software (and often do) that goes beyond human intellectual limits. The result has been an increase in system accidents stemming from intellectual unmanageability related to interactively complex and tightly coupled designs that allow potentially unsafe interactions to go undetected during development.

The solution to this problem is for engineers to enforce the same discipline on the software parts of the system design that nature imposes on the physical parts. Safety, like any quality, must be built into the system design. When software acts as a controller in complex systems, it represents or *is* the system design—it embodies or enforces the system safety constraints by controlling the

components and their interactions. Control software, then, contributes to an accident by not enforcing the appropriate constraints on behavior or by commanding behavior that violates the constraints. In the batch reactor example of Section 2.2, the software needed to enforce the system safety constraint that water must be flowing into the reflux condenser whenever the flow of catalyst to the reactor is initiated. This system behavioral constraint translates to a constraint on software behavior (a software requirement) that the software must always open the water valve before the catalyst valve.

This control model provides a much better description of how software affects accidents than a failure model. The primary safety problem in computer-controlled systems is not software "failure" but the lack of appropriate constraints on software behavior, and the solution is to identify the required constraints and enforce them in the software and overall system design. System engineers must identify the constraints necessary to ensure safe system behavior and effectively communicate these behavioral constraints to the software engineers who, in turn, must enforce them in their software.

The relaxation of physical constraints also impacts human supervision and control of automated systems and the design of interfaces between operators and controlled processes [8]. Cook argues that when controls were primarily mechanical and were operated by people located close to the operating process, proximity allowed sensory perception of the status of the process via direct physical feedback such as vibration, sound, and temperature. Displays were directly linked to the process and thus were essentially a physical extension of it. For example, the flicker of a gauge needle in the cab of a train indicated (1) the engine valves were opening and closing in response to slight pressure fluctuations, (2) the gauge was connected to the engine, (3) the pointing indicator was free, etc. In this way, the displays provided a rich source of information about the controlled process and the state of the displays themselves.

The introduction of electromechanical controls allowed operators to control the process from a greater distance (both physical and conceptual) than possible with pure mechanically linked controls. That distance, however, meant that operators lost a lot of direct information about the process—they could no longer sense the process state directly and the control and display surfaces no longer provided as rich a source of information about it (or the state of the controls themselves). The designers had to synthesize and provide an image of the process state to the operators. An important new source of design errors was the need for the designers to determine beforehand what information the operator would need under all conditions to safely control the process. If the designers had not anticipated a particular situation could occur and provided for it in the original system design, they might also not anticipate the need of the operators for information about it during operations.

Designers also had to provide feedback on the actions of the operators and on any failures that might have occurred. The controls could now be operated without the desired effect on the process, and the operators might not know about it. Accidents started to occur due to incorrect feedback. For example, major accidents (including Three Mile Island) have involved the operators commanding a valve to open and receiving feedback that the valve had opened as a result, when in reality it had not. In these cases, the valves were wired to provide feedback that power had been applied to the valve, but not that it had actually opened. Not only could the design of the feedback about failures be misleading, but the return links were also subject to failure themselves.

Thus, electromechanical controls relaxed constraints on the system design allowing greater functionality. At the same time, they created new possibilities for designer and operator error that had not existed or were much less likely in mechanically controlled systems. The later introduction of computer and digital controls afforded additional advantages and removed even more constraints on the control system design—and introduced more possibility for error. It is this freedom from

constraints that makes the design of such systems so difficult. The constraints shaped the system design in ways that efficiently transmitted valuable physical process information and supported the operators' cognitive processes. Proximity provided rich sources of feedback that involved almost all of the senses, enabling early detection of potential problems. We are finding it hard to capture and provide these same qualities in new systems that use computer controls and displays.

In summary, accidents result from a lack of appropiate constraints on system design. The role of the system engineer or system safety engineer is to identify the design constraints necessary to maintain safety and to ensure that the system design, including the social and organizational aspects of the system and not just the physical ones, enforces them.

3.2 Control Loops and Process Models

Instead of decomposing systems and accident explanations into structural components and a flow of events as do most event-based models, an accident model based on system theory describes systems and accidents in terms of a hierarchy of control based on adaptive feedback mechanisms. Some basic concepts from systems theory are needed here.

In system theory, open systems are viewed as interrelated components that are kept in a state of dynamic equilibrium by feedback loops of information and control. The plant's overall performance has to be controlled in order to produce the desired product while satisfying cost and quality constraints. In general, to effect control over a system requires four conditions [1]: (1) the controller must have a goal or goals (e.g., to maintain the set point), (2) the controller must be able to affect the state of the system, (3) the controller must be (or contain) a model of the system, (4) the controller must be able to ascertain the state of the system.

Figure 3 shows a typical control loop where an automated controller is supervised by a human controller. The dotted lines indicate that the human supervisor may have direct access to system state information (not provided by the computer) and may have ways to manipulate the controlled process other than through computer commands.

The human and/or automated controller(s) obtains information about (observes) the process state from measured variables (*feedback*) and uses this information to initiate action by manipulating *controlled variables* to keep the process operating within predefined limits (constraints) or *set points* (the goal) despite disturbances to the process. In general, the maintenance of any open-system hierarchy, either biological or man-made, will require a set of processes in which there is communication of information for regulation or control [6].

Control actions will, in general, lag in their effects on the process because of delays in signal propagation around the control loop: an actuator may not respond immediately to an external command signal (called *dead time*); the process may have delays in responding to manipulated variables (*time constants*); and the sensors may obtain values only at certain sampling intervals (*feedback delays*). Time lags restrict the speed and extent with which the effects of disturbances (both within the process itself and externally derived) can be reduced and impose extra requirements on the controller, for example, the need to infer delays that are not directly observable.

Any controller—human or automated—must contain a model of the system being controlled (see condition 3 above) [7]. This model at one extreme may contain only one or two variables (such as that required for a simple thermostat) while at the other extreme it may require a complex model with a large number of state variables and transitions (such as that needed for air traffic control). Whether the model is embedded in the control logic of an automated controller or in the mental model maintained by a human controller, it must contain the same type of information: the required relationship among the system variables (the control laws), the current state (the current values of the system variables), and the ways the process can change state. This model is used to

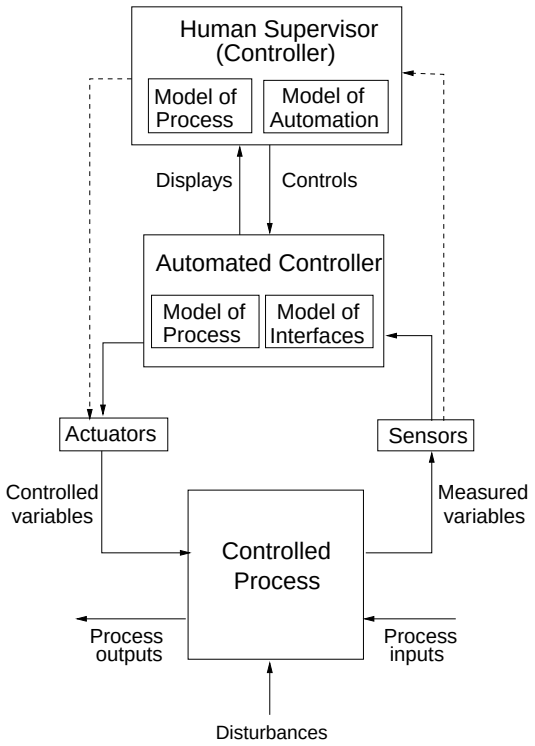

Figure 3: A typical control loop and the process models involved.

determine what control actions are needed, and it is updated through various forms of feedback.

Human controllers interacting with automated controllers, in addition to having a model of the controlled process, must also have a model of the automated controllers' behavior in order to monitor or supervise it (Figure 3). Accidents may result from inaccuracies in this model. In the loss of the American Airlines B-757 near Cali, Colombia, the pilots did not understand the model used by the computer for labeling waypoints. In the Nagoya A320 accident, the pilots' mental models of the automation behavior did not match the automation design. Unfortunately, surveys and studies are finding that many operators of high-tech systems do not understand how the automation works (for example, [5]).

There may, of course, be multiple human and automated controllers in the control loop, and computers may be in other parts of the control loop than shown in Figure 3. For example, computers may act as automated decision aids that provide information to the human controller but do not directly issue control commands to the process actuators: If the software provides decision aiding, however, it is indirectly controlling the process and it must contain a model of the process. Common arguments that in this design the software is not safety-critical are not justified—it is still a critical part of the functioning of the control loop and software errors can lead to accidents.

This discussion has been simplified by speaking only of process models. Models will also need to include the relevant properties of the sensors, actuators, and on occasion the environment. An example is the need for an automated controller to have a model of its interface to the human controller(s) or supervisor(s). This interface, which contains the controls, displays, alarm annunciators, etc., is important because it is the means by which the two controller's models are synchronized, and lack of synchronization between the models can lead to system accidents.

3.3 Socio-Technical Levels of Control

In systems theory, systems are viewed as hierarchical structures where each level imposes constraints on the activity of the level beneath it—that is, constraints or lack of constraints at a higher level allow or control lower-level behavior [6]. Control laws are constraints on the relationships between the values of system variables. Safety-related control laws or constraints therefore specify those relationships between system variables that constitute the nonhazardous system states, for example, the power must never be on when the access door is open. The control processes (including the physical design) that enforce these constraints will limit system behavior to safe changes and adaptations.

Modeling complex organizations or industries using system theory involves dividing them into hierarchical levels with control processes operating at the interfaces between levels [23]. Figure 4 shows a generic socio-technical control model. Each system, of course, must be modeled to reflect its specific features. The model is similar to the one devised by Rasmussen but his model contains only one control structure and his focus is on operations. The model in Figure 4 has two basic hierarchical control structures—one for system development (on the left) and one for system operation (on the right)—with interactions between them. An aircraft manufacturer, for example, might only have system development under its immediate control, but safety involves both development and operational use of the aircraft, and neither can be accomplished successfully in isolation: Safety must be designed into the system, and safety during operation depends partly on the original design and partly on effective control over operations. Manufacturers must communicate to their customers the assumptions about the operational environment upon which the safety analysis was based, as well as information about safe operating procedures. The operational environment in turn provides feedback to the manufacturer about the performance of the system during operations.

Between the hierarchical levels of each control structure, effective communications channels are needed, both a downward *reference channel* providing the information necessary to impose constraints on the level below and an upward *measuring channel* to provide feedback about how effectively the constraints were enforced. Feedback is critical in any open system in order to provide adaptive control. At each level, inadequate control may result from missing constraints, inadequately communicated constraints, or from constraints that are not enforced correctly at a lower level.

The top two levels of each of the two generic control structures are government and general industry groups. The government control structure in place to control development may differ from that controlling operations—a different group at the FAA, for example, is responsible for issuing aircraft type certifications than that responsible for supervising airline operations. The appropriate constraints in each control structure and at each level will vary but in general may include technical design and process constraints, management constraints, manufacturing constraints, and operational constraints.

At the highest level in both the system development and system operation hierarchies are Congress and state legislatures.[3] Congress controls safety by passing laws and by establishing and funding government regulatory structures. Feedback as to the success of these controls or the need for additional ones comes in the form of government reports, congressional hearings and testimony, lobbying by various interest groups, and, of course, accidents.

The next level contains government regulatory agencies, industry associations, user associations, insurance companies, and the court system. Unions may play a role in ensuring safe system operations (such as the air traffic controllers union) or worker safety in manufacturing. The legal

[3]Obvious changes are required in the model for countries other than the U.S. The U.S. is used in the example here because of the author's familiarity with it.

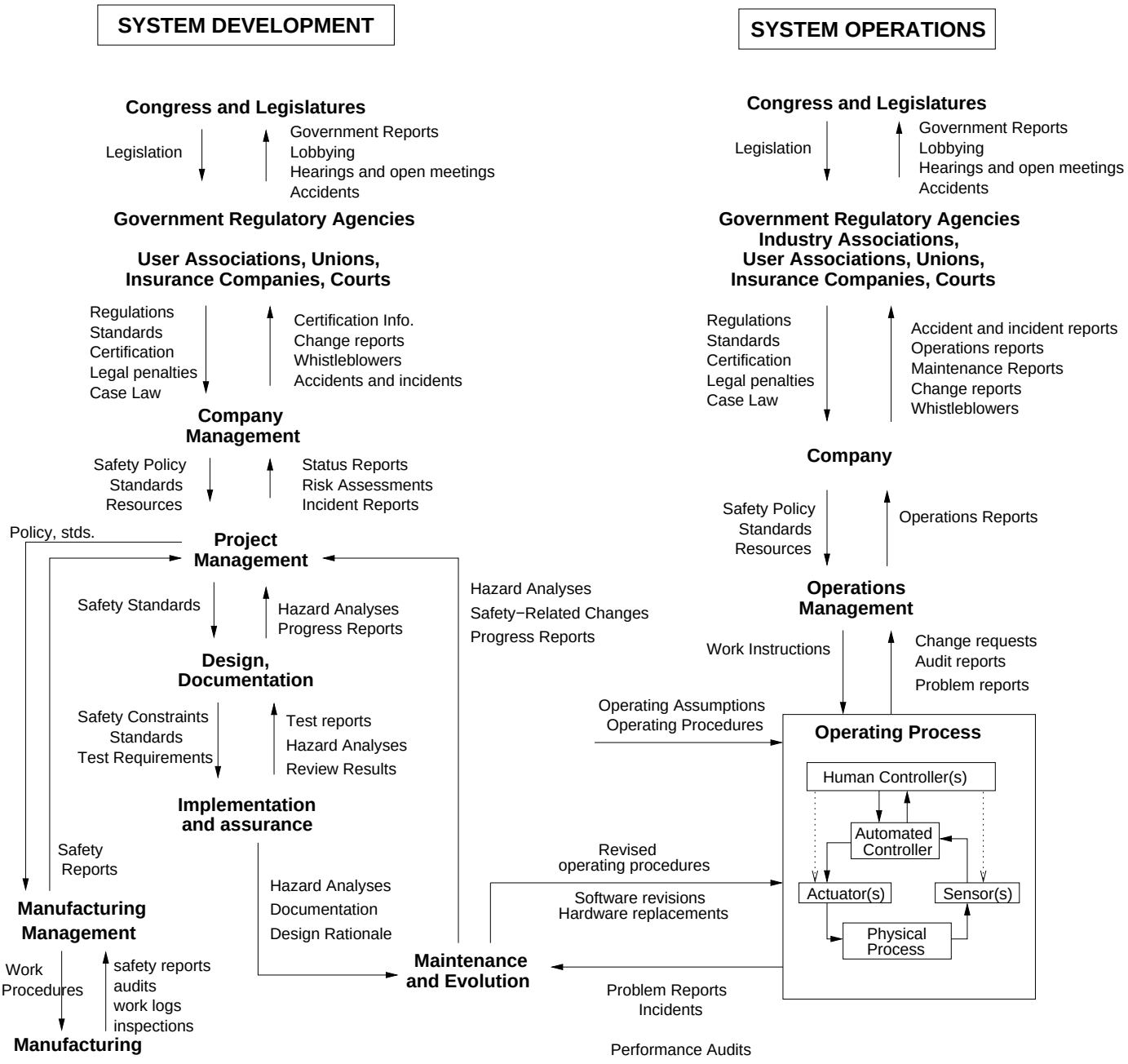

Figure 4: General Form of a Model of Socio-Technical Control.

system tends to be used when there is no regulatory authority and the public has no other means to encourage a desired level of concern for safety in company management. The constraints generated at this level and passed down to the companies are usually in the form of policy, regulations, certification, standards (by trade or user associations), or threat of litigation. Where there is a union, safety-related constraints on operations or manufacturing may result from union demands and collective bargaining.

In the development control structure (shown on the left), constraints imposed on behavior by government and other entities must be reflected in the design of company safety policy, standards, and allocation of resources. Recent trends from management by *oversight* to management by *insight* reflect differing levels of feedback control exerted over the lower levels and a change from prescriptive management control to management by objectives, where the objectives are interpreted and satisfied according to the local context [23]. Attempts to delegate decisions and to manage by objectives requires an explicit formulation of the value criteria to be used and an effective means for communicating the values down through society and organizations. The impact of specific decisions at each level on the objectives and values passed down need to be adequately and formally evaluated. While some generic functions will be required at a particular level to avoid accidents, the details about how the functions will be accomplished may be left to the lower levels. New objectives may also be added at each level. Feedback is required to measure how successfully the functions were performed.

As an example, while government and/or company standards may require a hazard analysis be performed, the system designers and documenters (including those designing the operational procedures and writing user manuals) may have control over the actual hazard analysis process used to identify specific safety constraints on the design and operation of the system. The design constraints identified as necessary to control system hazards are passed to the implementers and assurers of the individual system components along with standards and other requirements. Success is determined through test reports, reviews, and various additional hazard analyses. At the end of the development process, the results of the hazard analyses as well as documentation of the safety-related design features and design rationale should be passed on to the maintenance group to be used in the change process.

A similar process involving layers of control is found in the system operation control structure (the right half of Figure 4). In addition, there will be (or at least should be) interactions between the two structures. For example, the safety design constraints used during development form the basis for operating procedures and for performance and process auditing.

As in any control structure, time lags may affect the flow of control actions and feedback and may impact the efficiency of the control loops. For example, standards can take years to develop or change—a time scale that may keep them behind current technology and practice. In general, a common way to deal with time lags is to delegate control responsibility to lower levels that are not subject to as great a delay in obtaining information or feedback from the measuring channels. In periods of quickly changing technology, time lags may make it necessary for the lower levels to augment the control processes passed down from above or to modify them to fit the current situation. Accident analysis needs to include the influence of these time lags.

Aside from a few isolated proposals such as MORT, Johnson's attempt to incorporate management factors into a fault tree format [11], accident models and hazard analysis techniques have omitted all the levels of this socio-technical safety control structure except the lowest, technical level. Effective accident analysis and prevention, however, requires going beyond the current approaches.

In the next section, general factors leading to accidents are identified by applying the concepts of constraints, basic control loops, and levels of control, as presented in this and the previous two

sections.

3.4 A Classification of Accident Factors

It was hypothesized earlier that accidents result from inadequate control, i.e., the control loop creates or does not handle dysfunctional interactions in the process—including interactions caused both by component failures and by system design flaws. Starting from this basic definition of an accident, the process that leads to accidents can be understood in terms of flaws in the components of the system development and system operations control loops in place during design, development, manufacturing, and operations. This section presents a classification of those flaws. The classification can be used during accident analysis (or accident prevention activities) to assist in identifying all the factors involved in the accident and in showing their relationships. Figure 5 shows the general classification.

In each control loop at each level of the socio-technical control structure, unsafe behavior results from either a missing or inadequate constraint on the process at the lower level or inadequate enforcement of the constraint leading to its violation. Because each component of the control loop may contribute to inadequate control, classification starts by examining each of the general control loop components and evaluating their potential contribution: (1) the controller may issue inadequate or inappropriate control actions, including inadequate handling of failures or disturbances in the physical process; (2) control actions may be inadequately executed, or (3) there may be missing or inadequate feedback. These same general factors apply at each level of the socio-technical control structure, but the interpretations (applications) of the factor at each level may differ.

For each of the factors, at any point in the control loop where a human or organization is involved, it will be necessary to evaluate the context in which decisions are made and the behavior-shaping mechanisms (influences) at play in order to understand how and why unsafe decisions have been made.

Note that accidents caused by basic component failures are included here. Component failures may result from inadequate constraints on the manufacturing process; inadequate engineering design such as missing or incorrectly implemented fault tolerance; lack of correspondence between individual component capacity (including humans) and task requirements; unhandled environmental disturbances (e.g., EMI); inadequate maintenance, including preventive maintenance; physical degradation over time (wearout), etc. Component failures may be prevented by increasing the integrity or resistance of the component to internal or external influences or by building in safety margins or safety factors. They may also be avoided by operational controls, such as operating the component within its design envelope and by periodic inspections and preventive maintenance. Manufacturing controls can reduce deficiencies or flaws introduced during the manufacturing process. The effects of component failure on system behavior may be eliminated or reduced by using redundancy. The model goes beyond simply blaming component failure for accidents and requires that the reasons be identified for why those failures occurred and led to an accident.

3.4.1 Inadequate Enforcement of Safety Constraints

The first factor, inadequate control over (enforcement of) safety constraints, can occur either because hazards (and their related constraints) were not identified (1.1 in Figure 5) or because the control actions do not adequately enforce the constraints (1.2). The latter may, in turn, result from flawed control algorithms (1.2.1), inconsistent or incorrect process models used by the control algorithms (1.2.2), or by inadequate coordination among multiple controllers and decision makers (1.2.3).

1. **Inadequate Enforcement of Constraints (Control Actions)**
   - 1.1 Unidentified hazards
   - 1.2 Inappropriate, ineffective, or missing control actions for identified hazards
     - 1.2.1 Design of control algorithm (process) does not enforce constraints
       - Flaw(s) in creation process
       - Process changes without appropriate change in control algorithm (asynchronous evolution)
       - Incorrect modification or adaptation
     - 1.2.2 Process models inconsistent, incomplete, or incorrect (lack of linkup)
       - Flaw(s) in creation process
       - Flaws(s) in updating process (asynchronous evolution)
       - Time lags and measurement inaccuracies not accounted for
     - 1.2.3 Inadequate coordination among controllers and decision makers (boundary and overlap areas)
2. **Inadequate Execution of Control Action**
   - 2.1 Communication flaw
   - 2.2 Inadequate actuator operation
   - 2.3 Time lag
3. **Inadequate or missing feedback**
   - 3.1 Not provided in system design
   - 3.2 Communication flaw
   - 3.3 Time lag
   - 3.4 Inadequate sensor operation (incorrect or no information provided)

Figure 5: A Classification of Control Flaws Leading to Hazards

Inadequate Control Algorithms: Control algorithms may not enforce safety constraints (1.2.1) because they are inadequately designed originally, the process may change and thus they become inadequate, or they may be inadequately modified by maintainers (if they are automated) or through various types of natural adaptation if they are implemented by humans. Leplat has noted that many accidents relate to *asynchronous evolution* [14] where one part of a system (in our case the hierarchical control structure) changes without the related necessary changes in other parts. Changes to subsystems may be carefully designed, but consideration of their effects on other parts of the system, including the control aspects, may be neglected or inadequate. Asynchronous evolution may also occur when one part of a properly designed system deteriorates. In both these cases, the erroneous expectations of users or system components about the behavior of the changed or degraded subsystem may lead to accidents. The Ariane 5 trajectory changed from that of the Ariane 4, but the inertial reference system software did not. One factor in the loss of contact with SOHO (SOlar Heliosperic Observatory) in 1998 was the failure to communicate to operators that a functional change had been made in a procedure to perform gyro spin down.

Communication is a critical factor here as well as monitoring for changes that may occur and feeding back this information to the higher-level control. For example, the safety analysis process that generates constraints always involves some basic assumptions about the operating environment of the process. When the environment changes such that those assumptions are no longer true, the controls in place may become inadequate. Embedded pacemakers, for example, were originally assumed to be used only in adults, who would lie quietly in the doctor's office while the pacemaker was being "programmed." Later they began to be used in children, and the assumptions under

which the hazard analysis was conducted and the controls were designed no longer held and needed to be revisited.

Inconsistent Process Models: Section 3.2 stated that effective control is based on a model of the process state. Accidents, particularly system accidents, most often result from inconsistencies between the models of the process used by the controllers (both human and automated) and the actual process state (1.2.2). When the controller's model of the process (either the human mental model or the software model) diverges from the process state, erroneous control commands (based on the incorrect model) can lead to an accident—for example, (1) the software does not know that the plane is on the ground and raises the landing gear or (2) it does not identify an object as friendly and shoots a missile at it or (3) the pilot thinks the aircraft controls are in *speed* mode but the computer has changed the mode to *open descent* and the pilot issues inappropriate commands for that mode or (4) the computer does not think the aircraft has landed and overrides the pilots' attempts to operate the braking system.[4]

During software development, the programmers' models of required behavior may not match that of the engineers' (commonly referred to as software requirements error), or the software may be executed on computer hardware during operations that differs from that assumed by the programmer and used during testing. The situation becomes more even complicated when there are multiple controllers (both human and automated) because each of their process models must also be kept consistent.

The most common form of inconsistency occurs when one or more of the process models is incomplete in terms of not defining appropriate behavior for all possible process states or all possible disturbances, including unhandled or incorrectly handled component failures. Of course, no models are complete in the absolute sense: The goal is to make them complete enough that no safety constraints are violated when they are used. We have defined (or at least made progress toward defining) what it means for a software model of the process to be complete in this sense [16] and are working on determining what the human controller's mental model must contain to safely control the process and to supervise automated controllers.[5]

How do the models become inconsistent? First, they may be wrong from the beginning (e.g., incorrect software requirements). In this case, the design of the controller itself is flawed: there may be uncontrolled disturbances, unhandled process states, inadvertent commands of the system into a hazardous state, unhandled or incorrectly handled system component failures, etc.

In addition to not starting with an accurate model, models may become incorrect due to lack of feedback, inaccurate feedback, or inadequate processing of the feedback. A contributing factor cited in the Cali B-757 accident report was the omission of the waypoints behind the aircraft from cockpit displays, which contributed to the crew not realizing that the waypoint for which they were searching was behind them (missing feedback). The model of the Ariane 501 attitude used by the attitude control software became inconsistent with the launcher attitude when an error message sent by the inertial reference system was interpreted by the attitude control system as data (incorrect processing of feedback), leading to the issuance of an incorrect and unsafe control command.

Other reasons for the process models to diverge may be more subtle. Information about the process state has to be inferred from measurements. For example, in the TCAS II collision avoidance system, relative range positions of other aircraft are computed based on round-trip message propagation time. The theoretical control function (control law) uses the true values of the controlled

[4] All of these examples have actually occurred.

[5] We are not hypothesizing how mental models operate but simply the basic information that must be included. The result should assist in the design of human–machine interfaces when safety is a design goal [2].

variables or component states (e.g., true aircraft positions). However, at any time, the controller has only measured values, which may be subject to time lags or inaccuracies. The controller must use these measured values to infer the true conditions in the process and, if necessary, to derive corrective actions to maintain the required process state. In the TCAS example, sensors include on-board devices such as altimeters that provide measured altitude (not necessarily true altitude) and antennas for communicating with other aircraft. The primary TCAS actuator is the pilot, who may or may not respond to system advisories. The mapping between measured or assumed values and true values can be flawed.

In addition, the control loop must necessarily include time lags, such as that between measuring values and receiving those values or between issuing a command and the actual process state change. Pilot response delays are important time lags that must be considered in designing the control function for TCAS or other aircraft systems as are time lags in the controlled process (the aircraft trajectory) caused by aircraft performance limitations. Delays may not be directly observable, but may need to be inferred. Depending on where in the feedback loop the delay occurs, different models are required to cope with the delays [4]: dead time and time constants require a model that makes it possible to predict when an action is needed before the need arises while feedback delays require a model that allows prediction of when a given action has taken effect and when resources will be available again. Such requirements may impose the need for some type of open loop or feedforward strategy to cope with delays.

To summarize, process models can be incorrect from the beginning (where correct is defined in terms of consistency with the current process state and with the models being used by other controllers) or they can become incorrect due to erroneous or missing feedback or measurement inaccuracies. They may also be incorrect only for short periods of time due to time lags in the process loop.

Inadequate Coordination Among Controllers and Decision Makers: When there are multiple controllers (human and/or automated), control actions may be inadequately coordinated (1.2.3), including unexpected side effects of decisions or actions or conflicting control actions. Communication flaws play an important role here.

Leplat suggests that accidents are most likely in *boundary areas* or in *overlap areas* where two or more controllers (human and/or automated) control the same process [14]. In both boundary and overlap areas, the potential exists for ambiguity and for conflicts among independently made decisions.

When controlling boundary areas, there can be confusion over who is actually in control (which control loop is currently exercising control over the process), leading to missing control actions. The functions in the boundary areas are often poorly defined. For example, Leplat cites an iron and steel plant where frequent accidents occurred at the boundary of the blast furnace department and the transport department. One conflict arose when a signal informing transport workers of the state of the blast furnace did not work and was not repaired because each department was waiting for the other to fix it. Faverge suggests that such dysfunctioning can be related to the number of management levels separating the workers in the departments from a common manager: The greater the distance, the more difficult the communication, and thus the greater the uncertainty and risk.

Coordination problems in the control of boundary areas are rife. A Milstar satellite was lost due to inadequate attitude control of the Titan/Centaur launch vehicle, which used an incorrect process model based on erroneous inputs in a software load tape. After the accident, it was discovered that nobody had tested the software using the actual load tape—everyone assumed someone else

was doing so. In this case, system engineering and mission assurance activities were missing or ineffective, and a common control or management function was quite distant from the individual development and assurance groups. A factor in the loss of the Black Hawk helicopters to friendly fire over northern Iraq was that the helicopters normally flew only in the boundary areas of the No-Fly-Zone, and procedures for handling aircraft in those areas were ill-defined. Another factor was that an Army base controlled the flights of the Black Hawks while an Air Force base controlled all the other components of the airspace. A common control point once again was high above where the accident occurred in the control structure. In addition, communication problems existed between the Army and Air Force bases at the intermediate control levels.

Overlap areas exist when a function is achieved by the cooperation of two controllers or when two controllers exert influence on the same object. Such overlap creates the potential for conflicting control actions (dysfunctional interactions among control actions). Leplat cites a study of the steel industry that found 67 percent of technical incidents with material damage occurred in areas of co-activity, although these represented only a small percentage of the total activity areas. In an A320 accident in Bangalore, India, the pilot had disconnected his flight director during approach and assumed that the co-pilot would do the same. The result would have been a mode configuration in which airspeed is automatically controlled by the autothrottle (the *speed* mode), which is the recommended procedure for the approach phase. However, the co-pilot had not turned off his flight director, which meant that *open descent* mode became active when a lower altitude was selected instead of *speed* mode, eventually contributing to the crash of the aircraft short of the runway [26]. In the Black Hawks' shootdown by friendly fire, the aircraft surveillance officer (ASO) thought she was responsible only for identifying and tracking aircraft south of the 36th parallel while the air traffic controller for the area north of the 36th parallel thought the ASO was also tracking and identifying aircraft in his area and acted accordingly.

3.4.2 Inadequate Execution of the Control Action

A second way for constraints to be violated in the controlled process is if there is a failure or inadequacy in the reference channel, i.e., in the transmission of control commands or in their execution (actuator fault or failure). A common flaw in system development is that the safety information gathered or created by the system safety engineers (the hazards and the necessary design constraints to control them) is inadequately communicated to the system designers and testers.

3.4.3 Inadequate or Missing Feedback

The third flaw leading to system hazards involves inadequate feedback. A basic principle of system theory is that no control system will perform better than its measuring channel. Important questions therefore arise about whether the controllers or decision makers (either automated or human) have the necessary information about the actual state of the controlled process to satisfy their objectives. This information is contained in their process models and updating these models correctly is crucial to avoiding accidents (1.2.2). Feedback may be missing or inadequate because such feedback is not included in the system design (3.1), flaws exist in the monitoring or feedback communication channel (3.2), the feedback is not timely (3.3), or the measuring instrument operates inadequately (3.4).

4 Summary and Uses for the Model

This paper has described a new accident model based on system theory. Each level of the socio-technical structure of a system can be described in terms of levels of control. Each level exercises control over emergent properties, in this case safety, arising from (1) component failures, (2) dysfunctional interactions among components, or (3) unhandled environmental disturbances at a lower level. Managing safety requires identifying the constraints on process behavior necessary to ensure safety and imposing these constraints (through design or operations) to limit the behavior of the process below to safe changes and adaptations.

The new model focuses particular attention on the role of constraints in safety management. Accidents are seen as resulting from inadequate control or enforcement of constraints on safety-related behavior at each level of the system development and system operations control structures. Accidents can be understood, therefore, in terms of why the controls that were in place did not prevent or detect maladaptive changes, that is, by identifying the safety constraints that were violated at each level of the control structure as well as why they were inadequate or, if they were potentially adequate, why the system was unable to exert appropriate control over their enforcement. The process leading to an accident (loss event) can be described in terms of an adaptive feedback function that fails to maintain safety as performance changes over time to meet a complex set of goals and values. The adaptive feedback mechanism allows the model to incorporate adaptation as a fundamental property.

We have found in practice that using STAMP helped us to separate factual data from the interpretations of that data: While the factors involved in accidents may be clear, their importance and the explanations for why the factors were present are often subjective. Our models were also more complete. Each of the explanations for the incorrect FMS input of R in the Cali American Airlines accident described in Section 2 appears in the analysis of that accident using the new systems accident model, but they appear at the different levels of the control structure where they operated. The modeling also helped us to understand the relationships among these factors.

While STAMP will probably not be useful in law suits as it does not assign blame for the accident to a specific person or group, it does provide more help in understanding accidents by forcing examination of each part of the socio-technical system to see how it contributed to the loss (and there will usually be contributions at each level). Such understanding should help in learning how to engineer safer systems, including the technical, managerial, organizational, and regulatory aspects.

To accomplish this goal, a framework for classifying the factors that lead to accidents was derived from the basic underlying conceptual accident model. This classification can be used in identifying the factors involved in a particular accident and in understanding their role in the process leading to the loss. The accident investigation after the Black Hawk shootdown identified 130 different factors involved in the accident. In the end, only an AWACS air traffic control operator was court martialed, and he was acquitted. The more one knows about an accident process, the more difficult it is to find one person or part of the system responsible.

STAMP should be useful not only in analyzing accidents that have occurred but in developing system engineering methodologies to prevent accidents. Hazard analysis can be thought of as investigating an accident before it occurs. Traditional hazard analysis techniques, such as fault tree analysis and various types of failure analysis techniques, do not work well for software and system design errors. Nor do they usually include organizational and management flaws. The problem is that these hazard analysis techniques are limited by a focus on failure events and the role of component failures in accidents and do not account for the complex roles that software and humans are assuming in high-tech systems.

STAMP provides a direction to take in creating new hazard analysis and prevention techniques that go beyond component failure and are more effective against system accidents, accidents related to the use of software, accidents involving cognitively complex human activities, and accidents related to societal and organizational factors. Because in a system accident model everything starts from constraints, the new approaches would focus on identifying the constraints required to maintain safety and then designing the system and operating conditions to ensure that the constraints are enforced. Such hazard analysis techniques would augment the failure-based methods and encourage a wider variety of risk reduction measures than simply adding redundancy to deal with component failures.

A system accident model could also point the way to very different approaches to risk assessment. Currently, risk assessment is firmly rooted in the probabilistic analysis of failure events. Attempts to extend current PRA techniques to software and other new technology, to management, and to cognitively complex human control activities have been disappointing. The arguments in this paper suggest that this way forward leads to a dead end. Significant progress in risk assessment for complex systems may require innovative approaches starting from a completely different theoretical foundation.

STAMP could also be used to improve performance analysis. Performance monitoring of complex systems has created some dilemmas. Computers allow the collection of massive amounts of data, but analyzing that data to determine whether the system is moving toward the boundaries of safe behavior is difficult. The use of a system accident model and the basic concept of safety constraints may provide directions for identifying appropriate safety metrics; determining whether control over those constraints is adequate; evaluating the assumptions about the technical failures and potential design errors, organizational structure, and human behavior underlying the hazard analysis; detecting errors in the operational and environmental assumptions underlying the design, and identifying any maladaptive changes over time that could increase risk of accidents to unacceptable levels.

We are currently working on validating STAMP with respect to accident analysis by applying it to example accidents and comparing the results with traditional event chain models. We are also deriving new engineering techniques based on it for preventing accidents (hazard analysis and approaches to designing for safety), assessing risk, and monitoring performance.

References

[1] W.R. Ashby. *An Introduction to Cybernetics.* London: Chapman and Hall, 1956.

[2] Edward Bachelder and Nancy Leveson. Describing and probing complex system behavior: A graphical approach. *Aviation Safety Conference*, Society of Automotive Engineers, Seattle, September 2001.

[3] Ludwig Benner Jr. Accident investigations: Multilinear events sequencing methods. *Journal of Safety Research*, 7(2):67–73, June 1975.

[4] B. Brehmer. Dynamic Decision Making: Human Control of Complex Systems. *Acta Psychologica*, Vol. 81, pp. 211-241.

[5] Bureau of Air Safety Investigation. Advanced Technology Aircraft Safety Survey Report. Department of Transport and Regional Development, Australia, June 1996.

[6] Peter Checkland. *Systems Thinking, Systems Practice.* John Wiley & Sons, New York, 1981.

[7] R.C. Conant and W.R. Ashby. Every good regulator of a system must be a model of that system. *International Journal of System Science*, 1, ppg. 89-97, 1970.

[8] Richard I. Cook. Verite, Abstraction, and Ordinateur Systems in the Evolution of Complex Process Control. *3rd Annual Symposium on Human Interaction with Complex Systems (HICS '96)*, Dayton Ohio, August 1996.

[9] W. Edwards. Dynamic decision theory and probabilistic information processing. *Human Factors*, 4, pp. 59–73, 1962.

[10] Y. Fujita. What Shapes Operator Performance? JAERI Human Factors Meeting, Tokyo, November 1991.

[11] William G. Johnson. *MORT Safety Assurance System*, New York: Marcel Dekker, 1980.

[12] Trevor A. Kletz. Human problems with computer control. *Plant/Operations Progress*, 1(4), October 1982.

[13] Trevor Kletz. Human problems with computer control. *Plant/Operations Progress*, 1(4), October 1982.

[14] Jacques Leplat. Occupational accident research and systems approach. In Jens Rasmussen, Keith Duncan, and Jacques Leplat, editors, *New Technology and Human Error*, pages 181–191, John Wiley & Sons, New York, 1987.

[15] Nancy G. Leveson. High-Pressure Steam Engines and Computer Software. *IEEE Computer*, October 1994 (Keynote Address from IEEE/ACM International Conference on Software Engineering, 1992, Melbourne, Australia). Also available from http://sunnyday.mit.edu.

[16] Nancy G. Leveson. *Safeware: System Safety and Computers.* Addison Wesley, 1995.

[17] Nancy G. Leveson. *Evaluating Accident Models using Recent Aerospace Accidents.* Technical Report, MIT Dept. of Aeronautics and Astronautics, 2001 (available at http://sunnyday.mit.edu/accidents).

[18] Dale A. Mackall. Development and Flight Test Experiences with a Flight-Critical Digital Control System. NASA Technical Paper 2857, National Aeronautics and Space Administration, Dryden Flight Research Facility, November 1988.

[19] Ralph F. Miles Jr. Introduction. In Ralph F. Miles Jr., editor, *Systems Concepts: Lectures on Contemporary Approaches to Systems*, pages 1–12, John F. Wiley & Sons, New York, 1973.

[20] Gary A. Klein, Judith Orasano, R. Calderwood, and Caroline E. Zsambok (Editors). *Decision Making in Action: Models and Methods.* Ablex Publishers, 1993.

[21] Perrow, C. *Normal Accidents: Living with High-Risk Technology.* Basic Books, Inc., New York, 1984.

[22] Jens Rasmussen. Human error and the problem of causality in analysis of accidents. In D.E. Broadbent, J. Reason, and A. Baddeley, editors, *Human Factors in Hazardous Situations*, pages 1–12, Clarendon Press, Oxford, 1990.

[23] Jens Rasmussen. Risk Management in a Dynamic Society: A Modelling Problem. *Safety Science*, vol. 27, No. 2/3, Elsevier Science Ltd., 1997, pages 183-213.

[24] Jens Rasmussen, Annelise Mark Pejtersen, and L.P. Goodstein. *Cognitive System Engineering.* John Wiley & Sons, 1994.

[25] Jens Rasmussen and Inge Svedung. *Proactive Risk Management in a Dynamic Society.* Swedish Rescue Services Agency, 2000.

[26] Nadine Sarter and David Woods. "How in the world did I ever get into that mode?": Mode error and awareness in supervisory control. *Human Factors 37*, 5–19.

[27] Nadine N. Sarter and David Woods. Strong, silent, and out-of-the-loop. CSEL Report 95-TR-01, Ohio State University, February 1995.

[28] Nadine Sarter, David D. Woods, and Charles E. Billings. Automation Surprises. in G. Salvendy (Ed.) *Handbook of Human Factors/Ergonomics*, 2nd Edition, Wiley, New York, in press.

[29] Jouko Suokas. On the reliability and validity of safety analysis. Technical Report Publications 25, Technical Research Center of Finland, Espoo, Finland, September 1985.

[30] Kim J. Vicente. A Field Study of Operator Cognitive Monitoring at Pickering Nuclear Generating Station. Technical Report CEL 9504, Cognitive Engineering Laboratory, University of Toronto, 1995.

[31] Kim J. Vicente. *Cognitive Work Analysis: Toward Safe, Productive, and Healthy Computer-Based Work.* Erlbaum Associates, April 1999.

[32] David D. Woods. Some results on operator performance in emergency events. in D. Whitfield (ed.), *Ergonomic Problems in Process Operations*, Institute of Chemical Engineering Symposium, Ser. 90, 1984.

[33] David D. Woods. Lessons from beyond human error: Designing for resilience in the face of change and surprise. Design for Safety Workshop, NASA Ames Research Center, October 8-10, 2000.

[34] Caroline E. Zsambok and Gary Klein (Editors). *Naturalistic Decision Making.* Lawrence Erlbaum Associates, 1997.