Evaluating Accident Models using Recent Aerospace Accidents[1][2]

Part I: Event-Based Models

Nancy Leveson

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

June 28, 2001

[1]This work in this report was partially supported by the NASA Design For Safety program and by a grant from the NASA IV&V Facility (West Virginia) Center Software Initiative. The results were presented at NASA Ames on March 22, 2001.

Preface

This report was inspired by a graduate reading group that was held at MIT during the fall semester, 2000. Each week, 20-25 graduate students and 3-4 faculty examined a different aerospace accident report (about evenly divided between aeronautics and astronautics). Our goal was to look at recent accidents (all but one of the accidents involved software) and determine how well traditional accident models fit these accidents in modern complex systems and whether there are common systemic factors that can be identified. The accident reports examined involved the Ariane 5; the Space Shuttle Challenger; the Mars Climate Orbiter; the Mars Polar Lander; the Titan IV/Milstar; an American Airlines B-757 near Cali, Colombia; a Lufthansa A320 at Warsaw; and a China Airlines A320 at Nagoya, Japan.

The first chapter describes the accident models evaluated. Then the limitations in using accident reports to understand accidents are examined in Chapter 2 to identify the boundaries of what can be learned in such an exercise and to provide caveats in drawing conclusions from such reports. The third chapter describes the accidents themselves and the causal factors that would be included for each of the models. Chapter 4 uses the results of the previous sections in an evaluation and comparison of the models and identifies common factors we found in the accidents and the accident reports.

The participants during the semester varied from week to week but the following people participated in at least some of the accident presentations and discussions:

Prof. Nancy Leveson (seminar organizer)
Prof. R. John Hansman
Prof. Brian Williams
Dr. Kristina Lundqvist (postdoc)
Masafumi Katahira (visitor from NASDA)
Hideki Nomoto (visitor from NASDA)

John Bellingham
Emily Craparo (undergrad)
Mirna Daouk
Hayley Davison
John Enright
Miwa Hayashi
Karen Marais
Israel Navarro
Natasha Neogi
Tom Reynolds
Jayakanth Srinivasan
John van Eepoel
Laurence Vigeant-Langlois
Maxime du Villepin
Marc Zimmerman

Executive Summary

Accident Models

Accident models are used to explain how accidents occur. The explanations of the etiology of accidents embodied in accident models forms the basis for investigating accidents, preventing future ones, and determining whether existing systems are suitable for use (risk assessment).

The models impose patterns on an accident and thus will influence both the data collected and the factors identified as causative. While accident models are a way to organize data and set priorities in accident investigations, at the same time they may either act as a filter in the collection of data that narrows the investigation or they may expand the investigation by forcing consideration of factors that are often omitted.

The report is divided into two parts. Part I considers event-based accident models including domino and single event, chains of events, and hierarchical. Part II will look at new proposals for models based on control theory.

Domino or Single Event Models

Domino or single event models were the first to be proposed; they arose in industrial (worker) safety. In these models, out of a large number of necessary conditions for the accident, one is chosen and labeled as the cause, even though all the factors involved were equally indispensable to the occurrence of the event. Most accidents involve a variety of events and conditions, and identifying only a single factor as the "cause" can be a hindrance in preventing future accidents.

One reason for the tendency to look for a single cause is to assign blame, often for legal purposes. Blame, however, is not an engineering concept; it is a legal or moral one. Usually there is no objective criterion for distinguishing one factor or several factors from the other factors that make up the cause of an accident.

Although the legal approach to causality may have benefits when establishing guilt and liability, it is of little use from an engineering perspective, where the goal is to understand and prevent accidents. It may even be a hindrance because the most relevant factors (in terms of preventing future accidents) may be ignored for nontechnical reasons. Even determining relative importance of factors to an accident may not be useful in preventing future losses (assuming that "importance" can be defined). Hadden argues that countermeasures to accidents should not be determined by the relative importance of the causal factors; instead, priority should be given to the measures that will be most effective in reducing losses.

Any particular condition in a complex technological system is unlikely to be either necessary nor sufficient to cause an accident. The high frequency of accidents having complex causes probably results from the fact that competent engineering and organizational structures eliminate the simpler causes. On the positive side, the very complexity of accident processes means that there may

be many opportunities to intervene or interrupt them. Therefore, thorough consideration of the conditions leading to accidents will be more useful than simplistic explanations.

Chains of Time-Ordered Events

The most common accident models include multiple events related by a forward chain over time. The events considered almost always involve some type of component failure, human error, or energy-related event. There may be other relationships represented by the chain in addition to a chronological one, but any relationship is almost always a direct, linear one.

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. The stopping point in tracing back events preceding an accident may be chosen because (1) it represents events that are familiar and thus acceptable as explanations for the accident, (2) the events included are those for which it is felt something can be done for correction, or (3) including additional events may raise political or other difficulties. 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, and the events, conditions, and links between them chosen to explain the "cause' are subjective and subject to bias.

The countermeasures to prevent accidents considered as chains of events usually involve either removing the 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).

Hierarchical Models

Accidents may be prevented most effectively not by simply eliminating the direct causes identified in the chain of events, but by manipulating other indirectly related factors. Such factors are difficult to incorporate into most chain-of-events models.

Hierarchical models provide multiple models of accident causation at different levels of abstraction. In the three-level model used here, the lowest level describes the accident mechanism (the chain of events). The second level of understanding causality includes conditions or lack of conditions that affected the occurrence of the events at the first level. The factors at the third or highest level, sometimes called *root causes* or *systemic factors* affect general classes of accidents; they are weaknesses of a technical, human, organizational, managerial, or societal nature that not only contributed to the accident being investigated but are likely to affect classes of accidents in the future. Often, responses to accidents involve fixing only a specific condition while leaving the general or systemic factors untouched. The result is usually a repetition of the same general class of accident but displaying slightly different symptoms.

Countermeasures aimed at preventing accidents described by hierarchical models require making changes at all the levels, particularly the level 3 systemic factors.

Limitations in Using Accident Reports to Understand Accidents

Investigating accidents to identify causal factors in order to prevent future repetition, the so-called *fly-fix-fly* approach to safety engineering, has been extremely successful in decreasing the accident rate in civil aviation. Some reasons for this are the use of standard designs and the slow pace of

technological innovation. When technology changes rapidly, however, or when radical new designs are introduced, the effectiveness of learning from past failures is more limited.

In addition, such an approach is dependent on the data reported by accident investigators and the causal factors recorded. The applicability of the results of accident reports to current or future systems also depends on whether the relevant circumstances (both within the system and in its environment) are similar, but the necessary information to make such a determination is often omitted from accident reports.

To formulate hypotheses about the cause of an accident, investigators usually collect data by questioning individuals and by gathering and examining other evidence. Both of these sources of data may involve filtering and subjectivity and the collection process itself can limit the information acquired.

The selection of events to include in accident explanation will be affected by the analyst's representation (model) of accidents and will include only those factors that agree with that mental model or belief about how or why losses occur. The selection of causal factors is just as subjective. To complicate things further, explanations for events involving human goals and motives will depend on assumptions that cannot be *directly* measured or observed by the accident investigator.

Not only may the accident data be systematically filtered and unreliable, but accident causes may be oversimplified, such as focusing only on human error or on technical failures.

All human activity takes place within and is influenced by the environment, both physical and social, in which it takes place. Operator error cannot be understood or prevented without understanding the environmental factors that influence those actions: It is often very difficult to separate design error from operator error. In highly automated systems, the operator is often at the mercy of the system design and operational procedures.

Just as important an oversimplification as blaming the operator is concentrating only on technical failures and immediate physical events, which can allow latent design errors to go uncorrected and to be repeated. And with the increasing role of software in complex systems (where the problem is always design error), concentrating our safety engineering efforts on physical failures and the use of redundancy to prevent them will become increasingly ineffective.

Large-scale engineered systems are more than just a collection of technological artifacts: They are a reflection of the structure, management, procedures, and culture of the engineering organization that created them, and they are also, usually, a reflection of the society in which they were created.

Accident Analysis

This section of the report contains detailed examination of some recent aerospace accidents, all but one of which involved software as an important factor. The accident reports are examined and attempts made to create both a chain-of-events model and a hierarchical model for each. A control theory model will be added in the next draft. The accidents chosen are the Ariane 5; the Space Shuttle Challenger (included for comparison purposes because of the thoroughness of the Rogers' Commission report); the Mars Climate Orbiter; the Mars Polar Lander; the Titan IV-B/Milstar loss; an American Airline B-757 near Cali, Colombia; a Lufthansa A320 at Warsaw; and a China Airlines A320 at Nagoya, Japan.

Comparisons and Evaluations of the Models

In general, event chains turned out to be useful in understanding the mechanism, particularly the physical mechanism, involved in an accident. They were less useful, however, in identifying and understanding the other factors in accidents. The aircraft accident reports, in addition, were found to be focused so much on establishing blame (and making sure the pilots were the object of that blame) that the events and conditions included were very limited.

The great variation in the factors identified as "causes" in the accident reports for similar accidents implies that event chain models provide too little guidance to the investigator. In addition, the accident reports all exhibited tremendous subjectivity and filtering in the events and conditions chosen and the wording used to describe them.

The stopping factor used in tracing backward along an event chain from the loss event was more important than originally assumed. The strangest (and most counterproductive) definition was that used in the NASA Mars Climate Orbiter report for identifying what it called *root causes* and *contributory causes*. According to the report, the definitions used come from a NASA standard. NASA should consider reevaluating the approach to accident investigation implied by that standard.

The hierarchical model was an improvement over the basic chain-of-events model in important ways. Separating events from the various types of explanations that could be given for those events allowed evaluation of the explanations in a more objective fashion and easier detection of omissions and biases. It also helped to identify conditions indirectly related to events or those related to all or to a subset of the events. Finally, it proved to be a great help in evaluating the completeness and coverage of the accident investigation.

Common Factors Found in the Accidents

Given the incompleteness, filtering, and subjectivity found in most of the accident reports considered, any analysis of common factors is necessarily limited in the conclusions that can be reached. However, some striking similarities in systemic factors were identified. Because the aircraft accidents focused so much on pilot error and operations and much less on engineering issues, the systemic factors that could be identified differed from those for the spacecraft accidents. For each type (spacecraft and aircraft), the systemic factors are grouped into three categories: flaws in the safety culture, ineffective organizational structures and communication, and ineffective or inadequate technical activities.

Spacecraft Accident Factors

1. Flaws in the Safety Culture

    - Overconfidence and complacency
    - Underestimating or not understanding software risks
    - Overrelying on redundancy
    - Assuming risk decreases over time
    - Ignoring warning signs

2. Ineffective Organizational Structure and Communication

    - Diffusion of responsibility and authority
    - Low-level status and inappropriate organizational placement of the safety program.
    - Limited Communication Channels and Poor Information Flow

3. Ineffective or Inadequate Technical Activities

   - Inadequate system engineering
   - Flawed review process
   - Inadequate specifications
   - Violation of basic safety engineering practices in the digital parts of system design
   - Inadequate software engineering
   - Flawed or inadequate analysis of software functions
   - Software reuse without appropriate analysis of its safety
   - Inadequate system safety engineering
   - Unnecessary complexity and software functions
   - Test and simulation environments that do not match the operational environment
   - Deficiencies in safety-related information collection and use
   - Operational personnel not understanding the automation.

Aircraft Accident Factors

1. Flaws in the Safety Culture

   - Overconfidence on Automation
   - Slow understanding of the problems associated with human–automation mismatch within the aviation industry.
   - Incorrect prioritization of changes to the automation
   - Ignoring warnings and near misses

2. Ineffective Organizational Structure and Communication

   - Limited Communication Channels and Poor Information Flow

3. Ineffective or Inadequate Technical Activities

   - Conflicting and inadequate documentation
   - Ineffective or inadequate cognitive engineering.
   - Flight crews not understanding the automation.
   - Inadequate system and safety engineering
   - Inadequate design of feedback to pilots

Part II: A Model Based on Control Theory

Chain-of-events models provide too little guidance in the selection of events to include in the accident explanation and, more important, in the selection of conditions to investigate. We need a more scientific way to model accidents that produces a better and less subjective understanding of *why* the accident occurred and how to prevent future ones.

In addition, chain of events models work best for component failure accidents, where one or several components fail, leading to a system failure or hazard. After World War II, however, a new type of accident has been identified where none of the components may fail. Perrow coined the term *system accident* for those accidents that arise in the interactions *among* components (electromechanical, digital, and human) rather than the failure of individual components [33]. The extraordinary interactive complexity of the systems we are trying to build as well as the introduction

of new technology, particularly digital technology, are part of the reason behind this change in the nature of the accidents we are experiencing. While better engineering techniques are reducing accidents related to hardware failure, system accidents are increasing in importance and will require new prevention approaches. New accidents models are needed that handle system accidents and the new types of accident mechanisms we are starting to experience in software-intensive systems.

The hypothesis to be considered in Part II of this report is that accident models based on control theory may satisfy these requirements. In control theory, systems are viewed as interrelated components that are kept in a state of dynamic equilibrium by feedback loops of information and control. Accidents occur when external disturbances or dysfunctional interactions among system components are not adequately handled by the control system. Safety then is viewed as a control problem, including control imposed by the management functions in an organization.

Part II of this report reviews what has been written about the use of control theory for accident models, proposes a basis for a new model, and compares it to the event-based models using the accidents modeled in Part I of this report.

If the new model proves to be useful in explaining accidents that have already occurred, then longer-term steps will involve creating and evaluating new hazard analysis methods and risk assessment techniques based on the new model.

Contents

Chapter 1

Accident Models

1.1 Introduction

Accident models underlie all efforts to engineer for safety. Models, in general, provide a means for understanding phenomena and recording that understanding in a way that can be communicated to others. Note that all models are abstractions—they simplify the thing being modeled by abstracting away what are assumed to be irrelevant details and focusing on the features of the phenomenon that are assumed to be the most relevant. That selection process in most cases is arbitrary and dependent entirely on the choice of the modeler, but it is critical in determining the usefulness and accuracy of the model in predicting future events.

Accident models are used to explain how accidents occur. An underlying assumption, therefore, is that there are common patterns in accidents and that they are not simply random events. The explanations of the etiology of accidents embodied in accident models forms the basis for investigating accidents, preventing future ones, and determining whether existing systems are suitable for use (risk assessment). Thus, accident models are used for three distinct purposes: (1) to understand or investigate accidents that have occurred; (2) to prevent future accidents; and (3) to assess the risk associated with an activity, the use of a product, or the operation of a system. While one may not be consciously aware that they are using any model when engaged in these activities, some (perhaps subconscious) model of the phenomenon is always part of the process.

The same model may not be the most appropriate one for each of the three engineering activities. A companion report provides a more complete taxonomy of accident models and evaluates their use for all three purposes [25]: This report investigates only the use of accident models in accident investigation.

When investigating accidents, accident models are used to help identify which factors will be considered: The models impose patterns on an accident and thus will influence both the data collected and the factors identified as causative. Thus models are a way to organize data and set priorities in accident investigations. At the same time, the accident model used by the investigators may act as a filter in the collection of data that narrows down the investigation or it may expand the investigation by forcing consideration of factors that are often omitted. The underlying accident model thus influences the investigation of the events leading to the accident and the conclusions drawn about the causes.

Because accident models influence what cause is ascribed to an accident, the countermeasures taken to prevent future accidents, and the evaluation of the risk in operating a system, the power and features of the model used will greatly affect our ability to identify and control hazards and thus prevent accidents.

In the rest of this section, several accident models are described. The models were selected because they are commonly used or because they represent current thinking about how accidents occur. The models described are: domino and single event; chains of events; and hierarchical. Models based on control theory will be considered in Part II of this report.

1.2 Domino and Single Event Models

Out of a large number of necessary conditions for the accident, one is often chosen and labeled as *the* cause, even though all the factors involved were equally indispensible to the occurrence of the event. For example, a car skidding in the rain may involve many factors including a wet road, incorrect driver reaction to the skid, excessive speed for the conditions, worn tires, incorrectly inflated tires, driver inattention, etc. None of these is sufficient to cause the skid, but one will often be cited as the "cause." A condition may be selected as the cause because it is the last condition to be fulfilled before the effect takes place, its contribution is the most conspicuous, or the selector has some ulterior motive for the selection. Thus, although it is common to isolate one condition and call it *the cause* (or the *proximate, direct*, or *root* cause) and the other conditions *contributory*, there is no basis for this distinction.

Most accidents involve a variety of events and conditions, and identifying only a single factor as the "cause" can be a hindrance in preventing future accidents. For example, in the crash of an American Airlines DC-10 at Chicago's O'Hare Airport in 1979, the U.S. National Safety Transportation Board (NTSB) blamed only a "maintenance-induced crack," and not also a design error that allowed the slats to retract if the wing was punctured. Because of this omission, McDonnell Douglas was not required to change the design, leading to future accidents related to the same design error [33].

In another DC-10 saga, explosive decompression played a critical role in a near-miss over Windsor, Ontario. An American Airlines DC-10 lost part of its passenger floor, and thus all of the control cables that ran through it, when a cargo door opened in flight in June 1972. Due to the extraordinary skill and poise of the pilot, Bryce McCormick, who had trained himself to fly the plane using only the engines because he was concerned about a decompression-caused collapse of the floor, the plane landed safely. After this event, McCormick recommended that every DC-10 pilot be informed of the consequences of explosive decompression and trained in the flying techniques that he and his crew had used to save their passengers and plane. FAA investigators, the National Transportation Safety Board, and engineers at a subcontractor to McDonnell Douglas that designed the fuselage of the plane, all recommended changes in the design of the aircraft. Instead, McDonnell Douglas attributed the Windsor incident totally to human error on the part of the baggage handler responsible for closing the cargo compartment door and not to any error on the part of their designers or engineers and decided all they had to do was to come up with a fix that would prevent baggage handlers from forcing the door.

One of the discoveries after the Windsor incident was that the door could be improperly closed but the external signs, such as the position of the external handle, made it appear to be closed properly. In addition, this incident proved that the cockpit warning system could fail, and the crew would then not know that they were taking off without a properly closed door:

> The aviation industry does not normally receive such manifest warnings of basic design flaws in an aircraft without cost to human life. Windsor deserved to be celebrated as an exceptional case when every life was saved through a combination of crew skill and the sheer luck that the plane was so lightly loaded. If there had been more passengers and thus more weight, damage to the control cables would undoubtedly have been more

severe, and it is highly questionable if any amount of skill could have saved the plane [6].

Almost two years later, in March 1974, a fully loaded Turkish Airline DC-10 crashed near Paris resulting in 346 deaths—one of the worst accidents in aviation history. Once again, the cargo door had opened in flight, causing the cabin floor to collapse, severing the flight control cables. Immediately after the accident, Sanford McDonnell stated the official McDonnell-Douglas position that once again placed the blame on the baggage handler and the ground crew. This time, however, the FAA finally ordered modifications to all DC-10s that eliminated the hazard. In addition, an FAA regulation issued in July 1975 required all wide-bodied jets to be able to tolerate a hole in the fuselage of 20 square feet. By oversimplifying the accident cause as simply baggage handler error and attempting only to eliminate that error rather than the basic engineering design flaws, fixes that could have prevented the Paris crash were not made.

One reason for the tendency to look for a single cause is to assign blame, often for legal purposes. Blame is not an engineering concept; it is a legal or moral one. Usually there is no objective criterion for distinguishing one factor or several factors from other factors that make up the cause of an accident. While lawyers and insurers recognize that many factors contribute to a loss event, for practical and particularly for liability reasons, they often oversimplify the causes of accidents and identify what they call the *proximate* (immediate or direct) cause. The goal is to determine the parties in a dispute that have the legal liability to pay damages, which may be affected by the ability to pay or by public policy considerations (e.g., to discourage company management or even an entire industry from acting in a particular way in the future).

Although the legal approach to causality may have benefits when establishing guilt and liability, it is of little use from an engineering perspective, where the goal is to understand and prevent accidents. It may even be a hindrance because the most relevant factors (in terms of preventing future accidents) may be ignored for nontechnical reasons.

In the lawsuits following the 1995 American Airlines 757 Cali accident, for example, American Airlines was held liable for the crash based on the Colombian investigators blaming the crew for the accident. A U.S. appeals court rejected that conclusion, which led to a lawsuit by American in a federal court in which American alleged that components of the automated aircraft system made by Honeywell Air Transport Systems and Jeppesen Sanderson helped cause the crash. American blamed the software, saying Jeppesen stored the location of the Cali airport beacon in a different file from most other beacons. Lawyers for the computer companies argued that the beacon code could have been properly accessed and that the pilots were in error. The jury concluded that the two companies produced a defective product and that Jeppesen was 17 percent responsible, Honeywell was 8 percent at fault, and American was held to be 75 percent responsible. While such distribution of responsibility may be important in determining how much each company will have to pay, it does not provide any important information with respect to accident prevention in the future. The verdict is interesting, however, in being one of the first cases not settled out of court where the role of software in an accident was acknowledged. And the jury rejected the oversimplified notion of causality being argued.

Determining the relative importance of various factors to an accident may not be useful in preventing future accidents. Haddon [8] argues that countermeasures to accidents should *not* be determined by the relative importance of the causal factors; instead, priority should be given to the measures that will be most effective in reducing losses. Simplistic explanations for accidents often do not provide the information necessary to prevent future losses, and spending a lot of time determining the relative contributions of causes to accidents is not productive outside the legal system. Rather, engineering effort should be devoted to identifying the factors that are easiest to

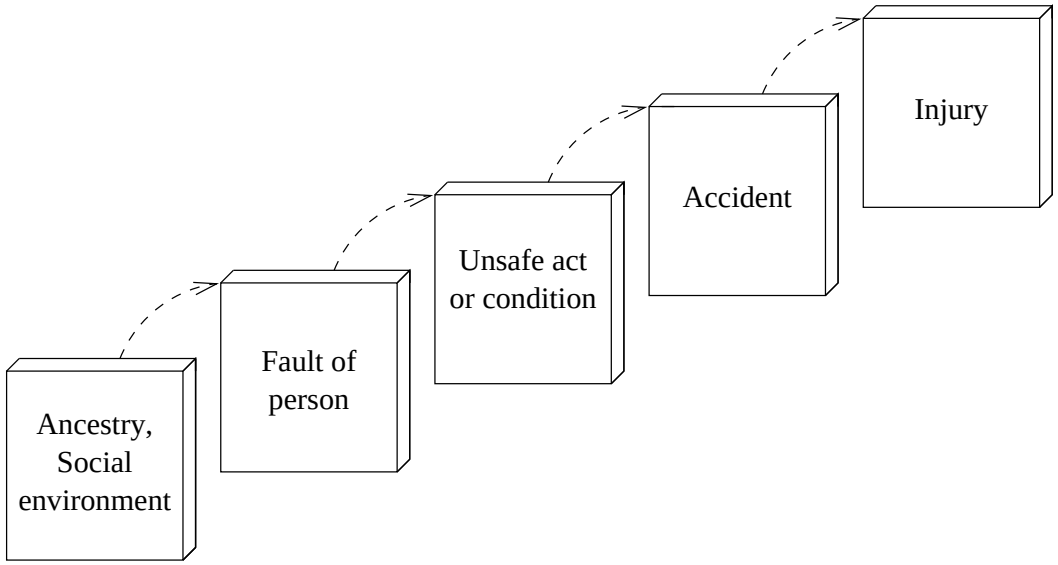

Figure 1.1: Heinrich's Domino Model of Accidents

change or over which we have the greatest control.

Most often, the single cause ascribed to an accident is related to some human action. The earliest formal accident models, such as Heinrich's Domino model, came from *industrial safety* (sometimes called *operational safety*) and reflect the factors inherent in protecting workers against industrial accidents (as opposed to *system safety* or the safety of complex systems). At the beginning, the focus in industrial accident prevention was on unsafe conditions, such as open blades and unprotected belts. While this emphasis on preventing unsafe conditions was very successful in reducing industrial injuries, the decrease naturally started to slow down as the most obvious accident factors were eliminated and the emphasis shifted to unsafe human acts: Accidents were regarded as someone's fault rather than as an event that could have been prevented by some change in the system.

Heinrich's Domino Model, published in 1931, was very influential in shifting the emphasis to human error. Heinrich compared the general sequence of accidents to five dominoes, standing on end in a line (Figure 1.1). When the first domino falls, it automatically knocks down its neighbor and so on until the injury occurs. In any accident sequence, according to this model, ancestry or social environment leads to a fault of a person, which is the proximate reason for an unsafe act or condition (mechanical or physical), which results in an accident, which leads to an injury. Note the second domino implies that all accidents result from a "fault of a person."

One of the problems in evaluating the role of human error in accidents is simply defining it. Accident models that focus only on direct precursor events to an accident, usually consider only operator error and maintenance error and not designer or manager error. Even then, the definition of human error is problematic.

Operator error is usually defined in terms of deviation from a norm. That is, a definition of normative behavior is established, perhaps in terms of defined procedures or normative behavior, and any deviation is labeled as erroneous. The problem with this definition is that operators almost always deviate from defined procedures in order to accomplish their tasks. Defined procedures cannot consider all local contingencies of the work context.

> In studies of operations, modification of instructions is repeatedly found and operators' violations of rules appears to be quite rational, given the actual work load and timing

> constraints. One implication in the present context 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, and who is, therefore, likely to be exposed to punishment [36].

In summary, any particular condition in a complex technological system is unlikely to be either necessary nor sufficient to cause an accident. The high frequency of accidents having complex causes probably results from the fact that competent engineering and organizational structures eliminate the simpler causes. On the positive side, the very complexity of accident processes means that there may be many opportunities to intervene or interrupt them. Therefore, thorough consideration of the conditions leading to accidents will be more useful than simplistic explanations.

1.3 Chains of (Time-Ordered) Events

The most common accident models include multiple events related by a forward chain over time. The events considered almost always involve some type of component failure, human error, or energy-related event. There may be other relationships represented by the chain in addition to a chronological one, but any relationship is almost always a direct, linear one represented by the notion that the preceding event or condition must have been present for the subsequent event to occur, i.e., if event X had not occurred, event Y would not have occurred.

Various events in the chain may be given labels such as proximate cause, primary cause, basic cause, contributory cause, root cause, etc. Unsafe conditions may be included in the chain or may be represented as factors that link events, i.e., an unsafe condition plus an event may lead to another event or an event may create an unsafe condition that leads to an event, etc. For example, rain (event) may lead to a wet road (condition) that may lead to a driver losing control (event). Whether the beginning point is an event or a condition simply reflects an arbitrary decision about where to stop the backward chaining.

The chain of events may be a simple chain or may include converging chains or even parallel chains. The point of convergence represents the interaction between independent event chains.

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. Pate-Cornell provides an example for an offshore oil platform:

> An initiating event is an event that triggers an accident sequence—e.g., a wave that exceeds the jacket's capacity that, in turn, triggers a blowout that causes failures of the foundation. As initiating events, they are mutually exclusive; only one of them starts the accident sequence. A catastrophic platform failure can start by failure of the foundation, failure of the jacket, or failure of the deck. These initiating failures are also (by definition) mutually exclusive and constitute the basic events of the PRA [Probabilistic Risk Assessment] model in its simplest form [32, p.121].

Note that although the accident sequence in this example is assumed to start with a particular event (in this case a wave), the reason for this failure is not included nor are ancillary reasons for other failures in the sequence. For example, the wave may be traced back to an earthquake or a storm. The failure of the foundation might be caused by the use of inferior construction materials which, in turn, may be related to lack of government oversight or budgetary problems.

In fact, the stopping point in tracing back events preceding an accident is arbitrary and may be chosen because (1) it includes events that are familiar and thus acceptable as explanations for the accident, it includes events for which it is felt something can be done for correction, or it excludes

prior events omitted for various political or other subjective reasons. The "root cause" identified in the chain of events usually has the following characteristics: (a) it is a deviation from a standard, (b) it is accepted as a familiar and therefore reasonable explanation, and (c) a "cure" is known [24:11]. The backward chain may also stop simply because information is missing about previous events or conditions.

> This means that allocations of causes to people or technical parts in the system is a purely pragmatic question regarding the stop rule applied for analysis after the fact. There is no well-defined 'start' of the causal chain involved in accidents, and the link chosen to represent the 'cause' and the 'error data' collected depends on the application of that data. [11:24]

Stop rules are not usually formulated explicitly and involve pragmatic and subjective decisions. The selection of the stop rule applied depends at least in part on familiarity and the availability of a cure as well as some of the more subjective reasons described in Section 2.1 and the role of the person making the decision (e.g., operator, supervisor, designer, or judge).

The stop rule applied will also depend on the objective of the analysis, e.g., to explain the course of events, to allocate responsibility or blame, or to identify possible improvements to avoid future accidents. Thus, the stop rule and events included in the chain may differ depending on the goal of the analysis or the analyst.

When the goal is to perform a probabilistic risk assessment (PRA), initiating events in the chain are usually assumed to be mutually exclusive (perhaps to simplify the mathematics). The only events considered are failure events—design errors are usually omitted and only come into the calculation in the probability of the failure event.[1] In the offshore oil platform example above, the probability density function (pdf) for the failure of the deck might reflect a poor design for the conditions the deck must withstand (a human design error) or the use of inadequate construction materials due to lack of government oversight or project budget limitations. None of these indirect factors is reflected directly in the PRA although certainly it is possible (and obviously desirable) for the PRA to include a description of the conditions under which the probabilities were derived. If not included, it may not be possible to determine whether conditions in the platform being evaluated differ from those built previously that might significantly alter the risk. The introduction of a new design feature or of active control by a computer, for example, might greatly affect the probability of failure.

Not only is the selection of the events and conditions included in the chain subjective, but there is subjectivity also in the links. Leplat notes that the links are justified by knowledge or rules of different types, including physical and organizational knowledge. The same event can give rise to different types of links according to the mental representations the analyst has of the production of this event. When several types of rules are possible, the analyst will remember those that agree with his or her mental model of the situation [20]. For example, in the Cali aircraft accident described in Chapter 3, two significant events are (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 (more detail is provided in Chapter 3):

[1] Accidents involving dysfunctional interactions among non-failing (operational) components are usually not considered.

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

- *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 flightpath.
- *Map Deficiency*: The identifier used to identify Rozo on the approach chart 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.
- *International Standards Deficiency*: ICAO did not resolve conflicts between map and FMS-naming standards.

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.

The countermeasures to prevent accidents considered as chains of events usually involve either removing the 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. Thus, the emphasis is on eliminating events or on breaking the accident sequence.

1.4 Hierarchical Models

If our goal is to better understand accident processes in order to determine how to engineer systems to prevent similar occurrences, then this goal might be achieved most effectively not by eliminating direct causes but by eliminating or manipulating other indirectly related factors. Achieving this goal requires that the accident model used must not limit our consideration of the factors affecting the loss event.

A problem with most chain-of-events models is that factors other than simple failure events and conditions, are difficult to incorporate. Important systemic factors might include structural deficiencies in the organization or factors related to the safety culture in the industry. In the seventies, Johnson proposed a model and sequencing method that describes accidents as patterns of direct events and causal factors arising from contributory factors, which in turn arise from systemic factors (Figure 1.2) [11].

A hierarchical model with slightly different types of information at the first two levels ([24]) is used to assist in understanding the accidents in Chapter 3. In contrast to Johnson's model where the levels represent different types of factors, each level here provides a different model of the accident causation at a different level of abstraction. In this general organization of causality (shown in Figure 1.3), the lowest level (Level 1) describes the mechanism of the accident—the chain of events. For example, an object appeared in front of the car, the driver hit the brake, the car skidded and hit the tree, the driver was thrown from the car and injured.

The second level of understanding causality (Level 2) includes the conditions or lack of conditions that allowed the events at the first level to occur. For example, the driver does not know how to prevent or stop the skid, the car is not equipped with anti-lock brakes or the driver uses them inappropriately, the driver was driving too fast, the street was wet from rain and thus friction was reduced, visibility was poor, the driver was not wearing a seat belt, the seat belt was ineffective. At

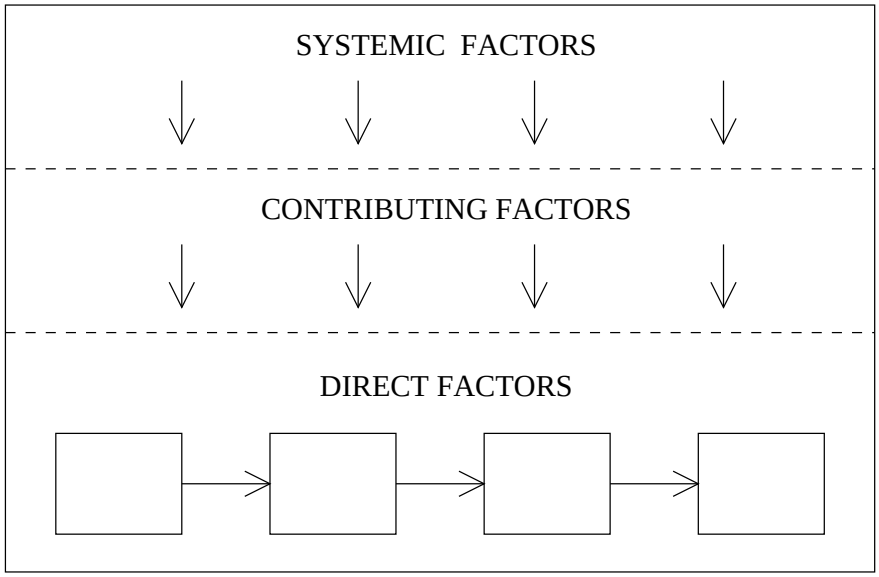

Figure 1.2: Johnson's Three Level Model of Accidents

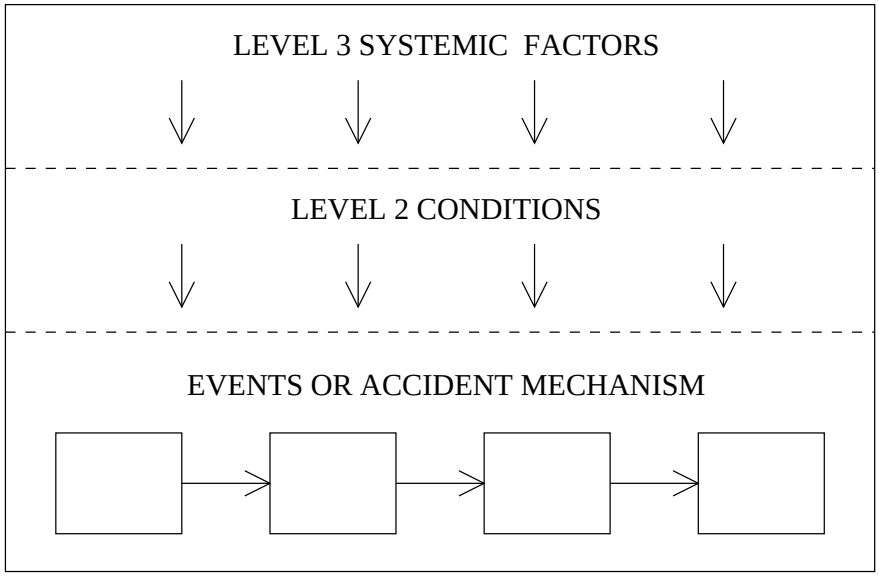

Figure 1.3: The Hierarchical Model Used in This Paper

this level, the cause or causes may be overspecified; not all conditions may have to be met before the accident will occur.

The factors at the third level are often referred to as the *root causes* or *systemic factors* of an accident. Systemic factors affect general classes of accidents; they are weaknesses that not only contributed to the accident being investigated but also can affect future accidents. Responses to accidents tend to involve fixing only a specific causal factor while leaving the more general or systemic factors untouched. Blame is more likely to be placed on operator errors or on specific component failures than on poor training, lack of general hazard controls, or management deficiencies. However, because accidents rarely repeat themselves in exactly the same way, patches to particular parts of the system may be ineffective in preventing future accidents. For example, if another Space Shuttle accident like that of *Challenger* occurs, it is unlikely to be caused by the exact same type of O-ring problem: An O-ring failure precipitated the accident, but the root causes identified in the accident investigation were related to organizational deficiencies. Preventing accidents in the future requires fixing those deficiencies.

The release of methyl isocyanate (MIC) from the Union Carbide chemical plant in Bhopal India in December 1984 has been called the worst industrial accident in history: Conservative estimates point to 2000 fatalities, 10,000 permanent disabilities (including blindness), and 200,000 injuries. After the accident, Union Carbide management blamed it on human error (or perhaps sabotage)—the improper cleaning of a pipe at the plant. A relatively new worker was assigned to wash out some pipes and filters, which were clogged. MIC produces large amounts of heat when in contact with water, and the worker properly closed the valves to isolate the MIC tanks from the pipes and filters being washed. Nobody, however, inserted a required safety disk (called a slip blind) to back up the valves in case they leaked. The worker who had been assigned the task of washing the pipes reportedly knew that the valves leaked, but he did not check to see whether the pipe was properly isolated because, he said, it was not his job to do so. Inserting the safety disks was the job of the maintenance department, but the maintenance sheet contained no instruction to insert this disk. The pipe-washing operation should have been supervised by the second shift supervisor, but that position had been eliminated in a cost-cutting effort.

A chain of events describing the accident mechanism for Bhopal might include:

E1 Worker washes pipes without inserting slip blind.

E2 Water leaks into MIC tank.

E3 Explosion occurs.

E4 Relief valve opens.

E5 MIC vented into air.

E6 Wind carries MIC into populated area around plant.

Stopping here, after identifying a single human operator error as the root event would be a mistake: Given the design and operating conditions of the plant, it seems that an accident was waiting to happen:

> However [water] got in, it would not have caused the severe explosion had the refrigeration unit not been disconnected and drained of freon, or had the gauges been properly working and monitored, or had various steps been taken at the first smell of MIC instead of being put off until after the tea break, or had the scrubber been in service, or had the water sprays been designed to go high enough to douse the emissions, or had

the flare tower been working and been of sufficient capacity to handle a large excursion [34, p.349].

Assigning probabilities to all these seemingly unrelated events and assuming independence would lead one to believe that this accident was merely a matter of a once-in-a-lifetime coincidence. On the surface, it does appear incredible that the vent scrubber, flare tower, water spouts, refrigeration unit, and various monitoring instruments were all out of operation simultaneously. However, a closer look shows a different picture and points more to design and management errors than random failure.

It is not uncommon for a company to turn off passive safety devices, such as refrigeration units, to save money, and gauges at plants are frequently out of service [2]. At the Bhopal facility, there were few alarms or interlock devices in critical locations that might have warned operators of abnormal conditions.

The operating manual specified that the refrigeration unit *must* be operating whenever MIC was in the system: The chemical has to be maintained at a temperature no higher than 5° Celsius to avoid uncontrolled reactions. A high temperature alarm was to sound if the MIC reached 11°. The refrigeration unit was turned off, however, to save money and the MIC was usually stored at nearly 20°. The plant management adjusted the threshold of the alarm, accordingly, from 11° to 20° and logging of tank temperatures was halted, thus eliminating the possibility of an early warning of rising temperatures.

Other protection devices at the plant had inadequate design thresholds. The vent scrubber, had it worked, was designed to neutralize only small quantities of gas at fairly low pressures and temperatures: The pressure of the escaping gas during the accident exceeded the scrubber's design by nearly two and a half times, and the temperature of the escaping gas was at least 80° Celsius more than the scrubber could handle. Similarly, the flare tower (which was supposed to burn off released vapor) was totally inadequate to deal with the estimated 40 tons of MIC that escaped during the accident [2].

Alarms at the plant sounded so often (the siren went off 20 to 30 times a week for various purposes) that an actual alert could not be distinguished from routine events or practice alerts. Ironically, the warning siren was not turned on until two hours after the MIC leak was detected and then was turned off after only five minutes—which was company policy [1]. Moreover, all the practice alerts did not seem to be effective in preparing for an emergency: When the danger during the release became known, many employees ran from the contaminated areas of the plant, totally ignoring the buses that were sitting idle ready to evacuate nearby residents. Plant workers had only a bare minimum of emergency equipment—a shortage of oxygen masks, for example, was discovered after the accident started—and they had almost no knowledge or training about how to handle nonroutine events.

The police were not notified when the chemical release began; in fact, when called by police and reporters, plant spokesmen first denied the accident and then claimed that MIC was not dangerous. Nor was the surrounding community warned of the dangers, before or during the release, or informed of the simple precautions that could have saved them from lethal exposure, such as putting a wet cloth over their face and closing their eyes. If the community had been alerted and provided with this simple information, many (if not most) lives would have been saved and injuries prevented [18].

These are only a few of the factors involved in this catastrophe, which include other technical and human errors within the plant, design and management negligence, regulatory deficiencies on the part of the U.S. and Indian governments, and general agricultural and technology transfer policies (which relate to the reason they were making such a dangerous chemical in India in the

first place). Any one of these perspectives or "causes" is inadequate by itself to understand the accident and to prevent future ones. Identifying only operator error (failing to insert the slip blind) as the cause of the accident ignores most of the opportunities for prevention. In fact, after the Bhopal disaster, both Union Carbide and the U.S. Occupational Safety and Health Administration (OSHA) announced that the same type of accident could not occur at Union Carbide's plant in Institute, West Virginia, which also makes MIC. Eight months later, however, a similar accident occurred at the Institute plant that led to brief hospital stays for approximately 100 people. The consequences were less serious in this second accident only because of such incidental factors as the direction of the wind and the fact that the tank happened by chance to contain a less toxic chemical at the time.

In most systems, where some care has been taken in their design, accidents or hazards will depend on a multiplicity of causal factors and each event on a complex combination of conditions, including technical, operator, and organizational or societal conditions and actions. For example, design errors in the DC-10 caused accidents only after millions of flight hours. For such accidents to occur, the conditions involved necessarily must be very rare or must be the result of a complex interaction and coincidence of factors; preventing any of these conditions might eliminate the accidents or reduce the consequences. However, this also implies that any simple model of accidents will be flawed and may limit our ability to understand past accidents and to prevent future ones in the most cost–effective way.

In *Friendly Fire: The Accident Shootdown of U.S. Black Hawks over Northern Iraq*, Scott Snook writes[3]:

> There weren't any bad guys; hence, no one to blame. There weren't any catastrophic failures of material or equipment; hence, nothing to fix. No gross negligence or act of God caused this tragedy. The more I looked for traditional culprits, the more I realized that this accident occurred not because something extraordinary had happened, but rather just the opposite. This accident happened because of, or perhaps in spite of, everyone behaving just the way we would expect them to behave, just the way theory would predict—given a clear understanding of the circumstances. Indeed, this accident was 'normal', not only in the sense that Perrow suggests—'that it [was] an inherent property of the system.' But rather it was normal because it occurred as the result of normal people behaving in normal ways in normal organizations. ...
>
> If no one did anything wrong; if there were no unexplainable surprises at any level of analysis; if nothing was abnormal from a behavioral and organizational perspective; then what have we learned?
>
> Perhaps the fundamental lesson is that as basic scientists, we do know quite a bit about individual, group, and organizational behavior. However, as applied theorists, we know much less about how complex untoward events cut across levels of analysis and time; and as practitioners, we seem largely incapable of either identifying or recognizing general sets of conditions that increase the likelihood of failures at all levels. ...
>
> Hopefully this study illustrates the limitations of adopting such cause-and-effect approaches when trying to understand untoward events in complex organizations. ... How far back up the causal stream are we willing to swim? ... What it depends on is who is conducting the search and why. Technicians stop when they find something broken that they can fix. Trainers stop when they find a weak skill that they can train. Lawyers stop when they find a responsible individual that they can prosecute. Political leaders stop when their constituents stop. Scientists stop when they learn something new.

[3]Thanks to Michael Holloway at NASA Langley for providing this quotation.

Chapter 2

Limitations in Using Accident Reports to Understand Accidents

Before evaluating recent aerospace accident reports using these models, it is important to understand the limitations of such reports and of what can be learned from them about how to prevent accidents.

Investigating accidents to identify causal factors in order to prevent future repetition, the so-called *fly-fix-fly* approach to safety engineering, has been extremely successful in decreasing the accident rate in civil aviation. Some reasons for this are the use of standard designs and the slow pace of technological innovation. When technology changes rapidly, however, or when radical new designs are introduced, the effectiveness of learning from past failures is more limited. One of the mistakes noted in the Titan IVB accident (see section 3.5) was that the quality assurance engineers based their risk analysis not on determining the steps critical to mission success, but instead focused on those problems that had occured in past launches and thus missed the new problem that led to the accident.

In addition, the fly-fix-fly approach is dependent on the data reported by accident investigators and the causal factors recorded. The applicability of the results of accident reports to current or future systems also depends on whether the relevant circumstances (both within the system and in its environment) are similar, but the necessary information to make such a determination is often omitted from accident reports.

None of these limitations means that studying accident reports cannot be useful, simply that there are serious limitations in the conclusions that can be drawn about general accident causation from such reports.

To formulate hypotheses about the cause of an accident, investigators usually collect data by questioning individuals and by gathering and examining other evidence. Both of these sources of data may involve filtering and subjectivity and the collection process itself can limit the information acquired.

2.1 Filtering

Rarely are all the causes of an accident perceived identically by everyone involved including engineers, managers, operators, union officials, insurers, lawyers, politicians, the press, the state, and the victims and their families. Such conflicts are typical in situations that involve normative, ethical, and political considerations on which people may legitimately disagree. Some conditions may be considered unnecessarily hazardous by one group yet adequately safe and necessary by another.

In addition, judgments about the cause of an accident may be affected by the threat of litigation or by conflicting interests.

Research data validates this hypothesis. Various studies have found causal attribution to depend on characteristics of the victim and of the analyst (e.g., hierarchical status, degree of involvement, and job satisfaction) as well as on the relationships between the victim and the analyst and on the severity of the accident [22].

For example, one study found that workers who were satisfied with their jobs and who were integrated into and participating in the enterprise attributed accidents mainly to personal causes. In contrast, workers who were not satisfied and who had a low degree of integration and participation more often cited nonpersonal causes that implied that the enterprise was responsible. Another study found differences in the attribution of accident causes among victims, safety managers, and general managers. Other researchers have suggested that accidents are attributed to factors in which the individuals are less directly involved. A further factor may be position in the organization: The lower the position in the hierarchy, the greater the tendency to blame accidents on factors linked to the organization; individuals who have a high position in the hierarchy tend to blame workers for accidents [22]. There even seem to be differences in causal attribution between accidents and incidents: Data on near-miss (incident) reporting suggest that causes for these events are mainly attributed to technical deviations while similar events that result in losses are more often blamed on human error [7, 14]. Examining physical evidence may not be any less subjective.

Other factors affecting causal filtering may be related to the design of the reporting system itself. For example, the NASA Aviation Safety Reporting System (ASRS) has a category that includes non-adherence to FARs (Federal Aviation Regulations). In a study of reported helicopter incidents and accidents over a nine-year period, this category was by far the largest category cited [10]. The predominance of FAR violations in the incident data may reflect the motivation of the ASRS reporters to obtain immunity from perceived or real violations of FARs and not necessarily the true percentages.

Filtering and bias in accident reports can occur due to individual interpretations of events, both by the individuals involved in the events and by the accident analysts. Events and actions involve both motives and goals. While the basic events are usually not subject to interpretation (although they may be filtered or distorted in the selection or description process), explanations of actions that include goals and motives always involve some type of interpretation. Individuals may be unaware of their actual goals and motivation or may be subject to various types of pressures to reinterpret their actions. Explanations by analysts not involved in the events may be influenced by their own mental models or additional goals and pressures.

Note the difference between an explanation based on goals and one based on motives. Consider the basic events of an accident that occurs while driving a car during a snow storm and sliding into a telephone pole. This explanation represents the basic chain of events. An explanation based on goals would include the fact that the driver wanted to get home quickly (before the streets had been plowed). An explanation based on motive might include the fact that guests were coming for dinner and the driver had to prepare the food before they arrived.

Both the explanation based on goals and that based on motive, depend on assumptions that cannot be *directly* measured or observed by the accident investigator. Leplat illustrates this dilemma by describing three different motives for the event *"The operator sweeps the floor"*: (1) the floor is dirty, (2) the supervisor is present, or (3) the machine is broken and the operator needs to find other work [23]. Even if the people involved survive the accident, true goals and motives may not be revealed for various reasons.

Causal identification may also be influenced by the data collection methods. Data usually is collected in the form of textual descriptions of the sequence of events of the accident, which tend

to concentrate on obvious conditions or proximal events (those closely preceding the accident in time) and tend to leave out less obvious or direct events and factors. There is no simple solution to this inherent bias: On the one hand, report forms that do not specifically ask for nonproximal events often do not elicit them while, on the other hand, more directive report forms that do request particular information may limit the categories of conditions considered [15]. Another consideration is that coded data allows easy retrieval and computer searches but narrative data provides a richer data source.

For all these varied reasons, accident data is often systematically filtered and unreliable and the causal factors selected for a particular accident may involve a great deal of subjectivity. Note the influence of the accident model here: The analyst's mental representation (model) of accidents may cause him or her to remember or include only those factors that agree with that mental model or belief about how accidents occur [23]. This fact implies that for accident models and their use in accident investigation, it may be helpful to separate the events and factual data from the interpretation of that data (as attempted in the next chapter of this paper). In addition, it may be appropriate to allow presenting (and perhaps even encouraging) differing viewpoints.

2.2 Oversimplification

A second trap in identifying accident causes is oversimplification. Out of a large number of necessary conditions for the accident, one is often chosen and labeled as *the* cause, even though all the factors involved were equally indispensable to the occurrence of the event. A condition may be selected as the cause because it is the last condition to be fulfilled before the effect takes place, its contribution is the most conspicuous, or the selector has some ulterior motive for the selection.

Oversimplification of the causal factors in accidents can be a hindrance in preventing future accidents. Two common oversimplifications are focusing only on human error or on technical failure.

2.2.1 Operator Error as the Cause of Accidents

The most common oversimplification in accident reports is blaming the operator. In any system where operators are involved, a "cause" may always be hypothesized as the failure of the operator to step in and prevent the accident: Virtually any accident can be ascribed to human error in this way. Even when operator error is more directly involved, considering that alone as a cause is too limiting to be useful in identifying what to change in order to increase safety most effectively.

In the nineteenth century, for example, coupling accidents on railroads were one of the principal causes of injury and death to railroad workers [9]. In the seven years between 1888 and 1894, 16,000 railroad workers were killed in coupling accidents and 170,000 were crippled. Managers claimed that such accidents were due only to worker error and negligence, and therefore nothing could be done aside from telling workers to be more careful. The government finally stepped in and required that automatic couplers be installed. As a result, fatalities dropped sharply. According to the June 1896 issue of Scientific American:

> Few battles in history show so ghastly a fatality. A large percentage of these deaths were caused by the use of imperfect equipment by the railroad companies; twenty years ago it was practically demonstrated that cars could be automatically coupled, and that it was no longer necessary for a railroad employee to imperil his life by stepping between two cars about to be connected. In response to appeals from all over, the U.S. Congress passed the Safety Appliance Act in March 1893. It has or will cost the

> railroads $50,000,000 to fully comply with the provisions of the law. Such progress has already been made that the death rate has dropped by 35 per cent.

It is common to see statements that 70–80% of aircraft accidents are "caused" by pilot error. Another commonly cited statistic is that 85 percent of work accidents are due to unsafe acts by humans rather than unsafe conditions [11]. However, closer examination of the data shows that it may be biased and incomplete: the less that is known about an accident, the more likely it will be attributed to operator error [11]. Thorough investigation of serious accidents almost invariably finds other factors. Perrow cites a U.S. Air Force study of aviation accidents that concludes that the designation of human error, or pilot error, is a convenient classification for accidents whose real cause is uncertain, complex, or embarrassing to the organization. Other reasons for the operator error statistics include: (1) operator actions are generally reported only when they have a negative effect on safety and not when they are responsible for preventing accidents; (2) blame may be based on unrealistic expectations that operators can overcome every emergency; (3) operators may have to intervene at the limits of system behavior when the consequences of not succeeding are likely to be serious and often involve a situation the designer never anticipated and was not covered by the operator's training; and (4) hindsight often allows us to identify a better decision in retrospect, but detecting and correcting potential errors before they have been made obvious by an accident is far more difficult. The report on the Clapham Junction railway accident in Britain concluded:

> There is almost no human action or decision that cannot be made to look flawed and less sensible in the misleading light of hindsight. It is essential that the critic should keep himself constantly aware of that fact. [Hidden, A. (Chairman), pg. 147]

All human activity takes place within and is influenced by the environment, both physical and social, in which it takes place. Operator error cannot be understood or prevented without understanding the environmental factors that influence those actions. It is often very difficult to separate design error from operator error. In highly automated systems, the operator is often at the mercy of the system design and operational procedures. One of the causes ascribed to the Cali American Airlines 757 accident was "failure of the flightcrew to revert to basic radio navigation at the time when the FMS-assisted navigation became confusing and demanded an excessive workload in a critical phase of flight." Such simplistic descriptions of human error take attention away from the reasons why the design of the automation was confusing or required an excessive workload at a critical flight phase.

Pilots may be blamed for accidents without examining the organizational factors that impacted their poor decision making. For example, one factor often implicated in weather-related commuter aircraft accidents is that pilots, who must make decisions about taking off in poor weather, are often not paid if they do not fly. Pilot contracts with major commercial carriers reduce the problem somewhat by ensuring that they will be paid even if the flight is canceled. Obviously there are other pressures and the issues should not be oversimplified, but unless all the factors underlying decision making are identified, the cause of accidents may not be understood well enough to prevent future ones.

During and after World War II, the Air Force had serious problems with aircraft accidents—for example, from 1952 to 1966, 7715 aircraft were lost and 8547 people killed [9]. Most of these accidents were blamed on pilots. Some aerospace engineers did not believe the cause was so simple and started to argue that safety must be designed and built into aircraft just as are performance, stability, and structural integrity. Although a few seminars were conducted and papers written about this approach, the Air Force did not take it seriously until they began to develop intercontinental ballistic missiles: there were no pilots to blame for the frequent and devastating explosions of these

liquid-propellant missiles. In having to confront factors other than pilot error, the Air Force began to treat safety as a system problem, and system safety engineering programs were developed to deal with them.

Because the role of operator error in accidents is so important, it must necessarily play a central role in any comprehensive accident model, but we must take care that it does not become the *only* or *primary* factor considered.

2.2.2 Physical Failures as the Cause of Accidents

Just as important an oversimplification as blaming the operator is concentrating only on technical failures and immediate physical events. This type of overly narrow focus may lead to ignoring some of the most important factors in an accident. For example, an explosion at a chemical plant in Flixborough, Great Britain, in June 1974, resulted in 23 deaths, 53 injuries, and $50 million in damages to property (including 2,450 houses) up to five miles from the site of the plant.[1] The official accident investigators devoted most of their effort to determining which of two pipes was the first to rupture. The British Court of Inquiry concluded that "The disaster was caused by a coincidence of a number of unlikely errors in the design and installation of a modification," and "Such a combination of errors is very unlikely ever to be repeated" [5]. While these conclusions are true if only specific technical failures and immediate physical events are considered, they are not true if a broader view of accidents is taken.

For example, the pipe rupture at Flixborough was only a small part of the cause of this accident. A full explanation and prevention of future such accidents requires an understanding of the management practices of running the Flixborough plant without a qualified engineer on site and allowing unqualified personnel to make important modifications to the equipment, of making engineering changes without properly evaluating their safety, and of storing large quantities of dangerous chemicals close to potentially hazardous areas of the plant [4]. The British Court of Inquiry investigating the accident amazingly concluded that "there were undoubtedly certain shortcomings in the day-to-day operations of safety procedures, but none had the least bearing on the disaster or its consequences and we do not take time with them." Fortunately, others did not take this overly narrow viewpoint, and Flixborough led to major changes in the way hazardous facilities were allowed to operate in Britain.

Considering only immediate physical failures as the causes of accidents can allow latent design errors to go uncorrected and to be repeated. An example, which is a common but dangerous practice judging from its implication in a surprising number of accidents, is wiring a valve to detect only that power has been applied to open or close it and not that the valve has actually operated. This design "flaw" has been implicated in a surprising number of accidents. For example, one Air Force system included a relief valve opened by the operator to protect against overpressurization. A secondary valve was installed as backup in case the primary relief valve failed. However, the operator must know if the first valve did not open so that the second valve can be activated. An explosion occurred when on one occasion, the position indicator and open indicator lights both illuminated but the primary relief valve was *not* open. A post-accident investigation discovered that the indicator light circuit was wired to indicate *presence of power* at the valve, but it did not indicate valve *position*. Thus, the indicator showed only that the activation button had been pushed, not that the valve had operated. An extensive probabilistic risk assessment of this design had included a low probability of simultaneous failure for the two relief valves, but had ignored the

[1] Many more lives might have been lost had the explosion not occurred on a weekend when only a small shift was at the plant, the wind was light, and many of the nearby residents were away at a Saturday market in a neighboring town.

possibility of a design error in the electrical wiring: The probability of the design error was not quantifiable. If it had been identified, of course, the proper solution would have been to eliminate the design error, not to assign a probability to it. The same type of design error was a factor in the Three Mile Island accident: An indicator misleadingly showed that a discharge valve had been ordered closed but not that it had actually closed. In fact, the valve was blocked in an open position. With the increasing role of software in complex systems (and considering that the only way software can contribute to accidents is through a design error), concentrating on physical failures and the use of redundancy to prevent them will become increasingly ineffective.

Large-scale engineered systems are more than just a collection of technological artifacts: They are a reflection of the structure, management, procedures, and culture of the engineering organization that created them, and they are also, usually, a reflection of the society in which they were created. Accidents are often blamed on operator error or equipment failure without recognition of the industrial, organizational, and managerial factors that made such errors and defects inevitable. The causes of accidents are frequently, if not almost always, rooted in organizational culture, management, and structure. These factors are all critical to the eventual safety of the engineered system. Oversimplifying the factors involved in accidents limits our ability to prevent them.

Chapter 3

Accidents

With these caveats about generalizing from accident reports in mind, we start by examining a set of reports on aerospace accidents: Ariane 501; Space Shuttle Challenger; Mars Climate Orbiter; Mars Polar Lander; Titan IV/Milstar; American Airlines B-757 near Cali, Columbia; Lufthansa A320 at Warsaw, and China Airlines A320 at Nagoya, Japan. With the exception of Challenger, all involved software in some way. The Challenger accident is included for comparison purposes—the Rogers Commission report describes an unusually thorough and comprehensive accident investigation.

3.1 Ariane 5

3.1.1 Background

On June 4, 1996, the maiden flight of the Ariane 5 launcher ended in failure: Only about 40 seconds after initiation of the flight sequence, at an altitude of only 2700 m, the launcher veered off its flight path, broke up, and exploded. The accident report describes the accident in terms of a chain of technical events, their inter-relations, and causes.

To understand the events, some background is necessary. The Flight Control System of the Ariane 5 is of a standard design, and much of the software built for the Ariane 4 was reused on the Ariane 5.

The attitude of the launcher and its movements in space are measured by an Inertial Reference System (SRI, using the French acronym). The SRI has its own internal computer, in which angles and velocities are calculated on the basis of information from a strap-down inertial platform, with laser gyros and accelerometers. The data from the SRI are transmitted through the databus to the On-Board Computer (OBC), which executes the flight program and controls the nozzles of the solid boosters and the Vulcain cryogenic engine, via servovalves and hydraulic actuators.

In order to improve reliability, there is considerable redundancy at the hardware level. There are two SRIs operating in parallel, with identical hardware and software. One SRI is active and the other is in "hot" standby[1]. If the OBC detects that the active SRI has failed, it immediately switches to the standby unit, provided that this backup unit is functioning properly. Likewise, there are two OBCs, and a number of other units in the Flight Control System are also duplicated.

Exception handling was used to improve software reliability and fault tolerance. To oversimplify somewhat, when the computer hardware or operating system detects an error (called an *exception* or *fault*) in the application software, it usually stops the execution of the application software.

[1] A hot standby operates in parallel with the primary unit but its outputs are ignored.

Obviously, there are good reasons to stop as most of the time continuing will result in bad consequences. When *exception handling* is used in the application software, instead of just stopping its execution, the hardware or operating system can signal to the application software that a fault or exception has occurred. This is called *raising an exception*, and the type of exception raised provides some limited information about what type of error occurred. The types of exceptions that can be detected are primarily faults confined to a single instruction, such as an overflow (e.g., bits have been lost in a variable value as the result of an arithmetic operation or from assigning a variable value to a location that does not have enough bits to hold the entire value), an attempt to divide by zero, or an attempt to access a variable in protected memory. In order to *handle* the signaled exception, the application program must include special exception-handling code that has been designed to detect the signal and perform some repair. If no exception-handling code is included or if the repair attempt is unsuccessful, execution of the application software usually is stopped. Alternatively, exception-handling software may contain logic to continue execution, perhaps simply skipping the erroneous instructions.

The losses from the accident included the $5 billion payload, which in accord with common practice for test flights was not insured, as well as other unknown financial losses. The accident also had repercussions in the form of adding two qualification flights that had not been planned, pushing back commercial flights by more than 2.5 years, and extending Ariane 4 operations at least until 2003 (it was originally to be phased out in 2000).

3.1.2 Chain of Events

The report describes the following chain of events:

E1: The required prelaunch alignment of the strapdown inertial platform was performed by the inertial reference system (SRI) software.

E2: The SRI Flight Mode was started (as planned for Ariane 5) three seconds before H0 (command for main cryogenic engine ignition).

E3: The alignment function operated for 50 seconds (as required for Ariane 4) after starting the SRI Flight Mode.

E4: The SRI, as designed, sensed the horizontal velocity of the platform and, among other things, calculated the Horizontal Bias (BH) variable, an internal alignment value used as an indicator of alignment precision over time. The conversion of this BH variable from a 64-bit floating point value to a 16-bit signed integer value led to an overflow exception (one of a set signaled as an *Operand Error*).

E5: At H0+36.7 seconds, approximately 30 seconds after liftoff, the backup computer, which was working on hot standby for guidance and attitude control, detected the operand error and turned itself off, as designed to do after detecting corrupted or bad data.

E6: Approximately 0.05 seconds later, the active inertial reference system, whose software and hardware was identical to that of the backup inertial reference system, shut itself down for the same reason. The flight control computer could not switch to the backup inertial reference system as it was already inoperative. With both the primary and backup inertial reference computers shut down, correct guidance and attitude information could not be obtained by the control computer and loss of the mission was inevitable from this point forward.

E7: After the detection of the Operand Error, the active inertial reference system transmitted diagnostic information to the launcher's main computer, where it was interpreted as flight data and used for flight control calculations.

E8: The flight control calculations, using the diagnostic information as flight data, issued a command to the booster nozzles and somewhat later the main engine nozzle to make a large correction for an attitude deviation that had not occurred.

E9: Full nozzle deflections of the solid boosters and the Vulcain main engine led to an angle of attack of more than 20 degrees.

E10: The rapid change of attitude and the high aerodynamic loads stemming from the high angle of attack, created aerodynamic forces that caused the launcher to disintegrate at 39 seconds after H0.

E11: Destruction was automatically initiated upon disintegration, as designed, at an altitude of 4 km and a distance of 1 km from the launch pad. The boosters separated from the main stage, in turn triggering the self-destruct system of the launcher.

3.1.3 Identified Cause of the Accident

The report describes what they call the "primary" cause as:

> The failure of the Ariane 501 was caused by the complete loss of guidance and attitude information 37 seconds after start of the main engine ignition sequence (30 seconds after liftoff). This loss of information was due to specification and design errors in the software of the inertial reference system.
>
> The extensive reviews and tests carried out during the Ariane 5 Development Programme did not include adequate analysis and testing of the inertial reference system or of the complete flight control system, which could have detected the potential failure.

Elsewhere in the report, the "technical cause" of the accident is described as:

> An Operand Error when converting the horizontal bias variable BH, and the lack of protection of this conversion, which caused the SRI computer to stop.

To the report's credit, it does not try to isolate a single cause, but the description of the identified cause does leave out some factors that are included in or implied by the report or potential factors the report does not comment upon but which should have been investigated as noted below.

3.1.4 Hierarchical Model

The chain of events in Section 3.1.2 describes the accident mechanisms. Although the chain of events in the report (and repeated above) is complete with respect to the events that occurred on the launch date and the direct causal relationship between all the events in the chain, it does not provide enough information to understand (except from a physical standpoint) why the accident occurred. An expanded chain of events that goes beyond the launch data would have been very useful in understanding why this accident occurred; the report is not complete enough, however, to reconstruct such a set of events. There is supposedly a non-public, internal report on the Ariane 501 accident that may include the necessary information, but ESA has limited its distribution (perhaps for political reasons) and it is not available outside ESA.

A few events occurring before the day of the flight can be inferred from the causal factors included in the report although a more thorough list of events might have uncovered other problems:

E0-1: During software development, an analysis was performed on every operation that could give rise to an exception, including an operand error, to determine the vulnerability of unprotected code. In particular, the conversion of floating point values to integers was analyzed and operations involving seven variables were determined to be at risk of leading to an Operand Error.

E0-2: Exception handlers were written for four of the seven variables and not for the other three, including the BH variable.

E0-3: External reviews of the software were held (external reviewers are mentioned in the report but no information is provided about what types of reviews were done, who participated, or how they were performed), but the software problem was apparently not found.

E0-4: Simulations and test of the SRI software were performed using the Ariane 4 trajectory data. No test or simulation was performed to verify that the SRI software would behave correctly when being subjected to the countdown and flight time sequence and the trajectory of Ariane 5.

E0-5: System test was performed using simulated output from the SRI and not the SRI computer: The two SRI computers were not present. Although for reasons involving physical laws, it is not possible to test the SRI as a blackbox in the flight environment (without the test being unrealistic), the report notes that it would have been possible to do ground testing by injecting simulated accelerometric signals in accordance with predicted flight parameters while also using a turntable to simulate launcher angular movements. This test was not done.

E1: On the day of the launch, ... (see events E1–E11).

Level 2 Conditions

Level 2 of a hierarchical model contains the conditions that allowed the events or accident mechanisms at Level 1 to occur and that need to be changed to prevent the repetition of a similar accident related to the function of the SRI. Note that the conditions may be overspecified, that is, not all need be present for the accident to occur. Level 2 (and Level 3) were constructed from information present in the accident report. As stated, the basic chain of events in the original accident report (and in Section 3.1.2 and the extended chain created for this report do not provide enough information about the events to understand completely why the accident occurred from a chain-of-events point of view. Therefore, the following analysis includes some speculation, unanswered questions, and incompleteness.

In addition, while the report does provide enough information to prevent a reoccurrence of the exact events for the Ariane 5 (for example, among other things, exception handing software for the BH variable could be added so that the SRI computer does not turn itself off when an Operand Error exception is raised in the alignment function or the SRI software alignment function could be turned off after liftoff as it is not used then), it does not provide enough information to prevent similar but not identical accidents arising from the same systemic problems that led to this accident. Level 2 conditions (direct factors) are those that need to be changed to prevent a repetition of this same accident scenario while Level 3 systemic factors are those that need to be changed to prevent future accidents that have different detailed event scenarios but stem from the same root causes.

The use of a hierarchical model to identify the questions to be asked should be helpful in guiding the accident investigation process. Although I had read and discussed in classes the Ariane 5 report several times previously, new questions arose when I constructed the hierarchical model for this

report that had not arisen before and that were prompted by the attempt to fill in the levels of the model and the mappings between levels.

C1: *Ariane 4 SRI software was reused on the Ariane 5* [↓E3].

The reason for this decision is only described as "for commonality reasons" in the report, which could be interpreted too many ways to get much information out of the statement.

C2: *Logic to satisfy all requirements completely would take more than the maximum workload target of 80% set for the SRI computer* [↓E0-2, E0-1(?)].

An 80% load factor is standard in most embedded systems and is usually considered the maximum acceptable. The extra margin is kept for several reasons, including to allow for mistakes in computation of performance requirements (i.e., to allow a margin of error), adding new requirements and functions, and maintenance and upgrade. To remove this condition requires either reducing the requirements (not considered in the accident report but it should have been) or changing computers (also not mentioned). Often because of the length of the development process for spacecraft, hardware that seemed perfectly adequate in the early design stages or was the most appropriate at hardware selection time is obsolete by the time the system is ready for first launch but at that time it is too late to make changes.

C3: *The software contains code to convert the BH value originally stored as a 64-bit floating point value to a 16-bit integer value.* [↓E4, E0-1]

A significant omission in the report is why the conversion from 64-bit floating point to 16-bit integer was done at all. The potential loss of information in such a conversion would seem on the surface to be a mistake (although it is not called out as such in the report) and would violate standard safe software engineering practice unless a very good reason were involved. One reason for the conversion might be the need for *mixed-type* arithmetic, although conversion of all values to floating point would be standard safe practice in that case. Another reason might be related to the 80% load factor criterion applied to memory usage. Without understanding why the conversion was included, fully understanding why the accident occurred is not possible. It is odd that the accident report does not question this programming decision.

This same type of conversion and loss of precision almost led to the loss of an Intelsat satellite Space Shuttle Endeavor attempted to rendezvous with and repair in May, 1992. The software routine used to calculate rendezvous firings failed to converge to a solution due to a mismatch between the precision of the state-vector variables used to describe the position and velocity of the Shuttle and the limits used to bound the calculation: The state-vector variables were double precision while the limit variables were single precision. Comparing such double-precision and single-precision variables is a questionable coding practice. Mixed-precision assignments and comparisons were used here for the same reason they were probably used on the Ariane 5, that is, limited memory. Although new hardware with increased memory had been installed on the Shuttle, the reason behind a decision not to change these variables from single to double precision at that time was not documented [27]. Like Ariane, a detailed analysis of potential loss of precision in the Shuttle code was not performed before the accident (or incident in this case); due to resource constraints, a decision was made to limit any analysis to safety-critical functions, and the task being performed was not considered to be safety critical.

C4: *The horizontal velocity of the Ariane 4 cannot reach a value beyond the limit of the software (the value will always fit in a 16-bit integer variable) because the trajectory during the first 40 seconds is different.* [↓E4]

The initial acceleration and trajectory of the Ariane 5 leads to a buildup of horizontal velocity five times more rapid than that of the Ariane 4.

C5: *Analysis indicated that three of the seven identified variables were either physically limited or there was a large margin of safety.* [↓E0-2].

The report says that a decision was made jointly by the Ariane 5 project partners at several contractual levels to provide "protection" for four of the variables while the other three were left unprotected. However, there is no explanation in the report of what they meant by "protection" or even if that goal was more specifically defined. The report seems to be equating protection with exception-handling, but an exception handler is not the only way, and in fact is not the best way in this case, to protect against overflow. A much simpler solution would simply be to check the value of the 64-bit floating point variable in the application code before the conversion to determine whether it was larger than the maximum value that could be held by a 16-bit integer (i.e., prevent the problem rather than try to handle it only after it occurs). In fact, library functions exist in Ada (the programming language used) that provide the maximum integer value that can be stored for the hardware platform on which the software is executing so that a test could have been performed quite easily. Such checks are often needed for real-time, embedded software (the application for which Ada was designed), and this type of protection falls into the category of basic defensive programming. There is no mention in the report about whether such a check was attempted in the code and failed to detect the overflow or was not included. Checking for potential overflow is so simple and fast in Ada, there does not seem to be any obvious reason why it would not be performed even if it had been determined through analysis that the variables would never be that large. Defensive programming should be standard practice in critical, real-time software.

C6: *Ariane 5 trajectory data was not used to analyze the behavior of the three unprotected variables.* [↓E4]

The report states "there is no evidence that any trajectory data was used to analyze the behavior of the three unprotected variables." However, this statement seems to imply that there also is no evidence that trajectory data was *not* used. It could have been used and the analysis either was incorrect and/or the wrong data was used. The report is unclear on this point. But given that the specifications included only Ariane 4 trajectory data, it is very unlikely that the Ariane 5 data (as opposed to Ariane 4 data) was used even if trajectory data *was* analyzed.

C7: *The justification for the decision to leave three variables unprotected was not included directly in the source code and was not part of any external reviews* [↓E0-3].

The report states that:

> No reference to justification of this decision was found directly in the source code. Given the large amount of documentation associated with any industrial application, the assumption, although agreed, was essentially obscured, though not deliberately, from any external review.

The report is ambiguous here: Was the assumption itself recorded somewhere, but not in the source code? Was the rationale for the assumption recorded? Neither?

C8: *A useful but not necessary feature was included in the Ariane 4 software to restart the countdown without waiting for normal alignment* [↓E3].

The alignment function computes meaningful results only before liftoff—during flight, the function serves no purpose:

> Alignment of mechanical and laser strap-down platforms involves complex mathematical filter functions to properly align the x-axis to the gravity axis and to find north direction from Earth rotation sensing. The assumption of preflight alignment is that the launcher is positioned at a known and fixed position. Therefore, the alignment function is totally disrupted when performed during flight, because the measured movements of the launcher are interpreted as sensor offsets and other coefficients characterising[sic] sensor behaviour.

Normal alignment takes 45 minutes or more. The alignment function in the Ariane 4 was designed to cope with the unlikely event of a hold in the Ariane 4 countdown, such as between liftoff minus 9 seconds when flight mode starts in the Ariane 4 SRI and liftoff minus 5 seconds when certain events are initiated in the launcher that would take hours to reset. The alignment function continues to operate for 50 seconds after the start of flight mode to allow enough time for the ground equipment to resume control in case of a hold. With this feature, the countdown could be restarted and a short launch window could still be used. The feature was used once in 1989 on Flight 33 of the Ariane 4. The Ariane 5 has a different preparation sequence, and thus cannot use the feature at all.

While there may be good reason for not wanting to rewrite or change software that is being reused (and potentially introduce errors), all such functions at a minimum need to be analyzed for their impact on safety when reused. The report does question the rationale behind allowing the alignment function to operate after the launcher has lifted off by including a recommendation that the alignment function of the SRI be switched off immediately after liftoff and, more generally, recommending that no software function should execute during flight unless it is needed.

It does not question, however, whether such unnecessary but useful features as were included to allow the alignment function to operate after flight mode was initiated should be included in the first place. There is always a tendency in software development for *creeping featurism* or the addition of more and more useful features. Including features that are not critical, however, adds to software and system complexity, makes testing and analysis harder, and degrades the safety of reusing the software (or at least adds a very costly, difficult, and error-prone analysis requirement when it is reused).

In addition, any system including both critical and non-critical functions should have firewalls between them (in this case between the alignment function and the critical attitude control functions) to ensure that the critical functions are not affected by errors in the non-critical code. This is a basic safety engineering principle (see Safeware [24], but its violation seems not to have been investigated, or at least not enough information is included in the report to determine whether the designers did not attempt to provide such separation or whether it was attempted and was unsuccessful.

C9: *The SRI is designed to shut itself down after an error is encountered* [↓E5, E6].

Restart of the inertial reference system, once it has shutdown, is not feasible because attitude is too difficult to recalculate after a processor shutdown. The report states that "the specification said in event of any kind of exception the failure should be indicated on the databus,

the failure context should be stored ..., and finally the SRI processor should be shut down." The report further says that it was "the decision to shut down the processor operation that finally proved fatal" and recommends that no sensors (such as the inertial reference system) should be allowed to stop sending best effort data.

It is always dangerous to second guess the investigators' conclusions from limited information, but it seems that this recommendation should be followed only if the fault (as in this case) does not include critical data. In many cases, sending known bad results may be as bad, or even worse, than reporting the inability to provide good data, i.e., the "best effort" data may be unsafe.

It is unclear, however, why the engineers would opt to shut down the entire inertial guidance system when an exception was raised in a non-critical function (the alignment function after liftoff). The report suggests

> The reason behind this drastic action [of shutting down the computer] lies in the culture within the Ariane programme of only addressing random hardware failures. From this point of view, exception—or error—handling mechanisms are designed for a random hardware failure which can quite rationally be handled by a backup system.

This type of omission might arise in two ways:

1. Not appreciating the different failure modes of software (design errors versus random failures), i.e., a backup computer running identical software does not provide any protection against software design errors.
2. Simply assuming that software errors will be eliminated during test and thus no additional protection is needed. This assumption is an *extremely* common (but obviously erroneous) assumption among engineers.

C10: *Neither the SRI requirements nor specification included Ariane 5 trajectory data* [↓E0-3, E0-4].

The report states that this was a choice—it was jointly agreed upon by the developers—and was not simply an inadvertent omission. There is no explanation given, however, for this decision. The decision is so strange that there must have been some rationale for it.

C11: *The SRI system specification does not indicate operational restrictions that emerge from the chosen implementation and thus should be considered during test* [↓E0-4].

Such limitations should always be included in the specifications for real-time, embedded systems. The accident report notes that this information was missing.

C12: *The simulated output of the SRI was used in system test and system simulations, rather than using the SRI itself or its detailed simulation* [↓E0-5].

Because of physical laws, it is not possible to test the SRI as a blackbox in the flight environment, unless one does a completely unrealistic flight test. *It is possible*, however, to do ground testing by injecting simulated accelerometric signals in accordance with predicted flight parameters, while also using a turntable to simulate launcher angular movements. The report concludes that had such a test been performed by the supplier or as part of the acceptance test, the failure mechanism would have been exposed (although "could have been exposed" would have been a safer conclusion).

The report also identifies the reason for the lack of such testing as the system designers assuming that:

- The SRI was considered to be fully qualified by unit testing and by previous use on Ariane 4.
- The precision of the navigation software in the flight control computer (OBC) depends on the precision of the SRI measurements, but in the system test facility this precision could not be achieved by the electronics creating the test signals.
- The simulation of failure modes is not possible with real equipment, but only with a model. (Note that this argument again assumes hardware-type failures, not logic errors—the latter should be detectable during simulation.)
- The base period of the SRI is 1 millisecond versus 6 milliseconds in the simulation at the system test facility. This difference adds to the complexity of the interfacing electronics and may further reduce the precision of the simulation.

C13: *Diagnostic information was passed from the SRI to the OBC in such a way that it could be confused with navigation data* [↓E7].

There are several possibilities to explain how this condition occurred. There is no information in the report about whether the OBC software included a check for diagnostic information and it was simply unsuccessful or whether such a check was omitted. The report provides a clue that a check of the diagnostic data was not attempted by including a recommendation that the flight software "verify the range of values taken by any internal or communication variables in the software." It is surprising that this was not routinely done, but memory or performance limitations may have figured in this decision.

A more important question is why was diagnostic information passed in such a way that it could be confused with navigation data. Once again, a basic system safety engineering design principle (as described in Chapters 15 and 16 of *Safeware* [24]) was violated.

Level 3 Systemic Factors

The Level 2 conditions listed above need to be changed to prevent a repetition of this same accident scenario but eliminating similar accidents—those having different detailed event scenarios but stemming from the same root or systemic problems—requires identifying and eliminating the systemic problems. Many of the systemic factors identified below are either omitted or not stressed in the original Ariane accident report. The recommendations in the report (as in many of the accident reports studied) focused primarily on preventing a repetition of the same events and not on systemic factors that would prevent a larger class of accidents.

Some of the differences in the factors I identified for the accidents in this report may also simply reflect my software engineering and system safety background and experience. As discussed in Chapter 3, accident reports will always reflect the background and role (management, operator, independent expert, etc.) of those writing it. This fact emphasizes the importance of including a wide range of people and expertise in the accident investigation.

The systemic factors are divided into three general categories found useful previously in analyzing a large number of accidents in a wide variety of industries (aerospace, medical, transportation, chemical, nuclear power, defense, and others) (see Chapter 4 of *Safeware* [24]) and drawn from the accident theory literature: (1) flaws in the safety culture, (2) ineffective organizational structure, and (3) ineffective or inadequate technical activities.

I. Flaws in the Safety Culture

S1 *Complacency about software: Underestimating and misunderstanding software-related risks* [↓C9, C12].

The accident report says that software was assumed to be correct until it was shown to be faulty. This assumption, closely related to a belief that testing software will lead to finding almost all the errors in it, is widespread in engineering and stems from an underestimation of the complexity of most software and an overestimation of the effectiveness of software testing. Even the software engineers in this case seemed to be overconfident, leading to unprotected variables, lack of assertions and range checks, etc. Complacency also probably played a part in the inclusion of unnecessary complexity and software functions (see S10) and many of the other ineffective or inadequate technical activities.

S2 *Consideration only of component failures as the cause of accidents and overrelying on redundancy* [↓C9, C12].

The Ariane 5 loss was a classic example of a *system accident*: Each component worked as specified but together the specified component behavior led to disastrous system behavior. Not only did each component in isolation work correctly but, in fact, many of the design features that contributed to the accident involved standard recommended practice.

Classical system engineering techniques to protect against accidents caused by component failure usually involve increasing the reliability (integrity) of the individual components, including redundancy to protect against component failure, and building margins of error into components to protect against analysis or manufacturing flaws. None of these techniques are useful in protecting against system accidents, where the problems result from system design flaws: Increasing component integrity will have no effect on accidents where the software operated exactly as it was required and designed to do (but the requirements were incorrect). Standard redundancy, while protecting against computer hardware failures, will again not protect against software requirements (or even coding) errors.

The accident report notes that according to the culture of the Ariane program, only random failures are addressed and they are primarily handled with redundancy. Thus there was a bias toward the mitigation of random failure and not system faults caused by design error. The misunderstanding of software failure modes (and how they differ from hardware failure modes) and the assumption that software failures will be like those of hardware are widespread and not just limited to the Ariane program. We need better system engineering processes and techniques. We cannot simply continue to use only those techniques that were developed in the past for a different type of system than we are building today and for different types of accidents and continue to base our engineering design decisions and tradeoffs on assumptions that no longer hold.

S3 *??Competition and budgetary pressure leading to flawed resolution of conflicting goals??* [↓C1(?), C12(?)].

Competitive and budgetary pressures are not mentioned in the report, but some level 2 conditions (for example, why the Ariane 4 software was reused) are not explained adequately and this factor could reasonably account for them. The launch market is potentially very large (estimated to be $60 billion in the next decade) and competitive. Europe is struggling to defend its greater than 50% market share from competitors such as ILS (a joint venture of Lockheed Martin, Russian, and Ukrainian firms using the Proton rocket), Boeing with its

Delta Rocket and Sea Launch project, and other possible challengers from Japan, China, and Brazil. Did this competition affect the development schedule and critical decisions such that Ariane 4 software was reused and proper analysis and testing was not performed?

II. Ineffective Organizational Structure

The report is almost totally silent about potential organizational structure issues: It does not describe the allocation of responsibility and authority for safety nor does it mention any organizational or management factors that may have influenced the accident. There is one hint that there may have been problems, however, in a final recommendation at the very end of the report.

S4 *Organizational and communication problems between the partners in the Ariane program* [↓??].

The accident report includes the following recommendation:

> A more transparent organization of the cooperation among partners in the Ariane 5 programme must be considered. Close engineering cooperation, with clear cut authority and responsibility, is needed to achieve system coherence, with simple and clear interfaces between partners.

Unfortunately, nothing else is included in the public report to provide any clues about the level 2 conditions related to organization and communication (aside from specification issues) that would better explain the accident in this regard and provide more clues to others about potentially dangerous practices.

III. Ineffective or Inadequate Technical Activities

S5 *Violation of basic safety engineering design practices in the digital parts of the system design* [↓C3, C8, C9, C13].

Although system safety engineering textbooks and standards include principles for safe design, software engineers are almost never exposed to these principles (and sometimes not even system engineers). As a result, software often fails to include these safe design practices. For example, some basic system safety engineering principles (design guidelines) [24] are to separate critical functions (in this case, attitude and guidance control) from non-critical functions such that problems in the non-critical ones cannot affect critical functions [↓C9]; to eliminate unnecessary functionality from critical components [↓C8]; and to design error-reporting messages such that they cannot be confused with critical data [↓C13].

There is no information about whether the software (and system engineers) did not know these principles (which have been developed from hard experience) or just did not include them for some reason. But such training is usually not included in computer science or software engineering classes and textbooks and often not even in engineering classes.

S6 *Flawed review process* [↓C6, C7].

The report says that the reviews included all the major partners in the Ariane 5 program so reviews *did* take place. For some reason, these reviews did not uncover the problems. Unfortunately, the report provides no detail about the review process, although this information may be in the internal ESA report on the accident nor why this process failed to uncover the multiple design flaws.

The only clue as to the deficiencies in the review process come from the recommendations that suggest including external (to the project) participants when reviewing specifications, code, and justification documents and to make sure that these reviews consider the substance of arguments, rather than check that verifications have been made. The latter seems to imply the common problem of reviewers and assurance personnel simply ticking off on checklists that activities have been performed without reviewing their quality and substance.

The quality of the review process is, of course, greatly influenced by the quality of the specifications.

S7 *Inadequate specifications* [↓C6, C7, C10, C11].

The report refers in several places to inadequate specification practices and notes that the structure of the documentation obscured the ability to review the critical design decisions and their underlying rationale. Inadequate documentation of design rationale to allow effective review of design decisions is a very common problem in system and software specifications and one that needs more research attention. Solving this problem is one of the goals of intent specifications [26]. The accident report recommends that justification documents be given the same attention as code and that techniques for keeping code and its justifications consistent be improved.

S8 *Flawed analysis with respect to software functions* [↓C5, C6].

The report says that the limitations of the SRI software were not fully analyzed in reviews, and it was not realized that the test coverage was inadequate to expose such limitations. If it was not possible to perform a complete system integration test (as was assumed by the developers), then simulation and other types of analyses become even more important as well as analysis of the assumptions underlying any simulation.

Although it is dangerous to speculate from the report, it seems clear that the possible implications of allowing the alignment software to operate during flight were not analyzed.

This is an important research topic in software engineering. Executable specifications as well as improved modeling and analysis techniques that are easier to use and review than those currently available are needed.

S9 *Software reuse without adequate analysis of its safety in a different environment* [↓C1, C4, C8, C12].

Although this factor is related to S8, it has been separated to stress its importance. Reuse (and the use of Commercial Off-The-Shelf Software–COTS) is common practice today in embedded software development. A very common misconception is that because software operated safely in other applications, it will be safe in the new one. This misconception arises from a confusion between software reliability and safety. Although component reliability is an important factor in component failure accidents, it is irrelevant for system accidents where none of the components involved fail. Almost all software-related accidents have been system accidents and involved flaws in the software requirements (the specified blackbox behavior behavior) and not implementation errors—that is, the software satisfied its requirements and operated "correctly."

Blackbox behavior of a component can only be determined to be safe by analyzing its effects on the system in which it will be operating, i.e., by consideration of the specific operational context. The fact that software was used safely in another environment provides no information about its safety in the current one. In fact, reused software is probably less safe as

the original decisions about the required software behavior were made for a different context. This fact may account for the large number of incidents and accidents that have involved reused software.

The philosophy of the Ariane 5 project, as stated in the accident report, that it was not wise to change software that had worked well on the Ariane 4 unless it was proven necessary to do so is well founded: Errors are often introduced when software is changed, particularly by those who did not originally write it. However, in this case, there is a tradeoff with safety that needs to be carefully evaluated. The best solution may not have been to leave the software as it was but to rewrite it from scratch—such a decision depends on the type of change required, the specific design of the software, and the potential effect of the feature on system safety. The cost of the decision to reuse the Ariane 4 software without rewriting it (in terms of both money and program delays) was much greater than the cost of doing a proper analysis. If the cost of the analysis for reused (or COTS) code is prohibitively expensive or beyond the state of the art, then redeveloping the software or rewriting it may be the appropriate decision.

The implication of factor S9 is not that software cannot be reused, but that a safety analysis of its operation in the current system context is mandatory. (Note that testing is not adequate to accomplish this goal). For this type of analysis to be both technically and financially practical, it is important that reused software contain only the features (specified software behaviors or requirements) necessary to perform the critical functions. Note, however, that COTS software often is built to include as many features as possible in order to make it commercially useful in a large variety of systems. If reuse and the use of COTS software are to result in acceptable risk, then new system and software modeling and analysis techniques will be needed.

S10 *Inadequate system safety engineering* [↓C5, C6, C9].

The report recommends reconsidering the definition of critical components, taking failures of software components into account (particularly single point failures). Clearly the handling of an overflow in the alignment function by turning off the guidance and control computer was flawed. And the use of a redundant backup SRI processor did not eliminate a single point failure of the guidance and control function caused by a software flaw. But not enough information is given about how the system safety analysis was performed (or even whether there was one) to provide detailed information about why it was unsuccessful.

There does appear to have been a flawed tradeoff analysis process. For example, the fact that some software variables were unprotected is not the root or systemic problem: There is always a limit as to how much checking and protection is feasible—spacecraft computers are limited by size and weight, which in turn limits processor power and memory. At the same time, the real-time and timing constraints for response to inputs may preclude extensive checking software. And extra margins (the 80% load factor) are necessary. The problem was that there was a lack of effective analysis to determine *which* variables should be protected. If the analysis determined that more protection code was needed than was possible given resource constraints, then some tradeoffs needed to be made. The report describes the decisions that resulted but not the process used to reach those decisions. Important information that is missing includes how the analysis and trade studies were performed and what additional information or additional analysis techniques could have allowed better decisions to be made.

S11 *Unnecessary complexity and software functions* [↓C2, C8].

One of the most basic concepts in engineering critical systems is to "keep it simple." The

seemingly unlimited ability of software to implement desirable features, as in this case, often pushes this basic principle into the background. *Creeping featurism* is common:

> 'And they looked upon the software, and saw that it was good. But they just had to add this one other feature' .... A project's specification rapidly becomes a wish list. Additions to the list encounter little or no resistance. We can always justify one more feature, one more mode, one more gee-whiz capability. And don't worry, it'll be easy—after all, it's just software. We can do anything.
>
> In one stroke, we are free of nature's constraints. This freedom is software's main attraction, but unbounded freedom lies at the heart of all software difficulty [30].

While the accident report does question the advisability of retaining the unused alignment function from the Ariane 4 software, it does not question whether the Ariane 4 software should have included such a non-required but convenient software function in the first place. Outside of the question of its effect on reuse (which may reasonably not have been contemplated during Ariane 4 software development), a tradeoff was made between perhaps delaying a launch and simplifying the software. The feature was used only once in Ariane 4 launches. The more features included in software and the greater the resulting complexity, the harder and more expensive it is to test, to provide assurance through reviews and analysis, to maintain, and to reuse in the future.

Engineers need to start making these hard decisions with a realistic appreciation of their effect on development cost and eventual system reliability and safety.

S12 *The test and simulation environment did not adequately reflect the operational environment* [↓C12].

A general principle in testing aerospace systems is to fly what you test and test what you fly. The test and simulation processes need to reflect the environment accurately. Although following this principle is often difficult or even impossible for spacecraft, there seems to be no excuse for not including Ariane 5 trajectory data in the specifications and simulations.

In addition, the report says that

> It would have been technically feasible to include almost the entire inertial reference system in the overall system simulations which [*sic*] were performed. For a number of reasons, it was decided to use the simulated output of the inertial reference system, not the system itself or its detailed simulation. Had the system been included, the failure could have been detected.

It is always dangerous to conclude that poor testing was the "cause" of an accident. After the fact, it is always easy to find a test case that would have uncovered a known error, but it is usually difficult to prove that the particular test case would have been selected beforehand even if testing procedures were changed. By definition, the cause of an accident can always be stated as a failure to test for the condition that was, after the fact, found to have led to the loss. However, in this case, there do seem to be omissions that reflect some poor decisions related to testing, particularly with respect to the accuracy of the simulated operational environment.

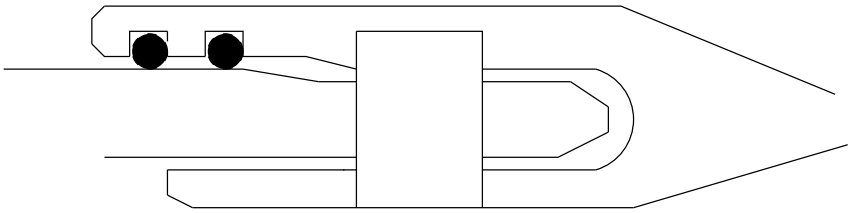

Figure 3.1: The Design of the Redundant O-Rings in the SRB

3.2 Challenger

While this accident is not related to software, the accident report is so good that it has been included to assist in comparison and evaluation of the accident models.

The report of the Presidential Commission on the Space Shuttle *Challenger* Accident [37] should be read by anyone interested in the engineering of safe systems: It provides an excellent example of a well-researched and comprehensive accident report that does not stop with a superficial treatment of the accident causes but goes beyond the easy attribution of "technical failure" or "human error." The problems the Rogers Commission found were not unique to NASA, but are common in many industries and organizations. Unless otherwise noted, the information about the accident provided is from the Rogers Commission report.

3.2.1 Background

The Space Shuttle is part of a larger Space Transportation System (STS) concept that arose in the 1960s when Apollo was in development. The concept originally included a manned Mars expedition, a space station in lunar orbit, and an Earth-orbiting station serviced by a reusable ferry, or Space Shuttle. Funding for this large an effort, on the order of that provided for Apollo, never materialized, and the concept was scaled back until the reusable Space Shuttle, earlier considered only the transport element of a broad transportation system, became the focus of NASA's efforts.

The Shuttle concept itself was scaled back, because of cost considerations, to a three-part system consisting of the orbiter, an expendable external fuel tank containing liquid propellants for the orbiter's engines, and two recoverable solid rocket boosters (SRBs). Contracts were awarded in 1972: Rockwell was responsible for the design and development of the orbiter, Martin Marietta was assigned the development and fabrication of the external tank, Morton Thiokol was awarded the contract for the SRBs, and Rocketdyne (a division of Rockwell) was selected to develop the orbiter main engines.

Three NASA field centers were given responsibility for the program. Johnson Space Center in Houston, Texas, was to manage the orbiter, while Marshall Space Flight Center in Huntsville, Alabama, was assigned responsibility for the orbiter's main engines, the external tank, and the SRBs. Kennedy Space Center in Merritt Island, Florida, would assemble the Shuttle components, check them out, and conduct launches.

Because of continual budget cuts and constraints, the original fleet of five Shuttle orbiters was reduced to four. The orbital test flight program began in early 1981, and, after the landing of STS4 in July 1982, NASA declared that the Shuttle was "operational." Including the initial orbital tests, the Shuttle flew 24 successful missions over a 57-month period before the *Challenger* accident.

The design of the Shuttle solid rocket booster (SRB) was based on the U.S. Air Force Titan III, one of the most reliable ever produced. Significant design changes were made, however, including changes in the placement of the O-rings. The solid rocket motor is assembled from four large

cylindrical segments—each over 25 feet long, 12 feet in diameter, and containing more than 100 tons of fuel; the resulting four joints are called field joints. These joints are sealed by two rubber O-rings that are installed when the motor is assembled (see Figure 3.1). The primary O-ring and its backup, the secondary O-ring, were designed to seal a tiny gap in the joints created by pressure at ignition. The O-ring seal must be pressure actuated very early in the solid rocket motor ignition. If this actuation is delayed, the rocket's combustion gases may blow by the O-ring and damage or destroy the seals. Redundancy was used to avoid this possibility: If the primary O-ring did not seal, then the secondary one was supposed to pressurize and seal the joint. Part of the Shuttle joint was longer than in the Titan in order to accommodate two O-rings instead of one. At the same time, the longer length made the joint more susceptible to bending under combustion pressure than the Titan joint.

Seventy-three seconds after the flight began, it ended with the hydrogen and oxygen propellants igniting in a huge ball of flame, destroying the external tank and exposing the orbiter to severe aerodynamic loads that caused complete structural breakup. All seven crew members died. The two SRBs flew out of the fireball and were destroyed by the Air Force range safety officer 110 seconds after launch. It was the worst accident in the history of manned flight and the first time any American astronauts were lost during a mission.

3.2.2 Chain of Events

E1 The flight began at 11:39 am on January 28, 1986. At 0.678s after liftoff, photographic data show a strong puff of gray smoke was spurting from the vicinity of the after field joint on the right Solid Rocket Booster. The vaporized material streaming from the joint indicated there was not complete sealing action within the joint. The black color and dense composition of the smoke puffs suggest that the grease, joint insulation, and rubber O-ring in the joint seal were being burned by the hot propellant gases. As Challenger cleared the tower, gas was already blowing by the rings, although the gap was temporarily plugged by burning rubber and putty. The last smoke was seen at 2.733s.

E2 At 37s, Challenger encountered the first of several high-altitude wind shear conditions. The wind shear created forces on the vehicle with relatively large fluctuations. The wind shear forces were immediately sensed and countered by the guidance, navigation, and control system. The steering system (thrust vector control) of the Solid Rocket Booster responded to all commands and wind shear effects.

E3 The first flickering flame appeared on the right SRB in the area of the aft field joint at 58.788s into the flight.

E4 As the flame plume increased in size, it was deflected rearward where it began to burn through the main fuel tank as well as one of the struts that held the rocket to the tank.

E5 Less than 14 seconds later, the strut gave way and the pointed nose of the SRB swiveled inward to pierce the fuel tank.

E6 The structural failure of the hydrogen tank released massive amounts of liquid hydrogen from the tank. The burning hydrogen enveloped and destroyed Challenger.

E7 The Orbiter, under severe aerodynamic loads, broke into several large sections which emerged from the fireball.

E8 The two SRBs were destroyed by the Air Force range safety officer.

3.2.3 Identified Cause of the Accident

The Rogers Commission concluded that the loss of STS Mission 51-L was caused by a failure in the joint between the two lower segments of the right solid rocket motor. The failure was the result of the destruction of the seals that are intended to prevent hot gases from leaking through the joint during the propellant burn of the rocket motor. A resultant gas leak through the right Solid Rocket Motor aft field joint initiated at or shortly after ignition eventually weakened and/or penetrated the External Tank, initiating vehicle structural breakup and loss.

The O-ring failure was due to a faulty design unacceptably sensitive to a number of factors. These factors were the effects of temperature, physical dimensions, the character of the materials, the effects of reusability, processing, and the reaction of the joint to dynamic loading. The commission noted in particular the potential role of a changed leak check:

> A likely cause of the O-ring erosion appears to have been the increased leak check pressure that caused hazardous blow holes in the putty. Such holes at booster ignition provide a ready path for combustion gases directly to the O-ring. The blow holes were known to be created by the higher pressure used in the leak check [37, p.156].

The Rogers Commission report, however, did not stop with the physical factors. It found that the Space Shuttle's SRB problem began with a faulty design of its joint and increased as both NASA and the Thiokol management first failed to recognize the problem, then failed to fix it when it was recognized, and finally treated it as an acceptable risk.

3.2.4 Hierarchical Model

Expanded chain of events

The Challenger and Titan/Milstar accident reports were the only two of those studied in this exercise that contained enough information to create an expanded chain of events to adequately explain the accident and its causal factors.

E0-1 Leon Ray (an engineer at Marshall) first concluded after tests in 1977 and 1978 that rotation of the SRB field joint under pressure caused the loss of the secondary O-ring as a backup seal. Nevertheless, in November 1980, the SRB joint was classified on the NASA Shuttle Critical Items List (CIL) as Criticality 1R.[2] The use of R, representing redundancy, signifies that NASA believed that the secondary O-ring would pressurize and seal if the primary O-ring failed to do so. The 1980 CIL does express doubt about the secondary O-ring successfully sealing if the primary should fail under certain conditions—when the motor case pressure reaches or exceeds 40 percent of maximum expected operating pressure—but the joint was assigned a 1R classification in November 1980.

E0-2 O-ring anomalies were found during the initial flights, but the O-ring erosion problem was not reported in the Marshall problem assessment system and given a tracking number, as were other flight anomalies. After tests in May 1982, Marshall management finally accepted the conclusion that the secondary O-ring was no longer functional after the joints rotated under 40 percent of the SRB maximum operating pressure, and the criticality was changed to Criticality 1 on December 17, 1982.

[2] Criticality 1 means that failure could cause the loss of life or vehicle. Criticality 1R means that the component contains redundant hardware. Over 700 pieces of the Shuttle hardware were listed as Criticality 1[29].

E0-3 Testimony at the Rogers Commission hearings supported the conclusion that, despite the change in criticality from 1R to 1, NASA and Thiokol management still considered the joint to be a redundant seal in all but a few exceptional cases. The disagreement went to a "referee" for testing, which was not concluded until after the *Challenger* accident. Since the design was now rated Criticality 1, a waiver was required to allow it to be used. However, the problem assessment system operated by Rockwell contractors at Marshall still listed the field joint as 1R on March 7, 1986, more than five weeks after the accident.

E0-4 Before SRS flight 41-B in February 1984, the O-ring erosion and blow-by problem occurred infrequently. STS 41-B had extensive erosion, but Thiokol filed a problem report concluding that the secondary O-ring was an adequate backup according to the tests they had made. Some engineers at Marshall and Thiokol disagreed with the Thiokol numbers, but approval to fly 41-C was given. After more erosion and blow-by was found on 41-C, NASA asked Thiokol for a formal review of the SRB joint-sealing procedures. Thiokol was asked to identify the cause of the erosion, determine whether it was acceptable, define any necessary changes, and reevaluate the putty then being used.[3]

E0-5 A change was made after STS-9 in the procedures used to check for leaks in the joint seal, which created more putty blow holes and increased the frequency of erosion. A letter in 1984 from Thiokol suggested that the chance of O-ring erosion had increased because of this change. During the first nine flights (before flight 41-B in January 1984), when the leak check pressure was lower, only one field joint anomaly had been found. When the leak-test pressure was increased for STS 41-B and later flights, over half the Shuttle missions experienced field joint O-ring blow-by erosion of some kind. The same experience was found with the nozzle O-ring.

E0-6 Thiokol and NASA witnesses at the Rogers Commission hearings agreed that they were aware that the increase in leak-test blow holes in the putty could contribute to O-ring erosion. Nevertheless, Thiokol recommended and NASA accepted the increased pressure to ensure that the joint passed the integrity tests. Thiokol established plans for putty tests to determine how the putty was affected by the leak check after the 41-C experience, but their progress in completing the tests was slow.

E0-7 On January 24 1985, STS 51-C was launched at the coldest temperature to that date—53 degrees. O-ring erosion occurred in both solid boosters, and blow-by was more significant than had been experienced before. The problem assessment report (which was started to track field joint erosion after flight 41-B) described the O-ring anomaly after 51-C as "as bad or worse than previously experienced...Design changes are pending test results" [37, p.136]. The design changes being considered included modifying the O-rings and adding grease around them to fill the void left by the putty blow holes created by the leak tests.

E0-8 Thiokol presented an analysis of the problem on February 8, 1985 that noted a concern that the seal could be lost, but it concluded that the risk of primary O-ring damage should be accepted. Risk acceptance was justified, in part, on the assumption that the secondary O-ring would work even with erosion. During the flight readiness assessment at Marshall for 51-D, Thiokol for the first time mentioned temperature as a factor in O-ring erosion and blow-by. Thiokol concluded that "low temperature enhanced probability of blow-by—51-C experienced worst case temperature change in Florida history" [37, p.136].

[3]Asbestos-filled putty was used to pack the space in the joint and prevent O-ring damage from the heat of combustion gases. Problems with the putty can lead to the burning of both O-rings and thus to an explosion.

E0-9 The joint seal problem occurred in each of the next four Shuttle flights. Thiokol conducted O-ring resiliency tests in response to the extensive O-ring problems found on flight 51-C in January and found that the key variable was temperature.

E0-10 STS 51-B was launched on April 29, 1985. Inspection of the O-rings after the flight revealed a more serious problem than they had previously experienced: The primary nozzle O-ring never sealed at all, and the secondary O-ring was eroded. The erosion was greater than that predicted as the maximum possible by the model they were using. As a result, the Marshall Problem Assessment Committee placed a launch constraint[4] on the Shuttle system that applied to flight 51-F and all subsequent flights (including the fatal 51-L *Challenger* flight). Thiokol officials testified at the Rogers Commission hearings that they were unaware of this launch constraint, but Thiokol letters referenced the report containing the constraint.

E0-11 Although launch constraints are required to be reported to NASA Level II management, this was not done. After the launch constraint was imposed, it was waived for each shuttle flight thereafter.

E0-12 After the 51-B erosion problems became known, Roger Boisjoly (an engineer at Thiokol) wrote a memo on July 31, 1985, recommending that a team be set up to solve the O-ring problem. The memo concluded that a catastrophe could occur if immediate action was not taken. On August 19, Thiokol and Marshall program managers briefed NASA Headquarters on the seal erosion problem and concluded that the O-ring seal was a critical matter but that it was safe to fly. An "accelerated pace" was recommended for the effort to eliminate seal erosion. Thiokol's Vice President for Engineering, noting that "the result of a leak at any of the joints would be catastrophic," announced on August 20 the establishment of a Thiokol task force to recommend short-term and long-term solutions.

E0-13 Early in October, Thiokol management received two separate memos from the O-ring task force describing administrative delays and lack of cooperation. One was written by Roger Boisjoly and warned about lack of management support for the O-ring team's efforts. The other memo started with the word "HELP!" and complained about the seal task force being "constantly delayed by every possible means"; the memo ended with the words "This is a red flag."

E0-14 Shuttle flight 61-A was launched on October 30, 1985, and experienced both nozzle O-ring erosion and field joint O-ring blow-by. These anomalies were not mentioned at the Flight Readiness Review for the next flight. Flight 61-B was launched on November 26, 1985; it also sustained nozzle O-ring erosion and blow-by.

E0-15 In December, R. V. Ebeling (manager of Thiokol's solid rocket motor ignition system) became so concerned about the seriousness of the O-ring problem that he told the other members of the seal taskforce that he believed Thiokol should not ship any more motors until the problem was fixed.

E0-16 On December 10, Thiokol wrote a memo to NASA suggesting that the O-ring problem be closed with respect to the problem-tracking process. Thiokol officials testified at the Rogers Commission hearings that this was in response to a request from the Director of Engineering

[4] A launch constraint means that the problem has to be addressed during the flight readiness cycle to ensure that NASA is staying within the "test experience base"—that is, the suspect element is operated only within the parameters for which it has worked successfully before.

at Marshall. Having the problem listed as closed meant that it would not be involved in the flight readiness reviews, although it could still be worked on. The letter was still in the review cycle at the time of flight 51-L (*Challenger*), but entries were placed on all Marshall problem reports (the closed-loop tracking system) indicating that the problem was considered closed. The Commission heard testimony that these entries were "in error."

E0-17 Flight 51-L was originally scheduled for July 1985, but by January 1985 the flight had been postponed to late November to accommodate changes in payloads. The launch was later delayed further and finally rescheduled for January 22, 1986. The launch was further delayed three times and scrubbed once. The first of these last three postponements was announced on December 23, 1985, in order to accommodate a slip in the launch date of mission 61-L. On January 22, the launch was changed from January 23 to January 25 because of problems caused by the late launch of flight 61-C.

E0-18 The third postponement occurred on January 25: Because of an unacceptable weather forecast for the Kennedy area, the launch was rescheduled for January 27. But the January 27 launch countdown had to be halted when the crew entry and exit hatch handle jammed. By the time the hatch handle was fixed, the crosswinds at Kennedy exceeded the maximum allowable for a return-to-launch abort, and the launch was canceled at 12:26 p.m. and rescheduled for January 28.

E0-19 At 2 p.m. on February 27, the mission management team met again. At that time, the weather was expected to clear, but temperatures were to be in the low twenties overnight and a temperature of 26 degrees was predicted at the intended time of launch (9:38 a.m.).

E0-20 Robert Ebeling, at the Thiokol plant, heard about the predicted low temperatures and called a meeting at 2:30 p.m. with Roger Boisjoly and the other Thiokol engineers. The engineers expressed great concern about launching in such low temperatures, since this was far below previous experience and below the temperatures for which the SRBs had been qualified.

E0-21 A teleconference was set up for 5:45 p.m. with the Thiokol engineering people in Utah, the Thiokol people at Marshall, and the Thiokol representative at Kennedy. NASA participants were all from Marshall. Concerns about the effect of low temperature on the O-rings and joint seal were presented by Thiokol, along with an opinion that the launch should be delayed. A recommendation was also made that Arnold Aldrich, the Program Manager at Johnson (Level II), be informed about the concerns, but this was never done.

E0-22 The Marshall and Thiokol personnel decided to schedule a second teleconference later in the day, after data could be faxed to the various locations. The Thiokol charts and written data were faxed to Kennedy, and the second phase of the teleconference started at 8:45 p.m. with additional personnel involved. The history of the O-ring erosion and blow-by in the SRB joints of previous flights was presented along with the test results at Thiokol. Bob Lund, Thiokol Vice President of Engineering, presented the conclusion of the Thiokol engineers that the O-ring temperature must be at or greater than 53 degrees at launch, which was the previous lowest launch temperature. Boisjoly testified at the Commission hearings that no pro-launch statement was made by anyone in the room.

E0-23 During this conference, the Marshall Deputy Director of Science and Engineering is reported to have said that he was "appalled" by Thiokol's recommendation. The manager of the SRB project at Marshall suggested that Thiokol was using different launch criteria than in the

previous 24 flights and that under those criteria, they would not be able to launch until April. After protesting that Thiokol was changing the launch criteria on the night before a scheduled mission, he reportedly exclaimed "You can't do that." The NASA personnel challenged the Thiokol conclusions and recommendations, stressing that the secondary O-ring would seal if the primary one did not.

E0-24 The teleconference was recessed at about 10:30 p.m. for an off-the-air caucus of Thiokol personnel. This recess lasted for about 30 minutes. During this time, the Thiokol manager of the Space Booster Project at Kennedy continued to argue against the launch. He testified that he said, "If we are wrong and something goes wrong on this flight, I wouldn't want to have to be the person to stand up in front of a board of inquiry and say that I went ahead and told them to go ahead and fly this thing outside what the motor was qualified to" [37, p.95].

E0-25 At Thiokol in Utah, about ten engineers participated in the discussion, and very strong objections to the launch were voiced. After the discussion between Thiokol management and engineers was completed, a final management review began. The Thiokol Senior Vice President said that a management decision had to be made, turned to the Thiokol Vice President of Science and Engineering, and asked him to take off his engineering hat and to put on his management hat.

E-26 The teleconference was reconvened at 11:00 p.m., at which time Thiokol management stated that they had reassessed the problem and were withdrawing their opposition to the launch—the temperature was a concern, but the data was inconclusive. They concluded that (1) there was a substantial margin to erode the primary O-ring by a factor of three times the previous worst case and (2) the "harder" O-rings would take longer to seat, but that if the primary seal did not seat, the secondary seal would. They therefore recommended that launch proceed. NASA Level I and II management and the Launch Director for 51-L were never told about the initial Thiokol concerns and opposition to the launch—all the discussions and conferences included only Marshall and Thiokol personnel.

E0-27 It rained heavily after *Challenger* was rolled out to the launch pad.

E0-28 Ice accumulated on the launch pad during the night. Rockwell was consulted about launch conditions, and they advised that the ice was an unknown condition. At the weather briefing for the Shuttle crew, the temperature and ice on the pad were discussed, but the crew was not informed of any concern about the effects of low temperature. After an ice inspection, the launch was delayed to allow more time for the ice to melt. After consultation with senior advisors and without having been told about the O-ring discussions at Marshall or about the doubts of the Thiokol engineers concerning flight safety at those low temperatures, Shuttle Director Jesse Moore gave the permission to launch. The ambient temperature at launch was 36 degrees measured at ground level, 15 degrees colder than that of any previous launch.

E0-29 The flight began at 11:39 a.m. and ended 110 seconds after launch (see events E1–E8).

Level 2 Conditions

C1 The longer length of the Shuttle solid rocket motor joint made it more susceptible to bending under combustion pressure than the original Titan joint. (↓E1, E0-2)

C2 Misrepresentation of criticality may have led some managers to believe—wrongly—that redundancy existed. Most of the problem reporting paperwork tracking the O-ring erosion problem that was generated by Thiokol and Rockwell still listed the field joint as Criticality 1R long after the status had been changed to Criticality 1. The Rogers Commission concluded that, as a result, informed decision making by key managers was impossible. (↓E0-3, E0-26)

C3 The increased air pressure forced through the joint in the new O-ring leak check created more putty blow holes, allowing more focused jets on the primary O-ring, and thereby increasing the frequency of erosion. Related changes were made in the type of putty used and the patterns for positioning it. (↓E1, E0-4, E0-5, E0-6)

C4 The government contractor that managed the SRB assembly had been changed and on-site O-ring inspections at Kennedy were discontinued. (↓E0-4)

C5 The assumption was widely held that the secondary O-ring would work even with erosion and that the secondary O-ring would seal if the primary one did not. (↓E0-1, E0-4, E0-8, E0-26)

C6 The launch constraint was never reported to NASA Level II management. The Rogers Commission was told that two entries on the O-ring erosion nozzle problem report had erroneously stated that the O-ring erosion problem had been resolved or closed. (↓E0-11)

C7 The Marshall problem reporting system showed the problem as considered closed. (↓E0-16)

C8 Lots of delays led to increased pressure to get the launch off. (↓E0-17, E0-18)

C9 Neither Thiokol nor NASA expected the rubber O-rings sealing the joints to be touched by hot gases of motor ignition, much less to be partially burned. However, as tests and then flights confirmed damage to the sealing rings, the reaction by both NASA and Thiokol was to increase the amount of damage considered "acceptable." Although Thiokol engineers were recommending that action be taken, at no time did management either recommend a redesign of the joint or call for the Shuttle's grounding until the problem was resolved. (↓E0-13, E0-19, E0-21, E0-26, E0-28)

C10 The anomaly tracking system for flight readiness reviews contained incorrect information. Procedures for anomaly tracking with respect to the O-ring problems were not followed. (↓E0-2, E0-11, E0-14, E0-16)

C11 Morton Thiokol did not accept the implication of tests early in the program that the design had a serious and unanticipated flaw. NASA did not accept the judgement of its engineers that the design was unacceptable, and as the joint problems grew in number and severity, NASA minimized them in management briefings and reports. (↓E0-13, E0-16)

C12 Neither NASA nor Thiokol fully understood the mechanism by which the joint sealing action took place. (↓E0-10, E0-26, E0-28)

C13 The Commission noted several significant trends in the flight readiness reviews. First, O-ring erosion was not considered early in the program when it first occurred. Then, when the problem grew worse after STS 41-B, there was an early acceptance of the phenomenon without much analysis or research. Later flight readiness reviews gave the problem only a cursory review and often dismissed it as within "acceptable" or "allowable" limits because it was "within the database" of prior experience. (↓E0-10)

C14 Both Thiokol and Marshall continued to rely on the redundancy of the secondary O-ring long after NASA had officially declared that the seal was a nonredundant single point of failure. In 1985, when temperature became a major concern after STS 51-C and the launch constraint was imposed, higher management levels (I and II) were never informed. (↓E0-23)

C15 The ambient temperature at time of launch was 36 degrees Fahrenheit, or 15 degrees lower than the next coldest previous launch. O-ring resiliency is directly related to its temperature. A warm O-ring that has been compressed will return to its original rounded shape much quicker than will a cold O-ring when compression is relieved. If the O-ring remains in its compressed position too long, then the gap between the O-ring and the upstream channel wall may not be sealed in time to prevent joint failure. It is probable that the O-rings in the right solid booster of the aft field joint were not following the opening of the gap between the tang and clevis at the time of ignition and thus did not seal the gap in time to preclude joint failure due to blow-by and erosion from hot combustion gases. (↓E1, E0-19, E0-20)

C16 There was heavy rain before the launch, approximately 7 inches compared with the 2.5 inches that would have been normal for that season, and water may have gotten into the joints. At the time of launch, it was cold enough that water present in the joint would freeze. Tests showed that ice in the joint could inhibit proper secondary seal performance. (↓E0-27, E0-28)

C17 The Commission report noted that a careful analysis of the flight history of O-ring performance would have revealed the correlation of O-ring damage and low temperature, but neither NASA nor Thiokol carried out such an analysis. Instead, they considered only the flights in which thermal distress had been observed and not the frequency of occurrence in all flights. When all flights are included, only three incidents of O-ring thermal distress occurred out of 20 flights with O-ring temperatures at 66°F or above, while all four flights at 63°F or below experienced it. "Consideration of the entire launch temperature history indicates that the probability of O-ring distress is increased to almost a certainty if the temperature of the joint is less than 65" [37]. (↓E0-7, E0-9, E0-10, E0-22, E0-26)

C18 Some Thiokol engineers testified that usually in reviews, the engineers were required to prove that they were ready to launch. In this discussion, however, they felt that the roles had been reversed and that the engineers were required to prove that the launch would *not* be successful. (↓E0-21, E0-22, E0-23, E0-24, E0-25)

C19 The Rogers Commission concluded that the Thiokol management reversed its position and recommended the launch of 51-L, at the urging of Marshall and contrary to the views of its engineers, in order to accommodate a major customer. (↓E0-22, E0-25, E0-26)

C20 Rockwell's recommendation on launch (with respect to ice on the pad) was ambiguous. However, they did tell NASA that the ice was an unknown condition. NASA's response to Rockwell's concern on ice was not appropriate, according to the accident report. (↓E0-28)

C21 No safety representative, reliability, or quality assurance engineer was invited to the January 27 teleconference between Marshall and Thiokol. (↓E0-21, E0-22, E0-23, E0-24, E0-26)

C22 There was no representative of safety on the Mission Management Team that made key decisions during the countdown on Jan. 28 (↓E0-28).

C23 The decision makers, and in particular the Shuttle Program Manager at Johnson, did not know all the facts. They were unaware of the recent history of problems concerning the

O-rings and the joint and were also unaware of the initial written recommendation of the contractor advising against the launch at temperatures below 53 degrees Fahrenheit and the continuing opposition of the engineers at Thiokol after the management reversed its position. They did not have a clear understanding of Rockwell's concern that it was not safe to launch because of ice on the pad. The Rogers Commission concluded that if the decision makers had known all of the factors and if they had been clearly stated and emphasized in the flight readiness process, it is highly unlikely they would have decided to launch 51-L on January 28, 1986. (↓E0-12, E0-21, E0-26)

C24 Solid Rocket Motor segments were assembled using approved procedures, but significant out-of-round conditions existed between the two segments joined at the right Solid Rocket Motor aft field joint. (↓E1)

C25 Wind shear caused the steering system to be more active than on any previous flight. The winds aloft caused control actions in the time interval of 32 seconds to 62 seconds into the flight that were typical of the largest values experienced on previous missions. (↓E2)

C26 It is possible that thrust vectoring and vehicle response to wind shear as well as planned maneuvers reinitiated or magnified the leakage from a degraded seal in the period preceding the observed flames. (↓E3)

Level 3 Systemic Factors

Roger's Commission did an excellent job of identifying the systemic factors; I have simply repeated them here.

I. Flaws in the Safety Culture

S1 *Overconfidence and complacency*

The Rogers Commission report notes that Shuttle flights had become routine. In addition, safety criteria were relaxed over time, and risks were tolerated because they had been experienced before—no adequate attempt was made to eliminate them. NASA and its subcontractors accepted escalating risk (although they probably did not realize it was escalating) because they had gotten away with it previously.

Feynman, in an appendix to the report, also noted that management had a very different perception of the risk than the engineers and judged it to be much lower than those who were more informed about the technical difficulties.

S2 *Overrelying on redundancy*

The engineers and managers relied on redundancy without properly evaluating whether the redundancy provided adequate protection. They continued to believe in the independence of failures of the redundant O-rings long after it was shown to be untrue. While some of this mistaken reliance has a basis in inadequate communication and documentation procedures, mistaken reliance was placed on redundancy even by those who knew the criticality had been changed from 1R to 1.

S3 *Assuming risk decreases over time*

NASA had a perception that less safety, reliability, and quality assurance activity would be required during routine Shuttle operations and that an operational program no longer

needed system safety attention. This perception may partly reflect the lack of attention in engineering education and practice on the operations phase. Once the Shuttle was declared to be "operational," several safety organizations were reorganized and reduced in size. The Chief Engineer at NASA headquarters, who had overall responsibility for safety, reliability, and quality assurance, had a staff of 20 people, only two of whom spent 10 percent and 25 percent of their time, respectively, on these issues. Moreover, some safety panels, which were providing safety review, went out of existence or were merged.

S4 *Ignoring warning signs*

Warning signs existed, but they were ignored. Even the engineers who were writing memos and speaking out at meetings to try to draw attention to the warning signs were unable to draw attention to the problems. As the report notes:

> Morton Thiokol did not accept the implication of tests early in the program that the design had a serious and unanticipated flaw. NASA did not accept the judgment of its engineers that the design was unacceptable, and as the joint [and O-ring] problems grew in number and severity, NASA minimized them in management briefings and reports. Thiokol's stated position was that 'the condition is not desirable, but is acceptable.'

S5 *Low priority assigned to safety*

While management may express their concern for safety, true priorities are shown during resource allocation. By the time of the fatal Challenger flight, an extensive Apollo safety program, established by Jerome Lederer in 1967, had become "silent" and undervalued. When resources were cut, reductions were made in the safety, reliability, and quality assurance functions that essentially made those functions ineffective. The work force was decreasing at the same time the flight rate increased (although this was also related to the lack of belief that safety was important in an operational program). The Rogers Commission report describes what it called NASA's "Silent Safety Program" and concludes that a properly staffed, supported, and robust safety organization might well have avoided the communication and organizational failures that influenced the launch decision.

S6 *Flawed resolution of conflicting goals*

Every program is subject to internal and external pressures and the need to make tradeoffs between conflicting goals. An important factor in this accident was the pressures exerted on NASA by an unrealistic flight schedule with inadequate resources and by commitments to customers. The nation's reliance on the Shuttle as its principal space launch capability created a relentless pressure on NASA to increase the flight rate. The Rogers Commission notes that the attempt to build up to 24 missions a year brought problems such as compression of training schedules, a lack of spare parts, the focusing of resources on near-term projects, and a dilution of the human and material resources that could be applied to any particular flight. In addition, customer commitments may occasionally have obscured engineering concerns. These pressures undoubtedly influenced the decisions made by the Marshall personnel and their discounting of the views of most of the Thiokol engineers and some of the Marshall engineers about the joint seal.

Other evidence of safety not getting high enough priority in the tradeoff process was the whittling down of safety margins (for example, the protection plan for the launch pad was inadequate) and the waiving of launch constraints at the expense of flight safety.

II. Ineffective Organizational Structure

S7 *Diffusion of responsibility and authority*

The organizational structures at Kennedy placed the safety, reliability, and quality assurance offices under the supervision of the very organizations and activities they were to check. In addition, they were dependent on the NASA management structure to get information and recommendations about safety problems and for implementing suggested changes. But these structures and communication lines were flawed. In addition, project managers for various elements of the Shuttle program felt more accountable to their center management than to the Shuttle program.

At the same time, the organizational responsibility for system safety was not adequately integrated and available to decision-making levels. In the absence of a structured process to integrate safety-related analyses and conformance to specifications, information about safety issues was several interfaces removed from the people involved in the decisions on schedules and launch. For example, there was no system that made it imperative that launch constraints and waivers of launch constraints be considered by all levels of management.

Safety was originally identified as a separate responsibility by the Air Force during the ballistic missile programs to solve exactly the problems seen here—to make sure that safety is given due consideration in decision making involving conflicting pressures and that safety issues are visible at all levels of decision making. In such programs, system safety has a direct link to the program manager, as well as to most other parts of the program, to make sure that decisions are made with the best information possible. Having an effective safety program cannot prevent errors in judgment in balancing conflicting requirements of safety, schedule, and cost, but it can at least make sure that decisions are informed and that safety is given due consideration.

S8 *Lack of independence and low-level status of the safety personnel*

Again, the safety offices at Kennedy and Marshall were placed under the supervision of the organizations and activities whose efforts they were to check. But there are clues of much more serious problems here. In the testimony to the commission, NASA's safety staff, curiously, was never mentioned. (This is also true for the more recent accident reports discussed in this chapter). They were not involved in critical discussions about safety issues that it would seem they should have been in the middle of, if not leading.

The Rogers Commission report concludes that the lack of an independent role for the safety engineers and their effectively low-level status undoubtedly contributed to the accident.

One seemingly important aspect of this problem that is *not* mentioned in the report is the NASA decision to put safety under the assurance umbrella instead of treating it as a central engineering concern. Safety has to be built into a system and it is not simply something that is assured or assessed after the fact. This common mistake is discussed further in Chapter 4.

S9 *Limited communication channels and poor information flow*

Obviously there was a problem in getting proper input to decision making. Marshall management had a propensity to contain serious problems and to attempt to resolve them internally rather than communicate them forward. The accident report noted miscommunication of technical uncertainties and failure to use information from past near misses (incidents). Relevant concerns were not being reported to management. Problem reporting requirements

were at best unclear and were reduced in 1983 to the point where they failed to get critical information (such as anomalous events, the status of safety-critical items, criticality levels of components, and launch constraints) to the proper levels of management and engineering: Management lost insight into flight safety problems.

Memoranda and analyses expressing concerns about performance and safety were subject to many delays in transmission up the organizational chain, as well as subject to numerous stages of editing and potential vetoes on further transmittals. Vital program information and relevant concerns frequently never got to the Program Manager, including, in this case, the Thiokol launch concerns and opposition to the launch.

III. Ineffective Technical Activities

S10 *Workforce reductions, loss of skills and experienced personnel.*

The Rogers Commission report notes that at the same time the flight rate was increasing, a variety of factors reduced the number of skilled personnel available to deal with it: retirements, hiring freezes, transfers to other programs like the Space Station, and changing to a single contractor for operations support (a significant number of employees elected not to change, which resulted in the need to hire and qualify new personnel). Many of the deficiences in the technical activities, including the safety program, can be traced to the lack of skilled personnel.

S11 *Inadequate system safety program*

Prior to 1983, NASA had a highly regarded hazard analysis and follow-up program [31]. The reductions in safety personnel naturally reduced their ability to perform required tasks. Hazards and safety-related problem reports were not investigated thoroughly and little or no trend analysis was performed on O-ring erosion and blow-by problems. There was also an inadequate response to facts obtained during testing about hazards. The hazard reporting, tracking, and evaluation system seems to have broken down entirely.

S12 *Reuse without adequate analysis*

Parts of the original Titan design were reused without adequate analysis of the different stresses to which they would be subjected. For example, the longer length of the Shuttle solid rocket motor joint made it more susceptible to bending under combustion pressure.

S13 *Failure to evaluate changes*

Accidents in general commonly occur after some change in the system. This fact is well known, and any changes should automatically trigger a safety assessment or at least increased attention. The Rogers Commission notes that several changes at that time might have been the source of the O-ring problems as evidenced by the increased blow-by and erosion. The critical changes might have been detected in time to avoid the accident if adequate attention had been paid to them. The changes to SRB procedures at Kennedy included an increase in the leak check pressure on the field joint, which sometimes blew holes through the protective putty; a change in the type of putty, changes in the patterns for positioning the putty, increased reuse of motor segment casings, discontinuation of on-site O-ring inspections, and a change in the government contractor who managed the SRB assembly.

S14 *Deficiencies in information collection and use*

One of the most important factors in any system safety program is the quality of the hazard information system. Both collection of critical information as well as dissemination to the appropriate people for action is required. What had once been regarded as an outstanding hazard information system had broken down amid the cost-cutting and complacency. For example, the anomaly-tracking system (used in flight readiness reviews) contained only anomalies "outside the database"; major problems could be removed from and then lost by the reporting system. Basically, if the hazard did not cause any obvious (or noticable) problems for one or two flights, the record of the hazard disappeared. Launch constraints could be waived without any record of a waiver or even explicit documentation of the constraint.

S15 *Inadequate joint test and certification program*

It is always easy to implicate poor testing once an accident has occurred and the conditions that caused it are illuminated. The Rogers Commission points out the deficiencies in the joint test and certification program but then correctly goes on to identify all the other factors that came into play in the events surrounding the accident.

3.3 Mars Climate Orbiter

3.3.1 Background

In 1993, NASA started the Mars Surveyor program with the objective of conducting an on-going series of missions to explore Mars. JPL was identified as the lead center for this program. Mars Climate Orbiter (MCO) and Mars Polar Lander (MPL) were the second installment in NASA's long-term program of robotic exploration of Mars, which was initiated with the 1996 launches of the currently orbiting Mars Global Surveyor and the Mars Pathfinder lander and rover.

Lockheed Martin Astronautics (LMA) was selected as the prime contractor to design and develop both the MCO and MPL spacecraft, lead flight system integration and test, and support launch operations. JPL retained responsibilities for overall project management, spacecraft and instrument development management, project system engineering, mission design, navigation design, mission operation system development, ground data system development, and mission assurance.

The Mars Surveyor Operations Project (MSOP) assumed responsibility for conducting flight operations after launch. MSOP is implemented as a geographically distributed partnership between JPL, LMA, and the science teams, with each partner performing distinct functions. LMA is responsible for all spacecraft operations functions including health and status monitoring and spacecraft sequence development. LMA also performs real-time command and monitoring operations from their facility in Denver, Colorado. JPL is responsible for overall project and mission management, system engineering, quality assurance, ground data system maintenance, navigation, mission planning, and sequence integration. Each of the science teams is responsible for planning and sequencing their instrument observations, processing and archiving the resulting data, and performing off-line data analysis. These operations are typically performed at the Principal Investigator's home institution.

MSOP was responsible for conducting flight operations for both MCO and MPL (Mars Polar Lander), as well as Mars Global Surveyor (MGS). MGS was the first flight mission in the Mars Surveyor program and was being supported by MSOP at the same time as MCO.

The Mars Climate Orbiter was launched December 11, 1998 atop a Delta II launch vehicle from Cape Canaveral Air Force Station, Florida. Nine and a half months after launch, in September 1999,

the spacecraft was to fire its main engine to achieve an elliptical orbit around Mars. It then was to skim through the Mars upper atmosphere for several weeks, in a technique called aerobraking, to move into a low circular orbit. Friction against the spacecraft's single, 5.5 meter solar array was to have lowered the altitude of the spacecraft as it dipped into the atmosphere, reducing its orbital period from more than 14 hours to 2 hours. On Sept. 23, 1999 the Mars Climate Orbiter was lost when it entered the Martian atmosphere on a lower than expected trajectory.

The report leaves out a lot of information that is necessary to understand and create complete models of the accidents. Two possible reasons for the deficiency may be the necessity to get the report out quickly so the results could impact the Mars Polar Lander project and the fact that most of the missing information is related to procedures within a subcontractor (LMA) and not NASA itself.

3.3.2 Chain of Events

The following chain of events appeared in the accident report:

E1: MCO was launched on December 11, 1998 and began a trajectory toward Mars.

E2: Throughout spring and summer of 1999, concerns existed at the working level regarding discrepancies observed between navigation solutions. Residuals between the expected and observed Doppler signature of the more frequent Angular Momentum Desaturation (AMD) events was noted but only informally reported.

E3: As MCO approached Mars, three orbit determination schemes were employed. Doppler and range solutions were compared to those computed using only Doppler or range data. The Doppler-only solutions consistently indicated a flight path insertion closer to the planet. These discrepancies were not resolved.

E5: On September 8, 1999, the final planned interplanetary Trajectory Correction Maneuver 4 (TCM-4) was computed. This maneuver was expected to adjust the trajectory such that soon after the Mars orbital insertion (MOI) burn, the first periapse altitude (point of closest approach to the planet) would be at a distance of 226 km. This would have resulted in the second periapse altitude becoming 210 km, which was desired for the subsequence MCO aerobraking phase.

E6: TCM-4 was executed as planned on September 15, 1999.

E7: Mars orbit insertion was planned for September 23, 1999. During the week between TCS-4 and MOI, orbit determination processing by the operations navigation team indicated that the first periapse distance had decreased to the range of 150-170 km.

E8: During the 24 hours preceding MOI, MCO began to feel the strong effects of the Mars gravitational field, and tracking data was collected to measure this and incorporate it into the orbit determination process.

E9: Approximately one hour prior to MOI, processing of the more accurate tracking data was completed. Based on this data, the first periapse altitude was calculated to be as low as 110 km. The minimum periapse altitude considered survivable by MCO is 80 km.

E11: The MOI engine start occurred at 09:00:46 (UTC) on September 23, 1999. All systems performed nominally until Mar's occultation loss of signal at 09:04:52 (UTC), which occurred

49 seconds earlier than predicted. Signal was not reacquired following the 21 minute predicted occultation interval.

E12: Exhaustive attempts to reacquire signal continued through September 25, 1999, but were unsuccessful.

E13: On September 29, 1999, it was discovered that the small forces ΔV's reported by the spacecraft engineers for use in orbit determination solutions was low by a factor of 4.45 (1 pound force=4.45 Newtons) because the inpulse bit data contained in the AMD file was delivered in lb-sec instead of the specified and expected units of Newton-sec. After the fact navigation estimates, using all available data through loss of signal, with corrected values for the small forces ΔV's, indicated an initial periapsis (lowest point of orbit) of 57 km, which was judged too low for spacecraft survival.

3.3.3 Identified Cause of the Accident

The accident investigation team identified what they called the "root" cause of the accident as well as factors labeled as "contributory." According to NASA Procedures and Guidelines (NPG 8621 Draft 1), a *root cause* is defined as: "Along a chain of events leading to a mishap, the first causal action or failure to act that could have been controlled systematically either by policy/practice/procedure/ or individual adherence to policy/practice/procedure." The root cause identified here is:

> The failure to use metric units in the coding of a ground software file, "Small Forces," used in trajectory models. Specifically, thruster performance data in English units instead of metric units was used in the software application code titled SM_FORCES (small forces). The output from the SM_FORCES application code as required by a Software Interface Specification (SIS) was to be in metric units of Newton-seconds (N-s). Instead, the data was reported in English units of pound-seconds (lb-s).
>
> The Angular Momentum Desaturation (AMD) file contained the output data from the SM_FORCES software. The SIS, which was not followed, defines both the format and units of the AMD file generated by ground-based computers. Subsequent processing of the data from the AMD file by the navigation software algorithm, therefore, underestimated the effect on the spacecraft trajectory by a factor of 4.45, which is the required conversion factor from force pounds to Newtons. An erroneous trajectory was computed using this incorrect data.

The NASA Procedures document used defines a contributing cause as: "A factor, event, or circumstance that led directly or indirectly to the dominant root cause, or which contributed to the severity of the mishap." Based on this definition, the investigators determined there were eight contributing causes that relate to recommendations for the Mars Polar Lander.

1. Undetected mismodeling of spacecraft velocity changes.
2. Navigation team unfamiliar with spacecraft.
3. Trajectory correction maneuver number 5 not performed.
4. System engineering process did not adequately address transition from development to operations.
5. Inadequate communications between project elements.

6. Inadequate training.

7. Verification and validation process did not adequately address ground software.

3.3.4 Hierarchical Model

Only the basic chain of events is included for MCO and not an expanded chain: Compared to the other accident reports, there is little in this one about events preceding the MCO launch, particularly those events during engineering that might explain when and how the software errors were introduced and why they were not detected during review and testing. This is unfortunate and greatly limits what could be learned from this accident in order to prevent future accidents.

I found it helpful, however, in understanding the accident to create an expanded chain of events for the operations phase pieced together from information in the report. The added events are italicized so that the MCO report authors are not unfairly criticized for any of my mistakes in augmenting the event chain from limited information.

E1 *The operations navigation team joined the project shortly before launch.*

E2 MCO was launched on December 11, 1998 and began a trajectory toward Mars.

E3 Throughout spring and summer of 1999, concerns existed at the working level regarding discrepancies observed between navigation solutions. Residuals between the expected and observed Doppler signature of the more frequent AMD events was noted but only informally reported.

- **E3a** *During the first four months of the MCO cruise flight, the AMD (Angular Momentum Desaturation) files were not used in the orbit determination process because of errors. Instead, the operations navigation team used email from the contractor to notify them when an ADM desaturation event was occurring, and they attempted to model trajectory perturbations on their own, based on this timing information.*
- **E3b** *The MCO AMD events occurred 10-14 times more often than was expected by the operations navigation team.*
- **E3c** *The files were not fixed for four months. In April, 1999, the operations team began using the correctly formatted files.*
- **E3d** *Within a week of starting to use the files, they were found to contain anomalous data that was indicating underestimation of the trajectory perturbations due to desaturation events.*
- **E3e** *Throughout spring and summer of 1999, concerns regarding discrepancies observed between navigation solutions were reported only informally.*

E4 As MCO approached Mars, three orbit determination schemes were employed. Doppler and range solutions were compared to those computed using only Doppler or range data. The Doppler-only solutions consistently indicated a flight path insertion closer to the planet. These discrepancies were not resolved.

E5 On September 8, 1999, the final planned interplanetary Trajectory Correction Maneuver-4 (TCM-4) was computed. This maneuver was expected to adjust the trajectory such that soon after the Mars orbital insertion (MOI) burn, the first periapse altitude (point of closest approach to the planet) would be at a distance of 226 km. This would have resulted in

the second periapse altitude becoming 210 km, which was desired for the subsequence MCO aerobraking phase. After the fact navigation estimates, using all available data through loss of signal, with corrected values for the small forces ΔV's, indicated an initial periapsis (lowest point of orbit) of 57 km which was too low for spacecraft survival.

E6 TCM-4 was executed as planned on September 15, 1999.

E7 Mars orbit insertion was planned for September 23, 1999. During the week between TCS-4 and MOI, orbit determination processing by the operations navigation team indicated that the first periapse distance had decreased to the range of 150-170km.

E8 During the 24 hours preceding MOI, MCO began to feel the strong effects of the Mars gravitational field, and tracking data was collected to measure this and incorporate it into the orbit determination process.

E9 Approximately one hour prior to MOI, processing of the more accurate tracking data was completed. Based on this data, the first periapse altitude was calculated to be as low as 110 km. The minimum periapse altitude considered survivable by MCO is 80 km.

E10 *During the MCO approach, a contingency maneuver plan was in place to execute an MCO Trajectory Correction Maneuver 5 (TCM-5) to raise the second periapsis passage of the MCO to a safe altitude. For a low initial periapse, TCS-5 could also have been used shortly before the Mars Orbit Insertion (MOI) as an emergency maneuver to attain a safer altitude. A request to perform a TCM-5 was discussed verbally shortly before the MOI onboard procedure was initiated, but was never executed.*

E11 The MOI engine start occurred at 09:00:46 (UTC) on September 23, 1999. All systems performed nominally until Mar's occultation loss of signal at 09:04:52 (UTC), which occurred 49 seconds earlier than predicted. Signal was not reacquired following the 21 minute predicted occultation interval.

E12 Exhaustive attempts to reacquire signal continued through September 25, 1999, but were unsuccessful.

E13 On September 27, 1999, the operations navigation team consulted with the spacecraft engineers to discuss navigation discrepancies regarding velocity change (ΔV) modeling issues. On September 29, 1999, it was discovered that the small forces ΔV's reported by the spacecraft engineers for use in orbit determination solutions was low by a factor of 4.45 (1 pound force=4.45 Newtons) because the inpulse bit data contained in the AMD file was delivered in lb-sec instead of the specified and expected units of Newton-sec. After the fact navigation estimates, using all available data through loss of signal, with corrected values for the small forces ΔV's, indicated an initial periapsis (lowest point of orbit) of 57 km which was judged too low for spacecraft survival.

Level 2 Conditions

Conditions not in the report could obviously not be included. Potentially missing information is described under Level 3 systemic factors.

C1: A separate navigation team was used for the MCO development and test phase. The operations navigation team came onboard shortly before launch and did not participate in any

of the testing of the ground software, in the Preliminary Design Review, nor in the Critical Design Review process. Critical information on the control and desaturation of the MCO momentum was not passed on to the operations navigation team. [↓E3d, E3e, E4]

C2: During the first four months of the MCO cruise flight, the ground software AMD files were not used in the orbit determination process because of multiple file format errors and incorrect quaternion (spacecraft attitude data) specifications. Instead, the operations navigation team used email from the contractor to notify them when an ADM event was occurring, and they attempted to model trajectory perturbations on their own, based on this timing information. Four months were used to fix the file problems, and it was not until April 1999 that the operations team could begin using the correctly formatted files. [↓E3]

C3: The MCO solar array was asymmetrical relative to the spacecraft body as compared to Mars Global Surveyor (MGS), which had symmetrical solar arrays. This asymmetrical design significantly increased the Sun-induced (solar pressure induced) momentum buildup on the spacecraft. To minimize this effect, a daily 180° flip was baselined to cancel the angular momentum buildup. Systems engineering trade studies performed later determined that this so-called "barbecue" mode was not needed, and it was deleted from the spacecraft operations plan. These systems engineering decisions and their impact on the spacecraft and the required operational procedures were not communicated to the operations navigation team. The increased AMD events resulting from this decision coupled with the erroneous angular momentum (impulse) data contributed to the MCO loss [↓E3b, E7].

C4: The file format and content errors early in the cruise mission contributed to the operations team not being able to quickly detect and investigate the incorrect units problem[5]. [↓E3a, E3c, E3d]

C5: The small forces ΔV's reported by the spacecraft engineers for use in orbit determination solutions was low by a factor of 4.45 (1 pound force=4.45 Newtons) because the impulse bit data contained in the AMD file was delivered in lb-sec instead of the specified and expected units of Newton-sec. [↓E3d]

Immediately after a thruster firing, the velocity change (ΔV) is computed using an impulse bit and thruster firing time for each of the thrusters. The results are critical for accurately determining the spacecraft's trajectory. The calculation of the thruster performance is carried out both onboard the spacecraft and in ground support system computers. Mismodeling occurred only in the ground software.

The AMD software installed on the spacecraft used metric units for the computation and was correct. In the ground software, the impulse bit reported to the ADM file was in English lbs-s rather than the metric units specified. Subsequent processing of these impulse bit values from the AMD file by the navigation software underestimated the effect of the thruster firings on the spacecraft trajectory. [↓E2]

C6: The software developers of the ground software "Small Forces" file did not follow the data format specified in the Software Interface Specification. [↓E2]

[5]It is surprising that these errors did not figure more in the report and the causes identified. An argument could be made that the errors at least indirectly led to the loss because the operations team could not use the software directly implicated (and thus did not have a chance to detect the units error) until relatively late. Perhaps more important, the fact that there were other errors in the software or the data files was a red flag to the investigation team that something was seriously wrong with the software process beyond simply one person misusing units.

C7: Two factors influenced the investigation of the mismodeling of the AMD maneuvers during the flight of MCO to Mars. First, there was limited observability of the total magnitude of the thrust because of the relative geometry of the thrusters used for AMD activities and the Earth-to-spacecraft line of sight. The navigation team can only directly observe the thrust effects along the line of sight using the measurements of the spacecraft's Doppler shift. In the case of MCO, the major component of thrust during an AMD event was perpendicular to the line-of-sight. The limited observability of the direct effect of the thruster effects meant a systematic error due to the incorrect modeling of the thruster effects could be present but undetected in the trajectory estimation. Second, the primary component of the thrust was also perpendicular to the spacecraft's flight path. In the case of MCO, this perturbation to the trajectory resulted in the actual spacecraft trajectory at the closest point of approach to Mars being lower than that estimated by the navigation team. [↓3d]

C8: The operations navigation team was not intimately familiar with the attitude operations of the spacecraft, especially with regard to the MCO attitude control system and related subsystem parameters. This unfamiliarity caused the operations navigation team to perform increased navigation analysis to quantify an orbit determination residual error. The error was masked by the lack of information regarding the actual velocity change (ΔV) imparted by the AMD events. A line-of-sight error was detectable in the processing of the tracking measurement data, but its significance was not fully understood. [↓E3d]

C9: Analysis, tests, and procedures to commit to a TCM-5 in the event of a safety issue were not completed nor attempted. Therefore, the operations team was not prepared for such a maneuver. The MCO project did not have a defined set of Go/NoGo criteria for using TCM-5. There was no process in place to review the evaluation and decision criteria by the project and subsystem engineers before commitment to TCM-5. [↓E10]

C10: The MOI maneuver timeline onboard the spacecraft took priority over TCM-5. This onboard procedure did not allow time for the upload, execution, and navigation verification of such a maneuver [↓E10].

C11: Any change to the baselined orbit scenario could have exceeded the time for the MCO aero-braking phase when MCO was needed to support the communications of the MPL spacecraft. The spacecraft operations or operations navigation personnel did not fully understand the criticality to perform TCM-5. [↓E10]

C12: Science experts were not fully involved in the decisions not to perform TCS-5 prior to Mars orbit insertion. [↓E10]

C13: The operations navigation team did not communicate their trajectory concerns effectively to the spacecraft operations team or project management [↓E3d, E3e, E4].

C14: The operations navigation team had insufficient technical knowledge of the spacecraft, its operation, and its potential impact on navigation computations. They assumed that MCO had MGS heritage and that therefore much of the MCO hardware and software was similar to that on MGS. For example, the operations navigation team did not know until long after launch that the spacecraft routinely calculated, and transmitted to Earth, velocity change data for the AMC events. An early comparison of these spacecraft-generated data with the tracking data might have uncovered the units problem that ultimately led to the loss of spacecraft. [↓E3, E10]

C15: When conflicts in the data were uncovered, the team relied on email to solve problems, instead of formal problem resolution processes such as the Incident, Surprise, Anomaly (ISA) reporting procedure. Failing to adequately employ the problem tracking system contributed to this problem "slipping" through the cracks. [↓E3c]

C16: During the cruise phase and immediately preceding MOI, inadequate statistical analyses were employed to fully understand the dispersions of the trajectory and how these would impact the final MOI sequence. This resulted in a misunderstanding of the actual vehicle trajectory. [↓E4, E5]

C17: A number of reviews took place without the proper representation of key personnel. Operations navigation personnel did not attend the spacecraft Preliminary and Critical Design Reviews. [↓E1]

C18: The overall project plan did not provide for a careful handover from the development project to the very busy operations project. MCO was the first JPL mission to transition a minimal number of the development team into a multi-mission operations team. Very few JPL personnel and no MCO navigation personnel, transitioned with the project. Also, MCO was the first mission to be supported by the multi-mission MSOP team. This was the first time a handover occurred of a Mars-bound spacecraft from a group that constructed and launched it to a new, multi-mission operations team. [↓E1]

C19: During the months leading up to MCO MOI, the MSOP team had some key personnel vacancies and a change in top management. The operations navigation personnel in MSOP were working MGS operations, which had experienced some inflight anomalies. They were expecting MCO to closely resemble MGS. They had not been involved in the initial development of the navigation plan and did not show ownership of the plan, which had been handed off to them by the MCO development team. The MSOP had no systems engineering and no mission assurance personnel who might have acted as an additional set of eyes in the implementation of the process. [↓all events]

C20: There were no independent peer reviews to validate the operations navigation team navigation analysis technique and to provide independent oversight of trajectory analyses. [↓E3c, E4]

C21: During the time leading up to the loss of MCO, the Mars Surveyor Operations Project (MSOP) was running three missions simultaneously (MGS, MCO, MPL). This tended to dilute the focus on any one mission, such as MCO. During the time before Mars orbit insertion (MOI), MCO navigation was handled by the navigation team lead and the MCO navigator. Twenty-four hour navigation coverage is not possible for the single navigator of MCO. [↓almost all events]

C22: The Software Interface Specification was developed but not properly used in the small forces ground software development and testing. [↓E3d]

C23: End-to-end testing to validate the small forces ground software performance and its applicability to the specification "did not appear" to be accomplished[6]. [↓E3d]

C24: It was "not clear" that the ground software independent verification and validation was accomplished for MCO. [↓E3d]

[6]The vague terminology used in the report to describe the conditions in C23, C24, and C25 seems to imply that these were not investigated

C25: The interface control process and the verification of specific ground system interfaces was not completed or was completed with insufficient rigor. [↓E3d]

Level 3 Systemic Factors

This report included almost no information about why the software units error was made (the labeled root cause of the accident). This may be partly related to the strange definition of *root cause* imposed by NASA standards. It may also reflect the fact that the error was introduced by a NASA contractor. The MPL report is similarly vague about the LMA software development process. The bulk of the detailed investigation focused on why the error was not caught by the operations team. In some respects, this focus makes the report more similar to the aircraft accident reports, where all the focus is on the pilot and pilot errors, than the other spacecraft accident reports.

I. Flaws in the Safety Culture

S1 *Assuming risk decreases during operations*

The safety and mission assurance group was not properly engaged during the operations phase. MSOP did not have a mission assurance manager. Such a presence earlier in the program might have helped to improve project communication and ensure that project requirements were met. The operations phase was also not adequately staffed. (↓C19, C21)

S2 *Overconfidence and complacency*

JPL's navigation of interplanetary spacecraft has worked well for 30 years. In the case of MCO, there was widespread perception that "orbiting Mars is routine." This perception resulted in inadequate attention to navigation risk management.

S3 *Emphasis on finding or correcting errors rather than preventing them*

The emphasis both in the investigation and in the report was on why the error was not detected after it was introduced rather than on why it occurred in the first place. In the report on the accident, a conclusion was that:

> The Board recognizes that mistakes and deficiencies occur on all spacecraft projects. It is imperative that all spacecraft projects have sufficient processes in place to catch mistakes before they become detrimental to mission success. Unfortunately for the Mars Climate Orbiter, the processes in place did not catch the root cause and contributing navigational factors that ultimately led to mission failure.

While this emphasis may simply reflect that of the MCO accident report writers, there is some evidence it reflected also the attitude of the development team.

S4 *Flawed resolution of conflicting goals and inadequate emphasis on risk management.*

A recommendation in the reports is to pay greater attention to risk identification and management.

> Risk management should be employed throughout the life cycle of the project, much the way cost, schedule, and content are managed. Risk, therefore, becomes the "fourth dimension" of project management—treated equally as important as cost and schedule.

The Board found that the project management team appeared more focused on meeting mission cost and schedule objectives and did not adequately focus on mission risk.

II. Ineffective Organizational Structure and Communication Deficiencies

S5 *Diffusion of responsibility and authority*

The report suggests that authority and accountability were a significant issue in the accident and that roles and responsibilities were not clearly allocated. There was virtually no JPL oversight of Lockheed Martin Astronautics subsystem development.

Project leadership did not instill the necessary sense of authority and responsibility in workers that would have spurred them to broadcast problems they detected so those problems might be "articulated, interpreted, and elevated to the highest appropriate level, until resolved."

> Line managers at the field centers need to be held accountable for the success of all missions at their centers....The line management should be held accountable for asking the right questions at meeting and reviews, and getting the right people to those reviews to uncover mission-critical issues and concerns early in the program. Institutional management also must be accountable for ensuring that concerns raised in their area of responsibility are pursued, adequately addressed, and closed out.

MCO was the first JPL mission to transition a minimal number of people from development to operations. It is of interest to note that this lack of end-to-end responsibility for a project has led to related problems in software engineering: Software maintenance is usually performed by a group separate from the development team, leading to very costly and time-consuming maintenance activities and the introduction of errors due to misunderstanding the software design and design decisions. There have been recommendations that software engineering go in the exact opposite direction from the new JPL strategy, that is, the same team be responsible for both development and maintenance (required software changes during operations). (↓C1, C13, C15, C18, C21)

S6 *Organizational and communication problems between partners*

MCO was the first mission to be supported by a geographically distributed, multi-mission operations team. Several lessons were learned about the difficulty of implementing such a distributed organization including communication problems between the partners. The report concluded that

> The overall project plan did not provide for a careful handover from the development project to the very busy operations project. Transition from development to operations—as two separate teams—disrupted continuity and unity of shared purpose. At the same time, NASA management of out-of-house missions was changed from "oversight" to "insight"[7]—with far fewer resources devoted to contract monitoring.

Another example of the problems was the isolation of the operations navigation team from the MCO development and operations team, as well as from its own line organization. For example, they did not know until long after launch that the spacecraft sent down tracking data that could have been compared with the ground data. (↓C1, C2, C3, C18)

[7] The same words appear in the Titan IV/Milstar report

S7 *Limited communication channels and poor information flow*

The communication between the project development team and the operations team was obviously deficient. For example, the report notes that "Critical information on the control and desaturation of the MCO momentum was not passed on to the operations navigation team." A decision was made that the barbecue mode was not needed and it was deleted from the spacecraft operations plan, but the operations navigation team was never notified. Communication was poor in the other direction too. Throughout spring and summer, concerns regarding discrepancies observed between navigation solutions were reported by the navigation operations team only informally and were not communicated effectively to the spacecraft operations team or project management. Email was used to solve problems rather than the problem tracking system.

> A critical deficiency in Mars Climate Orbiter project management was the lack of discipline in reporting problems and insufficient follow-up. The primary, structured problem-reporting procedure used by the Jet Propulsion Laboratory—the Incident, Surprise, Anomaly process—was not embraced by the whole team.

(↓C1, C2, C3, C8, C11, C12, C15)

III. Ineffective or Inadequate Technical Activities

S8 *Flawed review process*

The report recommends that "NASA conduct more rigorous, in-depth reviews of the contractor's and the team's work—something that was lacking on the Mars Climate Orbiter" and concludes that the "operations navigation team could have benefited from independent peer reviews to validate their navigation analysis technique and to provide independent oversight of the trajectory analysis."

There is no mention of the quality assurance function on software, but there is good reason to believe from the superficial evidence (and from other accidents) that something was awry here. (↓C19, C20)

S9 *Inadequate system engineering*

The Board concluded that they saw strong evidence that the systems engineering team and the systems processes were inadequate on the MCO. For example, the report notes that navigation requirements were set at too high a management level and that there was insufficient flowdown to the subsystem level and inadequate validation of the requirements. There were also many problems associated with the transition from development to operations.

In general, the report concludes that the operations phase did not have an adequate systems engineering function for operations. This deficiency led to missing a number of opportunities to identify the units problem leading to missing loss, contributed to the lack of understanding on the part of the navigation team of essential spacecraft design characteristics and the navigation challenges, resulted in inadequate contingency preparation process to address unpredicted performance during operations and a lack of understanding of several critical operations tradeoffs, exacerbated the communications difficulties between the subsystem engineers (navigation, AACS, propulsion), and exacerbated the isolation of the navigation team. (↓C8, C9, C10, C16, C18, C19)

S10 *Inadequate software engineering*

There was enough evidence to conclude that there were serious problems in the software development process, but the Board for some reason failed to investigate these. For example, during the first four months of the MCO cruise flight, the ground software files could not be used for orbit determination because of multiple file format errors and incorrect quaternion specifications. It is surprising that the MCO accident report does not suggest that these errors might reflect the same underlying process deficiencies as the later problem with the Small Forces files not following the data format specified in the Software Interface Specification.

The report instead suggests that the Small Forces error reflects poor training on the part of the software engineer responsible. However, nobody intelligent enough to write software should have to be told to read the interface specification. On the face of it, the error appears to be a case of gross negligence or incompetence. But such is unlikely in professionals hired for a technical position, so it would have been useful if the investigators had investigated this further. Some possible explanations include the programmers not having access to the specification (specifications are often finished after software development has begun), a poorly designed specification that made finding the information difficult, software programmers without the necessary science background to understand the units difference, etc.

There is also the question of why the error was not caught by testing or inspections. The report has a vague statement that end-to-end testing to validate the small forces ground software performance "did not appear" to be accomplished; that it was "not clear" that ground software independent verification and validation was accomplished for MCO; and that the interface control process and verification of specific ground system interfaces "either was not completed or was completed with insufficient rigor." The information provided here should be compared with that provided for a similar type of error that resulted in the loss of a Titan IV/Milstar satellite.

There was also brief mention in the report of other clues to software engineering problems such as "JPL's process of cowboy programming" and the "use of 20-year-old trajectory code that can neither be run, seen, or verified by anyone or anything external to JPL." These hints imply that there may be specification deficiencies and even lack of professionalism among JPL software developers (although they do not imply anything about the LMA software engineers who developed the Small Forces software). (↓C2, C4, C5, C6, C22, C23, C24, C25)

S11 *Inadequate system safety engineering*

As with other accidents described in this paper, there was "inadequate identification of mission-critical elements throughout the mission." The criticality of specific elements of the ground software that impacted the navigation trajectory was not identified, for example. I believe this common problem stems from the use of inappropriate tools, like FMEA and FMECA, which were developed for hardware and do not work for the different types of "failure" modes exhibited by software. Clearly, better tools are needed.

There was also "absence of a process, such as a fault tree analysis, for determining 'what could go wrong' during the mission." Such an analysis could have assisted with the mission contingency planning.

Finally, the report concludes that the "safety and mission assurance group was not properly engaged during the operations phase." The little information available in the report suggests that they were not properly engaged during the development phase either.

Despite the omissions in the report, it was the only one besides Challenger to recommend instituting a classic system safety engineering program, i.e., continually performing system analyses necessary to explicitly identify mission risks and communicate these risks to all segments of the project team and institutional management; vigorously working with this team to make tradeoff decisions that mitigate these risks in order to maximize the likelihood of mission success; and regularly communicating the progress of the risk mitigation plans and tradeoffs to project, program, and institutional management.

S12 *Deficiencies in information collection and use*

The report concludes that lack of discipline in reporting problems and insufficient followup was at the heart of the mission's navigation mishap.

> Steps must be taken to aggressively mitigate unresolved problems by creating a structured process of problem reporting and resolution. Workers should be trained to detect, broadcast, interpret, and elevate problems to the highest level necessary until resolved."

There is a structured process for reporting problems at NASA but it was not used. Not enough information is included in the report to determine why, but possible reasons are that it is difficult or unwieldy to use or involves too much overhead. (↓C15)

S13 *Operational personnel do not understand the automation*

Clearly the operations navigation team had inadequate understanding of the automation and therefore were unable to effectively monitor its operation. (↓C8, C11, C14, C18)

3.4 Mars Polar Lander

3.4.1 Background

Like MCO, Mars Polar Lander (MPL) was part of the Mars Surveyor program. It was launched from Cape Canaveral January 3, 1999 atop the same type of Delta II launch vehicle as MCO. It carried instruments to map the planet's surface, profile the structure of the atmosphere, detect surface ice reservoirs, and dig for traces of water beneath Mars' surface. The lander also carried a pair of basketball-sized microprobes that were (presumably) released as the lander approached Mars. The microprobes, part of NASA's New Millenium Program, were to penetrate up to 1 meter below the Martian surface to test 10 new technologies, including a science instrument to search for traces of water ice.

Like the successful Mars Pathfinder, MPL was to dive directly into the Martian atmosphere, using an aeroshell and parachute scaled down from Pathfinder's design to slow its initial descent. The smaller MPL did not use airbags (as did Pathfinder) but instead relied on onboard guidance, radar, and retro-rockets to land softly on the layered terrain near the south polar cap a few weeks after the seasonal carbon dioxide frosts had disappeared. After the heat shield was jettisoned, a camera was to take a series of pictures of the landing site as the spacecraft descended.

The lander was encased in an aerodynamic entry body consisting of a forward heatshield and a backshell (aft heatshield), which was to separate from the cruise stage about 5 minutes before atmospheric entry. The subsequent entry, deployment, and landing (EDL) sequence—with parachute deployment, heatshield jettison, lander leg deployments, radar ground acquisition, separation of backshell with parachute from the lander, and powered descent to the surface—was to last about 5.5 minutes.

The three landing legs were to be deployed from their stowed condition to the landed position at an altitude of about 1500 meters while the lander was still attached to the parachute. Each leg was fitted with a Hall Effect magnetic sensor that generates a voltage when its leg contacts the surface of Mars. The descent engines were to be shut down by a command initiated by the flight software when the first landing leg sensed touchdown. If the touchdown sensor in that leg failed to detect the touchdown, the second leg to touch down was to trigger the engine shutdown. This logic was intended to prevent the lander from tipping over when it has a skewed attitude relative to the surface during touchdown. It is important to get the engine thrust terminated within 50 milliseconds after touchdown to avoid overturning the lander. The flight software was also required to protect against a premature touchdown signal or a failed sensor in any of the landing legs.

3.4.2 Chain of Events

Because there was no telemetry data transmitted from the satellite during the EDL sequence, a chain of proximate events is not possible.

3.4.3 Identified Cause

The spacecraft design provided no entry, descent, and landing data, which prevented an analysis of MPL performance to provide some certainly about the cause. The investigation essentially involved doing a hazard analysis to identify potential causes. This exercise identified many fundamental failure modes that could cause loss of the lander as well as strategies for their mitigation. As the report concludes, a system-level FTA or similar hazard analysis method should be employed as an element of the systems engineering process. As the design evolves, the hazard analysis should be updated and the results summarized at each major project review. One positive result of this loss is that the value of systems engineering and system safety has been reinforced.

The report identified the following candidate MPL failure causes:

- Premature shutdown of descent engines.
- Surface conditions exceeding landing design capabilities
- Loss of control due to dynamic effects.
- Landing site not survivable.
- Backshell/parachute contacts lander.
- Loss of control due to center-of-mass offset.
- Heatshield fails due to micrometeoroid impact.

The investigation board found compelling evidence that premature shutdown of the descent engines due to spurious signals generated at lander leg deployment was the cause of the loss of MPL. Without flight data, however, the other failure modes cannot be ruled out.

The touchdown sensors characteristically generate a false momentary signal at leg deployment. This behavior was understood and the flight software was required to ignore it. The software requirements did not specifically describe these events, however, and consequently the software designers did not properly account for them. According to this accident scenario, the software interpreted the spurious signals generated at leg deployment (at an altitude of about 1500 meters) as valid touchdown events. When the sensor data were enabled at an altitude of 40 meters, the software shut down the engines and the lander free fell to the surface, impacting at a velocity of 22 meters per second (50 miles an hour) and was destroyed.

In the absence of flight data, there is no way to determine whether the lander actually reached the terminal descent propulsion phase of the mission. If it did, however, extensive tests have shown

that it would almost certainly have been lost due to the scenario outlined above, i.e., premature engine shutdown.

3.4.4 Hierarchical Model

For this report, I have generated an expanded chain of events for the scenario identified as most probable, i.e., premature engine shutdown by the software due to incorrect response to spurious signals upon leg deployment. Although this chain had to be created from what is only speculation, it is included for comparison with the other accident models in this report.

E1 At some point during system requirements specification or changes to the system requirements specification, a system requirement not to use the touchdown sensor data until 12 meters above the surface was not translated into a software requirement (see Figure 3.2).

E2 Landing leg deployment tests established the likelihood of transient response of the Hall Effect sensors to the leg dynamic effects. The software requirements were not changed.

E3 Design and code walkthroughs were performed but the requirements error was not found.

E4 Unit test did not detect the problem.

E5 Software integration tests did not detect the problem.

E6 A system leg deployment was performed on June 4 1998 during spacecraft testing with the flight software touchdown code operating. Even though transients due to dynamic response of the Hall Effect sensors were probably present, they were not detected, nor was touchdown detected when technicians pushed up on the footpads to simulate touchdown.

E7 A wiring error was detected in the June 4 test, and the legs were rewired to correct the error. The technicians again pushed up on the footpads and the sensors indicated a touchdown had been sensed.

E8 On June 16, 1997, a component test of the landing leg deployment was performed. That test provided an indication of the transient response from the Hall Effect sensors to the dynamics of the deployment. The importance of the transient was not recognized.

E9 A code walkthrough of the touchdown-sensing code was held on June 30, 1997, without consideration of the effects of a Hall Effect sensor transient on the outcome of the sequence.

E10 MPL was launched January 3, 1999, for arrival at Mars on December 3, 1999.

E11 The spacecraft approached Mars on December 3 in apparent good health. A final trajectory-correction maneuver, TCM-5, was executed 6.5 hours before entry.

E12 At 12:02 pm PST, the spacecraft slewed to entry attitude. At this attitude, the antenna pointed off-Earth, and the signal was lost as expected. Lander touchdown was expected to occur at 12:14 pm PST, with a 45 minute data transmission to Earth scheduled to begin 24 minutes later.

E13 [This part of the event sequence is based on speculation and later reconstruction of what might have happened given the design of the code.] The lander flight software started cyclic checking of the touchdown event state prior to lander entry.

E14 The software, according to its specification, checked for failed sensors using two consecutive readings. Any sensor indicating touchdown state for the two readings was marked as bad. Presumably the sensor(s) involved in the later problems passed the test and was marked as good.

E15 The three landing legs were deployed from their stowed condition to the landed position at an altitude of about 1500 meters while the lander was still attached to the parachute. Each leg was fitted with a Hall Effect magnetic sensor that generates a voltage when its leg contacts the surface of Mars. A landing leg generated a false momentary signal at leg deployment.

E16 The software recorded the spurious signal as a valid touchdown event. When the sensor data was enabled at 40 meters, the software commanded the engines to shut down.

E17 The engines shut down, the lander free fell to the surface, impacted at a velocity of 22 meters per second (50 miles an hour), and was destroyed.

E18 It was expected that the first data from the DS2 probes would be received on 4 December at 7:25 pm PST, about 7 hours after MPL touchdown. However, no communication from MPL or the probes was received.

E19 After the loss of the lander, the touchdown sensor problem was found during a test run on the 2001 Lander when a test engineer pushed a button indicating a touchdown too early in the test. He released the button when he realized his error and was surprised when thrust termination occurred prematurely.

E20 The report of the unexpected results during the 2001 Lander test led to a failure analysis (including a fault tree analysis) that in turn led to examination of the code and the preparation of code descriptions for reviews by outside reviewers, which uncovered the problem.

Level 2 Conditions

The causal factors identified in the two MPL accident reports (JPL and MPIAT) that I could not trace to any events included in my event chain are omitted here. For example, there is no explanation in either report about how the excessive overtime at LMA contributed to this accident scenario. While the software was "correct" (it satisfied its specification), overwork could have led to errors in the system engineering process or to fewer reviews, but not enough information was provided to determine the particular events affected. In general, I found it difficult to determine the causal factors (beyond the accident mechanism) with the information that was included. Thus, I could not determine the conditions leading to some important events, such as E7, E8, and E9.

C1 Requirements tracing either was not done or was flawed. There is no information provided in the report about how requirements tracing is done by JPL and LMA (or even who is responsible for this activity). (↓E1)

C2 The JPL report says

> This [not changing the software requirements after the lander leg deployment tests established the likelihood of Hall Effect sensor transient signals] may have been the result of the mechanical design personnel not informing the systems and software personnel of the results in a timely manner. Perhaps if the System Engineer was

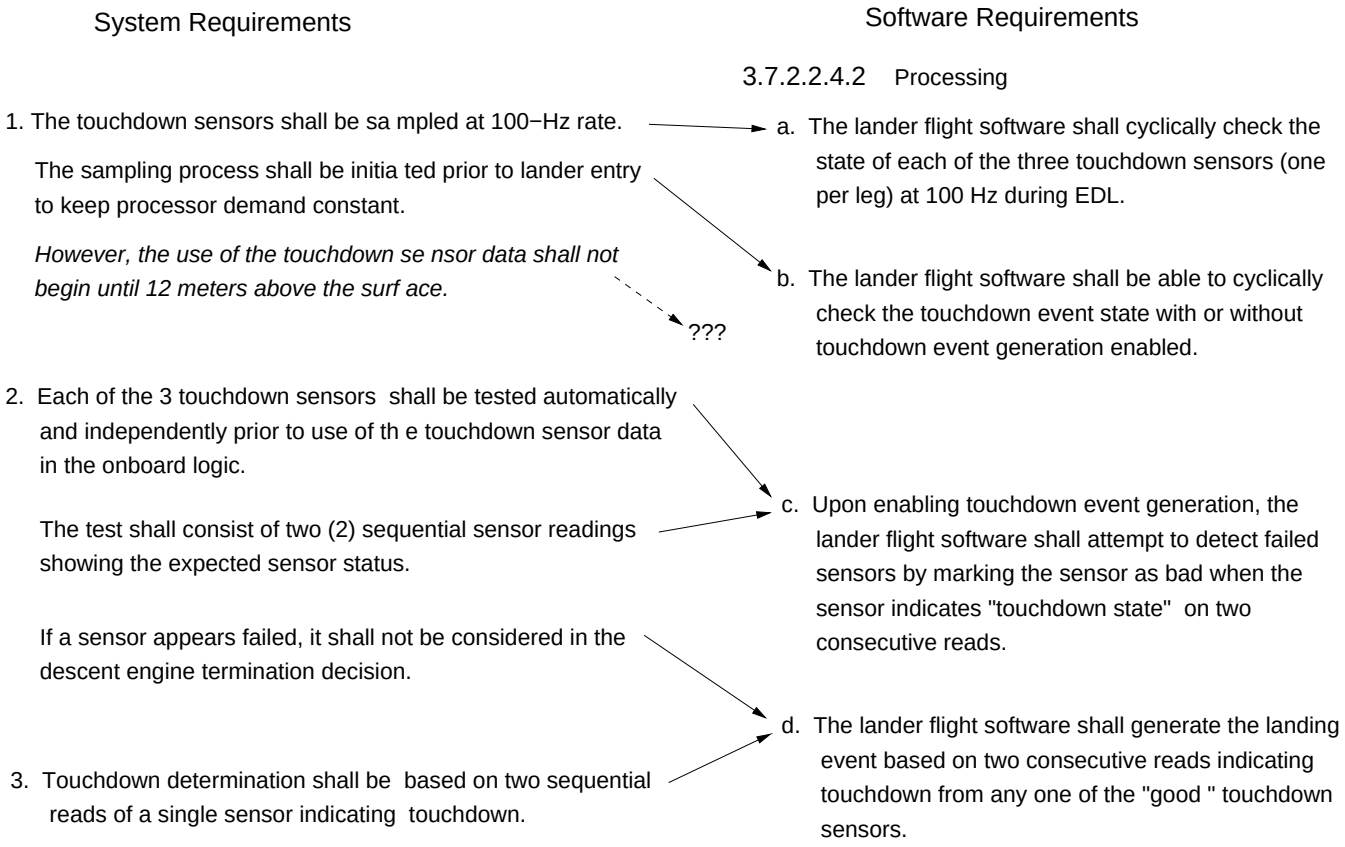

Figure 3.2: The Mapping of MPL System Requirements to Software Requirements

told, he or she may have thought that the problem was solved by the requirement not to use the touchdown sensor data until the 40-meter altitude had been reached. By that time, the transient would no longer be present[8].

(↓E2)

C3 The report attributes the failure to detect the error during design and code walkthroughs at least partially to the reviewers not using logic code diagrams and then suggests using flow charts in the future. Detailed software flow charts, however, have not been found to be much easier to read than code, which is one of the reasons they have faded from use.

Perhaps more important was the fact that software designers and system engineers participating in the walkthrough process were not aware of the transient behavior from the Hall Effect sensors. The Hall Effect sensor Product Integrity Engineer was not present at the walkthroughs and apparently did not review the part of the software specifications and/or software design that used the results of the Hall Effect sensors. There must also have been no mechanism for disseminating new information (in this case about Hall Effect sensor transients) to those designing components affected by that information. Thus, the transient effects were not discussed during the walkthroughs, nor were suitable test cases defined to test the software with conditions representing the transient response of the system. (↓E3)

C4 Unit test did not detect the problem because it was not in the software requirements and therefore they did not test for it. (↓E4)

C5 Software integration tests did not include appropriate test cases again because it was not in the software requirements. (↓E5)

C6 After the June 4 test, it was discovered that the Hall Effect sensors used in the system test were improperly wired because of an error in the wiring diagram, and the wiring error prevented the sensor response from being monitored. As a result, the spurious signals were not identified. (↓E6)

C7 The wiring error was fixed, but the leg deployment test was not repeated. The report may explain why it was not repeated, but I could not find it. (↓E7)

C8 The report might specify the conditions leading to these events, but I was unable to find anything documented. (↓E8, E9)

C9 The software requirements stated that it shall be able to check the touchdown event state with or without touchdown event generation enabled. But it did not state that the use of the touchdown event data should not begin until 12 meters above the ground although this part of the requirement was in the system specification. (↓E13)

C10 The sampling process was initiated prior to entry in order to keep processor demand constant. This requirement was the result of lessons learned from other missions: It was intended to avoid transients in the CPU loading that had caused problems on other programs. This requirement led the software designers to start sampling the sensor data well before the 40-meter altitude had been reached. The report states that in hindsight, it would have been better not to do any sampling of the touchdown sensors prior to the 40-meter altitude. The

[8]The report writers must not have had the opportunity to question the system engineers during the accident investigation or it seems like such speculation would not have been necessary.

transients in the Hall Effect sensor due to landing leg deployment would have been over by then. (↓E13)

C11 The software was required to protect against a failed sensor (see requirement 3.7.2.2.4.2 in Figure 3.2) by marking a sensor as bad when the sensor indicates "touchdown state" on two consecutive reads prior to its use to detect touchdown. It also protects against a premature touchdown signal by requiring two consecutive reads indicating touchdown from any one of the "good" touchdown sensors (sensors not detected to have failed). (↓E14)

C12 The false signal at leg deployment was understood and the flight software should have ignored this event, but the software requirements did not include a requirement to avoid it and the software designers did not know about it. The system level requirements document said that the touchdown data should not be used until 12 meters above the surface (later changed to 40 meters), but did not specifically state the failure modes that the requirement was protecting against—that is, possible transients.

Protection from transient signal behavior of the touchdown sensors was not specifically called out in the requirements. The requirement specifying that "the use of the touchdown sensor data shall not begin until 12 meters above the surface" was intended to eliminate any danger from sensor failure modes, including transients. However, that requirement was not included in the SRS during the requirements flowdown process, and it was not included in the requirements to be verified during system testing. The protection from transient signal behavior was not adequately captured in the system or subsystem requirement specifications, nor in the system-level test requirements. Therefore, system, subsystem, and test teams did not verify transient signal immunity during software and system testing. (↓E15)

Level 3 Systemic Factors

Because of the breadth of the JPL and external MPL accident investigations, a large number of systemic problems were identified and recommendations made. Only those factors are included here that appear to be related in some way to the software flaw identified as the most probable cause of the loss. The mapping of causal conditions to systemic factors was difficult because of insufficient detail in the reports about the conditions leading to specific events involved in the loss.

I. Flaws in the Safety Culture

S1 *Misunderstanding software-related risks*

Throughout the accident reports, there is an emphasis on failures as the cause of accidents and redundancy as the potential solution. The contribution of software to accidents is very different than that of hardware and the typical activities used in system safety engineering must be augmented to handle the software parts of the system. Understanding these differences and implementing alternative types of approaches that will be effective for software has been slow in engineering. For example, on page 22 of the JPL report, there is a figure (6-1) that shows the "potential failure modes" for the EDL sequence. Each type of potential hardware failure or misbehavior is identified for each stage except for software. Instead, a statement "Flight software fails to execute properly" is identified as common to all phases. The problem with this is that it provides no useful information—it is equivalent to substituting a single statement for all the other failures identified on the page with "hardware fails to execute properly."

Software itself cannot be dangerous or cause the loss of the lander—it is an abstraction without the ability to produce energy or alternatively to provide insufficient energy. Instead it contributes to accidents through issuing (or not issuing) instructions to other components in the system. In the case of the identified probable cause of MPL loss, the dangerous behavior would be "software prematurely shuts down the descent engines" and such an identified behavior would be much more helpful during development in mitigating the risk than the general (and meaningless) statement *Software fails to execute properly*, which provides no useful guidance to the system or software designers and reviewers. In fact, software probably should not appear in this figure at all—it should instead be identified in a later (more detailed) analysis step in terms of potential specific undesired software behaviors that could lead to many of the hardware failure modes or incorrect system behaviors listed (including *Premature shutdown of descent engines*, which is included in the figure.) (↓C3, C10, C12)

II. Ineffective Organizational Structure and Communication Deficiencies

S2 *Organizational and communication problems between the partners*

There was essentially no JPL line management involvement or visibility into the project and minimal involvement by JPL technical experts. Many other organizational problems are cited in the two accident reports. (↓C3?)

S3 *Limited communication channels and poor information flow*

One of the consequences of the tight project funding was that many key technical areas were staffed by a single individual. The reports note that the effect of inadequate peer interaction was, in retrospect, a major problem that led to a breakdown in intergroup communications.

In addition, system engineering did not keep abreast of test results from all areas and communicate their findings to other areas of the development project. Establishing and implementing this type of intergroup technical communication is one of the primary roles for system engineering. (↓C2, C3)

III. Ineffective or Inadequate Technical Activities

S4 *Workforce reductions; loss of skills and experienced personnel*

The JPL report concludes that the pressure of meeting the cost and schedule goals resulted in an environment of increasing risk in which too many corners were cut in applying proven engineering practices and the checks and balances required for mission success. Examples include incomplete systems testing and analyses. Single individuals were implementing many important engineering activities, preempting the normal checks and balances and technical interchanges. (↓C3?, C7?)

S5 *Inadequate system engineering process or practices*

In any project as complex as MPL, good system engineering is essential for success. However, the reports note that system engineering resources were insufficient to meet the needs of the project. For example, insufficient systems engineering during the formulation stage led to important decisions that ultimately required more development effort than originally foreseen and inadequate baseline design decisions and hazard identification.

The requirements flowdown process was also clearly flawed. Rationale for requirements did not appear to be included in the specification. During development, fault tree and other hazard analysis activities were used inconsistently and, as noted above, test results and new information derived during testing was not communicated to all the component designers that needed it. (↓C1, C12)

S6 *Violation of basic safety engineering design practices in the digital parts of the system design*

It is not uncommon for sensors involved with mechanical operations, such as the lander leg deployment, to produce spurious signals. The report states that the software designers did not include in the design of the software any mechanisms to protect against transients nor did they think they had to test for transient conditions. These types of mechanisms are or should be included in most real-time software reading sensor data. In addition, not including a check of the current altitude before turning off the descent engines seems so strange on the surface that there must have been some non-obvious but good reason for omitting such a check.

Another basic design principle for safety-critical software is that unnecessary code or functions should be eliminated or separated from safety-critical processing. In this case, code was executing when it was not needed. (↓C10, C12)

S7 *Flawed review process*

The two software errors involved should have been caught in review. Although it is not possible to have everyone present at the reviews, the software specifications should be readable and understandable by all the engineers and a group with appropriate expertise and knowledge should be reviewing them. Code is not readable and reviewable enough (sometimes even by software engineers) to be used in this process. It is my experience that having engineers review software code is a waste of time. It would be equivalent to giving a group of engineers only a detailed wiring diagram containing tens or hundreds of thousands of wires and asking them to identify any design errors. While the use of logic flow diagrams, as recommended in the report, might be useful to software engineers in finding coding errors, it would be extremely difficult and unlikely that engineers reviewing the software would detect requirements specification errors and incomplete or incorrect external software behavior (their most useful role during a software review) from such detailed logic flow diagrams. Engineers should be reviewing the externally visible software behavior presented in a suitably reviewable form while software engineers are most appropriately used for reviewing internal software design and logical flow. That is, the review process is enhanced by separating external behavior (the transfer function across the component) from the internal design to achieve that behavior.

In addition, looking for all errors is difficult. Identifying *unsafe* software behavior and concentrating on it during at least part of the review process would help focus review and make sure that critical issues are adequately considered. Such unsafe behavior will be (or should be) identified in the system safety process before software development begins and the design rationale and design features used to prevent the unsafe behavior should have been documented and can be the focus of such a review. (↓C3)

S8 *Inadequate specifications*

The JPL report notes that the system level requirements document did not specifically state the failure modes the requirement was protecting against—possible transients—and speculates that the software designers or one of the reviewers might have discovered the missing

requirement if they had been aware of the rationale for the requirement. This points out the importance of including requirements and design rationale in specifications.

The small part of the software requirements specification shown in the JPL report seems to avoid all mention of what the software should not do. In fact, some DoD standards even forbid such negative requirements statements. The result is that software specifications often describe nominal behavior well but are very incomplete with respect to required behavior of the software under off-nominal conditions and rarely describe what the software is *not* supposed to do. Most safety requirements and design constraints are best described using such negative requirements or constraints. Most software-related accidents can be traced to such omissions. (↓C4, C5, C9, C12)

S9 *Flawed analysis with respect to software functions*

Both the flight software and the ground uplink loss timer software contained classic logic errors involving incomplete handling of possible states. As such, they are potentially identifiable using formal and informal analysis and review techniques. For example, although identified as a database error, the uplink loss timer software error appears to involve not handling the case where the receive link fails during the EDL sequence and 24 hours elapses without the receipt of a command—an example of a very common type of specification flaw or logic omission. Software hazard analysis and requirements analysis techniques exist (and more should be developed) to detect this type of incompleteness (see [24]). (↓C10, C12)

S10 *Inadequate software engineering process or practices*

Several findings in the JPL report conclude that the LMA software development process is adequate and appropriate. However, there is no detailed description provided and other findings imply inadequacies in the LMA process. For example, the report contains a recommendation that "all hardware inputs to the software must be identified ... The character of the inputs must be documented in a set of system-level requirements." The software interface specification usually contains this information. Were there no interface specifications produced? Such an omission would imply a seriously flawed process. As another example, both of the identified software errors involved unhandled cases in the software that should be part of the software testing and review process. As with the MCO report, almost no detailed information is provided about the LMA software development process used that might help explain the errors. The Titan IV-B accident report (in the next section) provides an example of what might be learned from such an examination. (↓C3, C12)

S11 *Inadequate system safety engineering*

Attention and resources for system safety analysis appear to have been lacking until after the Mars Climate Orbiter loss. At that point, it was too late to be effective in preventing the MPL loss. Hazard analysis and design for safety must start in the earliest stages of development and must be an integral part of every aspect of the process. It is much easier (and cheaper) to design safety into a product than to attempt to find and correct the design after it is complete. Safety must be designed into a product, it cannot be "tested in" or "measured in" afterward.

The lack of flight data was positive in one respect because it made the investigation team perform a classic hazard analysis and demonstrated that such an analysis was both feasible and not prohibitively expensive. All the information to perform this hazard analysis was available before the accident. Such a hazard analysis would have been very helpful in designing MPL and in evaluating alternative designs.

Such hazard analyses have also been found to be extremely useful in building safer software. If performed early, the system hazard analysis can identify potentially hazardous software behavior and then requirements can be written to avoid it. In addition, requirements can be analyzed for safety and protection against potentially hazardous behavior designed into the logic. The JPL report recommends the development of fault trees to define the test cases needed for software, but this approach is too late and very unlikely to be effective anyway. Defining hazardous software outputs does not help much with test case selection because it does not define what inputs to use to produce the identified outputs (if we knew that, we could select all and only test cases that lead to incorrect outputs and testing would be quick and simple—in fact, testing would not be needed). Stress testing and off-nominal testing of the software and checking outputs to determine whether they lead to unsafe behavior is *very* important, but does not substitute for early effort to identify unsafe software behavior and design the software to prevent it.

One of the benefits of using system safety engineering processes is simply that someone becomes responsible for ensuring that particular hazardous behaviors are eliminated, their likelihood reduced, or their effects mitigated in the design. If someone had been assigned responsibility for tracking the hazard of premature engine shutdown, they could have traced the new information about the Hall Effect sensor transients to the logic that used the signals for thruster shutdown. Guaranteeing that this hazard tracking and analysis will occur is, of course, impossible, but assigning responsibility does increase that probability. Almost all attention during any development project is focused on what the system and software are supposed to do. A system safety engineer or software safety engineer is responsible for ensuring attention is also paid to what the system and software are *not* supposed to do—in this case, prematurely shut down the engine—and verifying that it will not occur. (↓C3, C9, C10)

S12 *Unnecessary complexity and software functions*

Once again, the problems stemmed from unnecessary code, but this time it involved executing necessary code at a time when it was not needed. The report states that this decision was based on trying to avoid transients in CPU loading. A tradeoff analysis needs to be performed to make such decisions. But executing software when it is not needed or including unnecessary code raises risk significantly. (↓C10)

S13 *The test and simulation environment did not adequately reflect the operational environment*

The touchdown sensing software was not tested with the lander in the flight configuration. Various other types of test cases were not included. While testing all possible software states is not, in general, feasible (or even more than a very small percentage of them), off-nominal and stress testing is often performed poorly or not at all.

The JPL report concludes:

> The laboratory needs to reinforce the system-level test principle of 'test as you fly, and fly as you test.' Departures from this principle must be carefully assessed, and if they are determined to be necessary, alternate measures, such as independent validation, should be incorporated. Such items must be reflected in the project risk management plan, communicated to senior management for concurrence, and reported at reviews.

(↓C7)

3.5 Titan IV/Milstar

3.5.1 Background

The Titan IV/Milstar accident report makes for an interesting comparison with the Mars Climate Orbiter accident report. The causal factors were seemingly very similar but the reports identify very different causes and different types of contributing factors.

On April 30, 1999, at 12:30 EDT, a Titan IV B-32/Centaur TC-14/Milstar-3 was launched from Cape Canaveral. The booster was a Titan IV B equipped with a Centaur Upper Stage. The mission was to place a Milstar satellite in geosynchronous orbit. As a result of some anomalous events, the Milstar satellite was placed in an incorrect and unusable low elliptical final orbit, as opposed to the intended geosynchronous orbit. Media interest was high due to this mishap being the third straight Titan IV failure and recent failures of other commercial space launches.

Lockheed Martin Astronautics (LMA) was the prime contractor for the mission. The Space and Missile Systems Center Launch Directorate (SMC) was responsible for insight and administration of the LMA contract.

The Lockheed Martin Titan IV B is a heavy-lift space launch vehicle used to carry government payloads such as Defense Support Program, Milstar, and National Reconnaisance Office satellites into space. It can carry up to 47,800 pounds into low-earth orbit and up to 12,700 pounds into a geosynchronous orbit. The vehicle can be launched with no upper stage, or with one of two optional upper stages, providing greater and varied capability.

The Lockheed Martin Astronautics Centaur is a cryogenic, high-energy upper stage. It carries its own guidance, navigation, and control systems, which measures the Centaur's position and velocity on a continuing basis throughout flight. It also determines the desired orientation of the vehicle in terms of pitch, yaw, and roll axis vectors. It then issues commands to the required control components to orient the vehicle in the proper attitude and position, using the main engine or the Reaction Control System (RCS) engines. The RCS provides thrust for vehicle pitch, yaw, and roll control, for post-injection separation and orientation maneuvers, and for propellant settling prior to engine restart.

Milstar is a joint services satellite communications system that provides secure, jam resistant, worldwide communications to meet wartime requirements. Milstar was the most advanced military communications satellite system to that date. The first Milstar satellite was launched February 7, 1994 and the second was launched November 5, 1995. This mission was the third launch.

There were three planned burns during the Centaur flight. The first burn would put the Centaur into a parking orbit. The second would move the Centaur into an elliptical transfer orbit that would carry the Centaur and the satellite to geosynchronous orbit. The third and final burn would circularize the Centaur in its intended geosynchronous orbit. A coast phase was planned between each burn. During the coast phase, the Centaur would progress under its own momentum to the proper point in the orbit for the next burn. The Centaur would also exercise a roll sequence and an attitude control maneuver during the coast periods to provide passive thermal control and settle the main engine propellants in the bottom of the tanks.

This accident is believed to be one of the most costly unmanned losses in the history of Cape Canaveral Launch Operations. The Milstar satellite cost about $800 million and the launcher an additional $433 million.

3.5.2 Chain of Events

E1 The flight performance of the Titan solid rocket motor upgrades and core vehicle was nominal, as was payload fairing separation.

E2 The Centaur separated from the Titan IV B approximately nine minutes and twelve seconds after liftoff and was injected into its orbit satisfactorily.

E3 The first burn occurred at nine and a half minutes after liftoff as planned. Throughout the flight, the Inertial Measurement System (IMS) transmitted zero or near zero roll rate to the Flight Controller software. With no roll rate feedback, the Centaur became unstable about the roll axis and did not roll to the desired first burn orientation. The Centaur began to roll back and forth, eventually creating sloshing of the vehicle liquid fuels in the tanks that created unpredictable forces on the vehicle and adversely affected flow of fuel to the engines. By the end of the first burn (approximately 11 minutes and 35 seconds after liftoff), the roll oscillation began to affect the pitch and yaw rates of the vehicle as well.

E4 Vehicle tumbling and fuel sloshing affected the attained vehicle acceleration, resulting in the Centaur guidance system predicting an incorrect time for main engine shutdown. Centaur's first burn did not achieve intended velocity, and consequently the vehicle was placed in an unintended park orbit.

E5 With first burn shutdown complete, the Centaur entered the first coast phase of the flight. Due to roll instability and transients created by the engine shutdown, the Centaur entered the coast period tumbling.

E6 The Reaction Control System (RCS) immediately attempted to stabilize the vehicle. Late in the park orbit, the Centaur was finally stabilized about the pitch and yaw axes, although it continued to oscillate about the roll axis. In stabilizing the vehicle, the RCS expended almost 85 percent of the RCS system propellant.

E7 The vehicle successfully pointed at the proper attitude for the second burn, and the engines ignited at approximately one hour, 6 minutes, and 28 seconds (T+01:06:28) after liftoff.

E8 Soon after entering the second burn phase, the vehicle again became unstable about the roll axis and began a diverging roll oscillation. Because the second burn is longer than the first, the excess roll commands eventually saturated the pitch and yaw channels. At approximately two minutes into the second burn, pitch and yaw control was lost (as well as roll), causing the vehicle to tumble for the remainder of the burn.

E9 Due to its uncontrolled tumbling during the burn, the vehicle did not achieve the planned acceleration for transfer orbit.

E10 The Centaur continued to tumble in the transfer orbit (the second coast phase) despite continued attempts by the RCS to stabilize the vehicle. The RCS depleted its remaining propellant approximately 12 minutes after the shutdown of the second burn.

E11 The vehicle's third burn started at T+02:34:15. It tumbled throughout the third burn, which started earlier and was shorter than programmed.

E12 Space vehicle separation occurred at approximately T+02:54:15, but the Milstar satellite was placed in a useless low elliptical final orbit, as opposed to the desired geosynchronous orbit (Figure 3.3).

E13 The Mission Director ordered early turn-on at T+01:17:00, but the controllers were unable to contact the satellite for approximately 3 hours. At T+06:14:00, control was acquired and various survival and emergency actions taken.

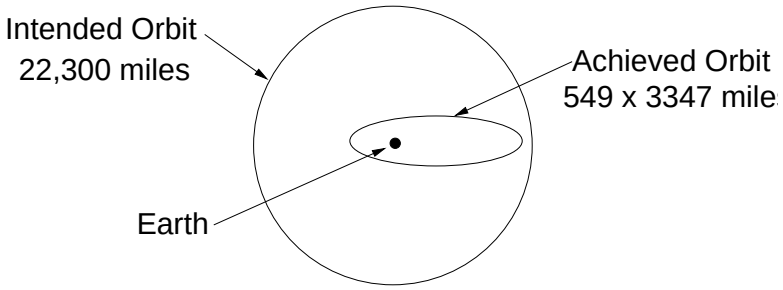

Figure 3.3: Achieved Orbit vs. Intended Orbit

E14 The mission was officially declared a failure on 4 May 1999, but personnel from LMA and the Air Force controlled the satellite for six additional days in order to place the satellite in a non-interfering orbit with minimum risk to operational satellites.

E15 The satellite had been damaged from the uncontrolled vehicle pitch, yaw, and roll movements, and there were no possible actions the ground controllers could have taken in response to the anomalous events that would have saved the mission. It appears the satellite performed as designed, despite the anomalous conditions. It was shut down on 10 May 1999.

3.5.3 Identified Cause

The Accident Investigation Board concluded that:

> Failure of the Titan IV B-32 mission is due to a failed software development, testing, and quality assurance process for the Centaur upper stage. That failed process did not detect and correct a human error in the manual entry of the I1(25) roll rate filter constant entered in the Inertial Measurement System flight software file. The value should have been entered as -1.992476, but was entered as -0.1992476. Evidence of the incorrect I1(25) constant appeared during launch processing and the launch countdown, but its impact was not sufficiently recognized or understood and, consequently, not corrected before launch. The incorrect roll rate filter constant zeroed any roll rate data, resulting in the loss of roll axis control, which then caused loss of yaw and pitch control. The loss of attitude control caused excessive firings of the Reaction Control system and subsequent hydrazine depletion. Erratic vehicle flight during the Centaur main engine burns caused the Centaur to achieve an orbit apogee and perigee much lower than desired, which resulted in the Milstar separating in a useless low final orbit.

In another part of the report, the cause is described as:

> On 5 February 1999, an LMA flight software engineer incorrectly entered a roll rate filter constant into the Inertial Navigation Unit software file. The error went undetected by both the internal quality assurance processes and the independent verification and validation (IV&V) process. The digital roll rate filter is an algorithm with five constants. The filter was designed to prevent the Centaur from responding to the effects of Milstar fuel sloshing and inducing roll rate errors at 4 radians/second. Early in the design phase of the first Milstar spacecraft, the manufacturer asked to filter that frequency. The spacecraft manufacturer subsequently determined filtering was not required at that

frequency and informed LMA. However, LMA decided to leave the filter in place for the first and subsequent Milstar flights for consistency.[9]

The correct value of the filter constant was -1.992476. The specific flight constant entered in error was the I1(25) constant. It was one of forty constants in a file commonly referred to as the I1s. It was incorrectly entered as -0.1992476. The incorrect constant went undetected during the signoff process by the responsible LMA Control Dynamics engineer and became part of a baseline file used for generating all flight software. The software input error was the catalyst for the mishap. The root cause of the mishap was the software development process that allowed a human error to go undetected.[10]

The Accident Investigation Board described the *root cause* in more detail, saying the root cause of the accident was the result of several contributing factors:

- **Software Development**
  - The software development process is not well defined, documented, or completely understood by any of the multiple players involved in that process.
  - The lack of focus and understanding of Inertial Measurement System software operations and Inertial Navigation Unit testing contributes to the poorly defined process for generation and test of the rate filter constants.
  - The software development process allows single point failures for mission critical data.
  - The consolidation of the major contracting companies responsible for Titan/Centaur development and the maturation of the Titan/Centaur program contribute to the poor understanding of the overall software development process.
- **Testing, Validation, and Verification**
  - An independent verification and validation program was developed and approved that does not verify or validate the I1 filter rate constants used in flight.
  - No formal processes exist to check validity of the I1 filter constants or monitor attitude rates once the flight tape is loaded into the Inertial Navigation Unit at Cape Canaveral Air Station (CCAS) prior to launch.
  - Inadequate and indirect communication among the responsible parties prevented correction of the problem observed during testing at Cape Canaveral Air Station prior to launch.
- **Quality/Mission Assurance**
  - A software quality assurance function exists at both Lockheed Martin Astronautics and Defense Contract Management Command, but operates without a detailed understanding of the overall process or program. In addition, transition from oversight to insight is not implemented properly because of that lack of understanding.

[9]Note the same wording was used in the Ariane accident report in explaining the reuse of the software. Both accidents were related to unused and unneeded software that was kept around for "consistency." Perhaps testing and review is not as thorough for such unneeded functions or they simply get less attention during reviews.

[10]Note the difference in the definition of the root cause here and in the MCO report, although the errors were similar. The Air Force definition of root cause allows them to better understand why the accident occurred than that used for MCO (which used the NASA standard definition). The root cause is described in the MCO report as the software units error whereas here the root cause is identified as the software development process and the software error itself only as the "catalyst for the mishap."

– The Space and Missile Systems Center Launch Directorate and the 3^{rd} Space Launch Squadron (3SLS) have undergone personnel reductions and are transitioning from a task oversight to process insight role. That transition has not been managed by a detailed plan. Air Force responsibilities under the insight concept have not been well defined and how to perform those responsibilities has not been communicated to the workforce.

3.5.4 Hierarchical Model

Unlike the other reports, an expanded chain of events that includes the origin of the problem (described as "the sequence of events that led to the mishap") is included. The description of the events involving software development is much more detailed and useful than the other reports involving software errors, probably because the investigators recognized that the root problem was in the software development process and therefore carefully documented it, whereas the other reports did not investigate the source of the software problems in as much detail.

The following event chain comes directly from the report, with only the first event added from a different part of the report (to allow mapping of Level-2 conditions to events):

E0-1 Early in the design phase of the first Milstar spacecraft, the manufacturer asked to filter roll rate errors resulting from Centaur responding to the effects of Milstar fuel sloshing. The spacecraft manufacturer subsequently determined that filtering was not required at that frequency and informed LMA. LMA left the filter in the software.

E0-2 A software Constants and Code Words Memo was generated by the LMA Control Dynamics (CD) group and sent to the LMA Centaur Flight Software (FS) group on December 23, 1997. It provided the intended and correct values for the first I1 constants in hardcopy form. The memo also allocated space for 10 other constants to be provided by LMA Avionics group at a later time and specified a path and file name for an electronic version of the first 30 constants. The memo did not specify or direct the use of either the hardcopy or electronic version for creating the constants database.

E0-3 In early February, 1997, the LMA Centaur FS group responsible for accumulating all the software and constants for the flight load was given discretion in choosing a baseline data file. Some manual manipulation of the input data was required. No specified or documented software development process existed for electronically merging all the inputs into a single file. When the FS engineer tried to access the file specified in the software Constants and Code Words Memo, he found it no longer existed at the specified location on the electronic file folder because it was now over a year after the file was originally generated. The FS engineer selected a different file as a baseline that only required him to change five I1 values. During manual entry of those five I1 roll rate filter values, the FS engineer incorrectly entered or missed the exponent for the I1(25) constant. The exponent should have been a 1 instead of a 0, making the entered constant one tenth of the intended value. That value became part of the file that was used to automatically build the Signoff Report, Firing Tables Report, and the Flight Load tape for use at CCAS (Cape Canaveral Air Station, where the Titan was launched). The FS engineer's immediate supervisor did not check the software manual entry. The Flight Load Tape was not used in the LMA Flight Analogous Software Test (FAST) test bed. That I1 file was not sent to Analex-Cleveland for autopilot validation because Analex-Cleveland only performs design validation.

E0-4 On or about February 17, 1999, the FS engineer who developed the Flight Load tape notified the CD engineer responsible for design of the first thirty I1 constants that the tape was

completed and the printout of the constants was ready for inspection. The CD engineer went to the FS offices and looked at the hardcopy listing to perform the check and sign off the I1 constants. The manual and visual check consisted of comparing a list of I1 constants from Appendix C of the Software Constants and Code Words Memo to the paper printout from the Flight Load tape. The formats of the floating-point numbers were different for each list. The CD engineer did not spot the exponent error for I1(25) and signed off that the I1 constants on the Flight Load tape were correct. The CD engineer's immediate supervisor, the lead for the CD section, did not review the Signoff Report or catch the error.

E0-5 The tapes sent to FAST did not contain the Inertial IMS filter constants because FAST did not use the flight values, only a set of generic default values.

E0-6 The flight load tape was sent to LMA engineers at CCAS, Analex-Denver, and Aerospace shortly after signoff on February 17, 1999. Analex-Denver did a range and bit check, and the value for the I1(25) constant was within the range of acceptable values.

E0-7 Analex-Cleveland received the Flight Dynamics and Control Analysis Report (FDACAR) containing the correct value for the roll filter constant. Their function is to validate the autopilot design values provided in the FDACAR. That does not include IV&V of the I1 constants in the flight format. The original design work is correctly represented by the constants in the FDACAR. In other words, the filter constant in question was listed in the FDACAR with its correct value of -1.992476, and not the value on the flight tape (-0.1992476).

E0-8 The LMA FAST lab was designed to test the compatibility and functionality of the Flight Control software and the Honeywell IMS software. The FAST lab used a simulation for the IMS filters built on the original, correctly specified values from the LMA CS engineer. It did contain the actual Flight Control software, but not the IMS filter constants as entered by the software personnel in the generation of the Flight Load tape. With a mix of actual flight software and simulated filters, the I1(25) error was not present and not detected during internal LMA testing before the Titan IV B-32 mishap.

E0-9 Analex-Denver used IMS default values for testing and did not validate the actual I1 constants used in flight. For their verification effort, Analex-Denver performed a range check of the program constants and Class I flight constants. They also verified the format conversions were done correctly. However, the format conversion they performed simply compared the incorrect I1(25) in the firing tables to the incorrect I1(25) after the conversion, and they matched.

E0-10 The incorrect I1 constant was first loaded into the flight hardware at CCAS on 14 April 1999. The same load of flight software and constants is used each time the Centaur is powered up from that point through launch. The LMA Guidance engineer and a LMA Data Station monitor at CCAS each noticed that the roll rate output was very low. They reviewed the results from previously run procedures and correlated the change with the first installation of the actual flight loads on 14 April 1999. Prior to that time, the I1 constants for the previous Titan/Centaur flight (TC-09), which was not a Milstar flight, had been used. The Guidance engineer initially discussed her observations with the LMA (Denver) Product Integrity Engineers (PIEs) for the Inertial Navigation Unit hardware and Inertial Navigation Unit system. The PIEs referred her to the LMA CD lead.

E0-11 On Friday, 23 April, 1999, the LMA Guidance engineer telephoned the LMA CD lead. The CD lead was not in his office so the Guidance engineer left a voice mail stating she noticed a

significant change in roll rate when the latest filter coefficients were entered, and asked for a return call to her or her supervisor. The Guidance engineer left an email for her supervisor at CCAS explaining the situation[11]. Her supervisor was on vacation, and due back in the office Monday morning 26 April 1999, when she was scheduled to work the second shift. The CD lead and the CD engineer who originally specified the filter values listened to the voice mail from the Guidance engineer. They called her supervisor at CCAS who had just returned from vacation. He was initially unable to find the email during their conversation. He said he would call back, so the CD engineer left the CD lead's office. The CD lead subsequently talked to the Guidance supervisor after he found and read the email. The CD lead told the supervisor at CCAS that the filter values had changed in the flight tape originally loaded on 14 April 1999, and the roll rate output should also be expected to change. Both parties believed the difference in roll rates observed were attributable to expected changes with the delivery of the flight tape[12]. LMA (Denver) engineers requested no hardcopy information and did not speak directly with the Guidance engineer or Data Station Monitor.

E0-12 During the day of the launch, when the tower was rolled back, data was collected that identified the pitch and yaw channels were responsive to environmental stresses (i.e., wind) but the roll channel was flat. The 3SLS engineers were present during launch processing and launch countdown to observe the data, but did not recognize the roll rate output as anomalous. An INU PIE noticed the low roll rates and performed a test to see if the gyros were operating correctly, but the test did not detect the problem.

Level 2 Conditions

C1 Even though the Milstar manufacturer decided the roll rate filter was not needed, LMA decided to leave it in place for the first and subsequent Milstar flights for "consistency." (This word is not explained further in the report.) (↓E0-1, E0-2)

C2 The flight software engineer who created the database dealt with over 700 flight constants generated by multiple sources, in differing formats, and at varying times (some with multiple iterations) all of which had to be merged into a single database. Some constant values came from electronic files that could be merged into the database, others from paper memos manually input into the database. Procedures for creating and updating the database were not formally documented and were left to the flight software engineer's discretion. There was also no formal, documented process to check or verify his work. The LMA Software Group personnel who are responsible for creating the database (from which the flight tapes are generated) were not aware that Analex IV&V testing did not use the as-flown (manually input) I1 filter constants in their verification and validation process. (↓E0-3)

C3 The manually input I1 filter rate values were only checked by one individual other than the flight software engineer who actually input the data. The LMA Control Dynamics engineer who designed the I1 rate filter did a manual visual check and signoff of the set of numbers produced by the software engineer. Those numbers were contained in two paper documents with different decimal and exponential formats for the three values cross-checked for each I1

[11] Note the similar implication of informal email reporting of problems with the MCO accident. Email seems to be a problem in allowing easy informal interaction that bypasses the formal problem-reporting and tracking functions and thus is easy to ignore or lose.

[12] Again notice the similarities with the other accidents investigated: In Challenger the engineers spotted the problem but were unable to get appropriate attention, and the MCO navigation operations spotted the MCO problem but also explained it away.

constant. The Control Dynamics engineer did not catch the error. He did not know that the design values were manually input into the database used to build the flight tapes and that the values were never formally tested in any simulation prior to launch. Once the incorrect filter constant went undetected in the Signoff Report, there were no other formal checks in the process to ensure the I1 filter rate values used in flight matched the designed filter. (↓E0-4)

C4 Analex-Cleveland has the responsibility to verify functionality of the design constant and not what is loaded into the Centaur for flight. (↓E0-7)

C5 The LMA Flight Analogous Simulation Test (FAST) facility was used predominantly to test Flight Control software developed by LMA, but the IMS software was provided by Honeywell, who verified and validated it. The FAST lab was originally constructed with the capability to exercise the actual flight values for the I1 roll rate filter constants, but that capability was not widely known by the current FAST software engineers until after the Titan IV B-32 mishap; knowledge of the capability was lost in the corporate consolidation/evolution process. Instead, FAST used a simulated data file and set of default roll rate filter constants. Had they used the actual flight values in their simulations prior to launch, they would have caught the error. (↓E0-8)

C6 The constants verified in the Signoff Report became the baseline for the flight tape and the "truth baseline" provided by LMA to Analex-Denver, per agreement between LMA and Analex. Analex-Denver used the IMS default values for testing, however, for several reasons. First, Analex-Denver, who is responsible for IV&V of the flight software to ensure the autopilot design is properly implemented in the software, was not aware that the I1 filter rate values provided to them by LMA as the truth baseline originated from a manual input and might not be the same as those in the autopilot design IV&V'd by Analex-Cleveland. Analex-Denver did not understand the generation or internal verification process for all the constants in the truth baseline nor did they know there was only one person checking them in the LMA verification process. In addition, the process did not require Analex-Denver to check the accuracy of the numbers in the truth baseline, only to do a range check and a bit-to-bit comparison against the firing tables that also contained the incorrect constant. Consequently, they did not verify that the designed I1 filter constants were the ones actually used on the flight tape.[13] (↓E0-9)

C7 The validation testing performed by Analex-Denver did not use the actual I1 constants contained on the flight tape. A set of generic or default I1 constants were used in their simulations because they believed the actual I1s could not be adequately validated in their rigid body simulations, i.e., the rigid body simulation of the vehicle would not exercise the filters sufficiently[14]. They found out after the mission failure that had they used the actual I1 constants in their simulation, they would have found the order of magnitude error. (↓E0-9)

C8 The Defense Contract Management Command software surveillance personnel were not aware that the I1 filter rate constants contained in the flight software were generated by a manual input and were never tested by LMA in their preflight FAST simulation or IV&V'd by Analex. (↓all)

[13]Note that the detail included in this report about why the error was not detected before launch assists in understanding how competent and dedicated engineers could have made it. If the MCO report had provided more information about the development process of the MCO software, the mistake might also seem less ridiculous.

[14]Note that almost identical words were used in the Ariane 5 report.

C9 The LMA Software Quality Assurance staff, not understanding the manual input and single check for the I1 constants, left the I1 constant-checking process to the LMA Control Dynamics and flight software engineers. (↓all)

C10 The Program Office did have support from Aerospace to monitor the software development and test process, but that support had been cut by over 50 percent since 1994. (↓all)

C11 The 3SLS has greatly reduced the number of engineers working launch operations. There was no master surveillance plan in place to define tasks for the remaining staff—the formal insight plan was still in draft—so they used their best engineering judgement to determine which tasks they should perform, which tasks to monitor, and how closely to analyze the data from each task. They had no documented requirement or procedures to review the data and no reference with which to compare. (↓E0-10, E0-11, E0-12)

C12 No formal processes existed to check validity of the I1 filter constants or to monitor attitude rates once the flight tape was actually loaded into the INU at CCAS prior to launch. Approximately one week before the Titan IV B-32 launch, LMA personnel at CCAS observed much lower roll rate filter values than they expected. When they could not explain the differences at their level, they raised their concerns to Denver LMA guidance product Integrity Engineers (PIEs) who were now at CCAS. The on-site PIEs could not explain the differences either, so they directed the CCAS personnel to call the control dynamics design engineers in Denver. Due to poor and indirect communications (voice mail and email) and lack of understanding of each other's responsibilities and processes, their concerns were not adequately addressed.

The LMA personnel at Denver never asked to see the actual data observed at CCAS, nor did they talk to the engineer and data analyst at CCAS who questioned the low filter rates. There was confusion and uncertainty from the time the issue was raised until it was "resolved" as to how it should be reported, analyzed, documented, and tracked since it was a "concern" and not a "deviation." (↓E0-10, E0-11)

C13 No one other than the control dynamics engineers who designed the I1 roll rate filter constants understood their use or the impact of filtering the roll rate to zero. At this point in the prelaunch process, no one was required to monitor or analyze the data nor was anyone responsible to perform a check to see if the attitude filters were properly sensing the earth's rotation rate. Consequently, no one was able to question the anomalous rate data recorded or correlate it to the low roll rates observed about a week prior to launch. A simple root sum square plot of the sensed attitude rates would have identified the problem. If someone who understood the I1 roll rate filter design had been monitoring the rate data, the error could have been detected. (↓E0-12)

C14 The INU PIE noticed the low roll rates on the day of launch and performed a rate check to see if the gyros were operating properly. Unfortunately, the programmed rate check uses a default set of I1 constants to filter the measured rates and consequently verified that the gyros were sensing the earth rate correctly. (↓E0-12)

Level 3 Systemic Factors

This report, in comparison with many of the others, did an excellent job in identifying the systemic factors involved in the accident.

I. Flaws in the Safety Culture

S1 *Assuming software risk decreases over time*

The Titan Program Office had no permanently assigned civil service or military personnel nor full-time support to work the Titan/Centaur software. They decided that because the Titan/Centaur software was "mature, stable, and had not experienced problems in the past"[15], they could best use their resources to address hardware issues. (↓C10)

S2 *Ignoring warning signs*

Engineers noticed problems with the software after it was delivered to the launch site, but nobody seemed to take them seriously. Perhaps the urgency of the problems discovered and reported by the LMA Guidance Engineer in April was not appreciated because everyone thought that the system had gone through a complete IV&V. (↓C12)

S3 *Complacency about software: Underestimating and misunderstanding software-related risks*

There seemed to be almost no checking of the correctness of the software after the standard testing during development. For example, on the day of the launch, the attitude rates for the vehicle on the launch pad were not properly sensing the earth's rotation rate but no one had the responsibility to specifically monitor that rate data or to perform a check to see if the attitude filters were properly sensing the earth's rotation rate. In fact, there were no formal processes to check the validity of the filter constants or to monitor attitude rates once the flight tape was actually loaded into the INU at the launch site. Hardware failures are usually checked up to launch time, but often testing is assumed to have removed all software errors and further checks are not needed. Note that even right before launch the programmed rate check used a default set of I1 constants to filter the measured rate instead of the actual constants.

There is nothing in the report about why ground control or the possibility for intervention during flight was not provided. Most likely it was not feasible. Another possibility is that nobody thought it would be necessary because the software would not contain errors or operator error was more likely than software error. It would have been interesting if the report had included a statement about this decision. (↓C13, C14)

S4 *Flawed resolution of conflicting goals*

The Program Office support to monitor the software development and test process had been cut by over 50 percent since 1994 and the 3SLS had greatly reduced the number of engineers working launch operations. Although budget decisions are always difficult when resources are reduced, quality assurance, system engineering, and operations are often assumed to be the least critical parts of the process and thus are the first to be cut. (↓C10, C11)

II. Ineffective Organizational Structure and Communication Deficiencies

S5 *Diffusion of responsibility and authority*

The Accident Investigation Board could not identify a single process owner responsible for understanding, designing, documenting, or controlling configuration and ensuring proper execution of the process. Responsibility seemed to be diffused among the various partners, without complete coverage. For example, Analex-Cleveland had responsibility to verify the functionality of the design constant but not the actual constant loaded into the Centaur for

[15] Here we go again

flight. Nobody seemed to be responsible for checking the load tape once it was installed at the launch site. The 3SLS had greatly reduced the number of engineers working launch operations. There was no master surveillance plan in place to define tasks for the remaining staff, who used their best engineering judgment to determine what tasks they should perform. This approach, however, did not ensure that anyone was responsible for specific tasks.

The report also mentions that the Air Force Space and Missile System Center Launch Directorate and the 3SLS had undergone personnel reductions and were transitioning from a task oversight to a "process insight" role[16]. A software quality assurance function existed and operated at both LMA and DCMC but operated without a detailed understanding of the overall process or program. In addition, transition from an oversight role to an insight role was not implemented properly by either agency because of that lack of understanding. (↓C2, C4, C6, C11)

S6 *Organizational and communication problems between the partners in the program*

The fragmentation/stovepiping in the flight software development process, coupled with the lack of an overall defined process, resulted in poor and inadequate communication and interfacing among the many partners and subprocesses. The accident report suggests that many of the various partners were confused about what the other groups were doing. For example, the LMA Software Group personnel who create the database (from which the flight tapes are generated) were not aware that Analex Independent V&V testing did not use the as-flown (manually input) I1 rate filter constants in their verification or validation process. Analex-Denver, who were responsible for the independent V&V of the flight software, were not aware that the I1 filter rate values provided to them by LMA as the "truth baseline" originated from a manual input and might not be the same as those in the software subjected to independent V&V by Analex-Cleveland. The Defense Contract Management Command software surveillance personnel were not aware that the I1 filter rate constants contained in the flight software were generated by a manual input and were never tested by LMA in their preflight FAST simulation or subjected to independent V&V by Analex-Cleveland. The LMA Software Quality Assurance staff, not understanding the manual input and single check for the I1 constants, left the I1 constant-checking process to the LMA Control Dynamics and Flight Software engineers.

The INU, for which all the flight software is developed, consists of two major software components developed by different companies. The LMA developed the Flight Control System software and is responsible for overall INU testing. Honeywell developed the IMS and is partially responsible for its software development and testing. The I1 constants are processed by the IMS, but are designed and tested by LMA. The focus of the LMA flight software process is on FCS versus IMS software. Key players in the flight software development, test, and mission/quality process (including LMA control dynamics engineers, flight software engineers, product integrity engineers, and software quality assurance personnel; Analex-Denver personnel; and DCMC personnel) focused their efforts on FCS operation and had little knowledge of IMS operations. (↓C5, C6, C7, C8, C9)

S7 *Limited communication channels and poor information flow*

Again, informal reporting and use of email seems to be implicated in critical information not getting to those who can use it. Note that tests right before launch detected the zero roll rate

[16]The Mars '98 program also was transitioning to "process insight."

but there was no communication channel established for getting that information to those who could understand it. (↓C12, C14)

III. Ineffective or Inadequate Technical Activities

S8 *Flawed review process*

The QA program was obviously inadequate. This accident is indicative of general problems with QA as generally practiced and why it is often ineffective. The LMA Quality Assurance Plan is a top level document that focuses on verification of process completion, not on how the processes are executed or implemented. Its basis is the original General Dynamics Quality Assurance Plan with recent updates to ensure compliance with ISO 9001. Thus, LMA Software Quality Assurance staff only verified that the signoff report containing the constants had all the proper signatures; they left the I1 constant generation and verification process to the FS and CD engineers. Software Quality Assurance involvement was limited to verification of software checksums and placing quality assurance stamps on software products that were produced.

At the same time, Analex developed (with LMA and government approval) an IV&V program that did not verify or validate the I1 filter rate constants actually used in flight.

Although a 3SLS engineer did see the roll rate data at the time of the tower roll back, he was not able to identify the problem with the low roll rate. He had no documented requirement or procedures to review the data, and no reference with which to compare. (↓C3, C4, C6, C11)

S9 *Inadequate specifications*

Prior to the Titan IV B-32 mission failure, there was no formal documentation of the overall process flow. The Centaur software process was developed early in the Titan/Centaur program: Many of the individuals who designed the original process are no longer involved in it due to corporate mergers and restructuring (e.g., Lockheed, Martin Marietta, General Dynamics) and the maturation and completion of the Titan IV design and development. Much of the system and process history and design rationale was lost with their departure.

No one other than the control dynamics engineers who designed the I1 roll rate filter constants understood their use or the impact of filtering the roll rate to zero. So when discrepancies were discovered right before launch, nobody understood them. If someone who understood the I1 roll rate filter had been monitoring the rate data, the error could have been detected and the mission failure averted. Good specifications that include design rationale are critical for long-lived systems. (↓C5)

S10 *Inadequate system engineering*

The SMC Launch Programs Directorate essentially had no personnel assigned to monitor or provide insight into the generation and verification of the software development process. The process used to develop and test the I1 constants actually used in the flight software was not well defined, documented, nor completely understood by any of the multiple players involved in that process. Procedures for creating and updating the database were not formally documented and were left to the flight software engineer's discretion. The root problem is probably not the lack of documentation itself but the lack of anyone being in charge of the entire process.

There were multiple players who performed portions of the process, but they only completely understood their specific part. For example, the LMA Control Dynamics personnel who design the I1 rate filter constants did not know their design values were manually input into the database used to build the flight tapes and that the values were never formally tested in any simulation prior to launch. While it is not possible for each engineer to be responsible for understanding and overseeing the entire process, this responsibility is exactly the role system engineering is supposed to play. The Accident Investigation Board could not identify a single process owner responsible for understanding, designing, documenting, controlling configuration, and ensuring proper execution of the overall process. (↓C2, C6, C11)

S11 *Inadequate system safety engineering*

System safety and hazard analysis are never mentioned in the report. It does say, however, that the QA engineers considered mostly those problems that had happened before.

> Their risk analysis was not based on determining steps critical to mission success, but on how often problems previously surfaced in particular areas on past launches. They determined software constant generation was low risk because there had not been previous problems in that area. They only verified that the signoff report containing the constants had all the proper signatures.

Considering only the causes of past accidents is not going to be effective for software problems or when new technology is introduced into a system. Computers are, in fact, introduced in order to make radical changes in functionality and design and the "fly-fix-fly" approach to safety will no longer be effective. Proper hazard analyses examining all the ways the system components can contribute to an accident need to be performed.

Note that each component of the process was done correctly except for the person who originally input the constants and the risk of improper manual entry of long digit strings is well known. The problem was that the overall process together did not contain the proper checks and balances.

S12 *Unnecessary complexity and software functions*

Once again we find software functionality that was not needed left in code. In this case, it was first thought that the filter was needed, but after discovering it was not necessary it was left in for the first and subsequent Milstar flights for "consistency." As with the Ariane 5 accident report, this term is not defined and could be interpreted in many ways. (↓C1)

S13 *The test and simulation environment did not adequately reflect the operational environment*

Again, the spacecraft was not tested as it would be flown nor flown as it was tested. The tests (even simulator tests) used default values and simulated values. Like Ariane, the engineers incorrectly thought the rigid-body simulation of the vehicle would not exercise the filters sufficiently.

The Flight Analogous Simulation Test (FAST) performed by LMA used a 300 Hertz filter simulation data file and not the flight tape values. The FAST test bed was designed so it *could* use the actual I1 roll rate filter constants; however, recognition of this capability was lost in the corporate consolidation/evolution process. As a result, the current software engineers performed FAST testing using a set of default roll rate filter constants. Later it was determined that had they used the actual flight values in their simulations prior to launch, they would have caught the error. (↓C5, C7)

S14 *Deficiencies in information collection and use*

The use of voice mail and email implies that either there was no formal anomaly reporting and tracking system or it was not known or used by the process participants.

The report says that there was confusion and uncertainty from the time the issue was first raised by the LMA engineer via email and voice mail until it was "resolved" as to how it should be reported, analyzed, documented, and tracked since it was a "concern" and not a "deviation." There is no explanation of these terms nor description of any formal reporting system. (↓C12)

S15 *Operational personnel do not understand the automation.*

The INU PIE did notice the low roll rates and performed a rate check to see if the gyros were operating properly. Unfortunately, the programmed rate check uses a default set of I1 constants to filter the measured rates, and consequently verified that the gyros were sensing the earth rate correctly. If the attitude rates were monitored at that time, or summed and plotted to ensure they were properly sensing the earth's gravitational rate, the roll rate problem could have been identified and the mission failure averted. (↓C13, C14)

3.6 Warsaw

3.6.1 Background

On September 14, 1993, a Lufthansa Airbus A320 landing at Warsaw airport in a thunderstorm overran the runway and collided with an earth bank located at the runway end. As a result of the crash, one crew member and one of the passengers were killed. The aircraft sustained damage caused by fire.

To understand the accident, some information is needed about the braking system on the A320. The braking system consists of three components:

1. **Ground Spoilers:** If the ground spoilers are selected to be ON, they will extend given the following *on ground* conditions are met:

   either oleo struts (shock absorbers) are compressed at both main landing gears (the minimum load to compress one shock absorber being 6300 kg),

   or wheel speed is above 72 kts at both main landing gears.

2. **Engine reversers:** If the engine reversers are selected ON, they will deploy if the following *on ground* condition is met: shock absorbers are compressed at both main landing gears.

3. **Wheel brakes:** The above conditions (wheel speed above 72 kts and both shock absorbers compressed) are not used to activate the brakes. With the primary mode of the braking system, the brakes may be used as soon as wheel speed at both landing gears is above a reference speed. With the alternate mode of the braking system, the brakes may be used as soon as the A/SKID-NOSE WHEEL STEERING switch has been selected to the OFF position by the crew.

In an emergency, the crew is *not* able to override the lockout and to operate the ground spoilers and engine thrust reversers manually.

3.6.2 Chain of Events

E1 Flight DLH 2904 from Frankfurt to Warsaw progressed normally until the Warsaw tower warned the crew that windshear existed on the approach to their assigned runway (Runway 11).

E2 The wind shifted in direction from a side wind to a tail wind.

E3 The plane landed beyond the normal touchdown point and going faster than normal.

E4 Upon landing, none of the braking systems (air brakes, reverse thrust, and wheel brakes) functioned.

E5 Nine seconds after touchdown, the spoilers and reverse thrust deployed.

E6 13 seconds after touchdown, the wheel brakes became effective. By this time, the aircraft was too far down the runway to stop before the runway end.

E7 The aircraft ran off the end of the runway, collided with an earth bank, and started to burn.

E8 The passengers were all evacuated safely except for one person who was asphyxiated from fire smoke. One pilot died when the aircraft hit the bank.

E9 Fire-fighting vehicles reached the crash site about three minutes after the emergency call and extinguished the external fire.

3.6.3 Identified Cause of Accident

The report identified the cause of the accident as incorrect decisions and actions of the flight crew taken when the information about windshear at the approach to the runway was received. Windshear was produced by a front just passing the airport; the front was accompanied by intensive variation of wind parameters as well as by heavy rain on the airport itself.

The actions of the flight crew were also affected by design features of the aircraft, which limited the feasibility of applying available braking systems as well as by insufficient information in the aircraft operations manual (AOM) relating to the increase of the landing distance.

3.6.4 Hierarchical Model

Level 1 Conditions

C1 At the moment the aircraft landed, a storm front, accompanied by heavy rain and windshear, was passing through the airport (↓E1).

C2 The crew heard three separate warnings about windshear in the runway approach area—two from aircraft landing ahead of them and one from the tower. (↓E3)

C3 Airspeed was increased by 20 knots after the windshear warning, as recommended in the A320 AOM (Aircraft Operations Manual) for windshear conditions. The Lufthansa Flight Crew Manual gives the procedures for the case of windshear as: "If the disposed landing distance allows, the speed may be increased maximum by 20 kts. This increased speed should be maintained until flare-out." The Airbus Flight Crew Operating Manual allows an increase in approach speed of 15 kts. (↓E3)

C4 In the course of the approach, the crew did not notice on the weather radar screen the atmospheric front between their position and the airport. The weather radar was switched off once the crew got visual contact with the airport. Once they turned the radar off, it could not help them to evaluate the situation properly by showing the passing cold front. The front was noticed by the aircraft that landed right before them on their weather radar screen. (↓E3)

C5 The wind shifted drastically right before the aircraft landed, from a side wind to a tail wind, but the pilots were not informed of the change or that the tower weather report might be inaccurate. The tower personnel did not have on hand the certified data on current direction and velocity of the surface wind. The data received from the meteorological services was delayed and, in addition, wind information was based on average values over a two hour period. (↓E3)

C6 The tail wind should have been visible to the crew by the difference between Air Speed and Ground Speed in the course of the approach and the wind vector on the EFIS (Electronic Flight Information System) during final approach. There was no reaction of the crew although the change in direction and velocity of wind was commented upon in the crew's conversation. (↓E3)

C7 The discrepancy between the EFIS wind display and the air traffic services data was apparently also not considered or not noticed. (↓E3)

C8 Because of wind shear, the pilot not only increased speed, but also increased thrust down to relatively very low altitude. This resulted in the aircraft being significantly above the ILS glide path and in an significant extension of the flare out phase (the period of time between the aircraft passing the runway threshold to the first recorded contact of the aircraft with the runway surface). (↓E3)

C9 The pilot at the controls (PF) decided to continue to land in spite of excessive tailwind, excessive ground speed, and a touchdown point considerably past the runway threshold while abandonment of landing and GO AROUND was justified and still feasible. The pilot in command approved and did not warn him that the tailwind (displayed on the EFIS) considerably exceeded the operational limit of 10 knots documented in the AOM (Aircraft Operator's Manual). (↓E3)

C10 Both pilots were highly experienced, and the Captain actually had slightly fewer hours total flight time than the F/O. This flight was the last stage of a checkout flight for the pilot who, having not flown for 90 days, was required according to the rules to be checked. In the light of recommendations of the AOM, the cooperation between pilots was incorrect (probably as a result of the non-typical crew composition). The pilot performing the check allowed too wide a margin of independence to the checked pilot. (↓E3, E8)

C11 The crew did not analyze whether the runway would be long enough for landing at the increased approach speed. The operator's flightcrew manual contains a notice that excessive touchdown speed may extend the rollout distance by about 25%, but in the tables of required landing distances, there is no explanation of how to determine the real landing (and rollout) distance when landing with the increased speed, even in the case of the 20 kts recommended in case of windshear. Note that the AOM manual specifies how to calculate corrections to these distances for "much less important factors" (according to the accident report), such as difference in pressure altitude, aircraft icing, or automatic landing. (↓E3)

C12 FULL flaps configuration was set. On this aircraft, it disables the braking system until touchdown is recorded. (↓E4)

C13 Upon landing, none of the braking systems (air brakes, reverse thrust, and wheel brakes) functioned. The right banked turn approach resulted in a casual contact with the runway, which was strong enough to compress one oleo switch as much as was needed for the oleo switch to come on, giving the crew the false impression that both gears had contact with the runway and that the aircraft had landed. They were unaware they had landed on only one wheel. (↓E4)

C14 The Airbus A320 automatic systems, dependent on compression of the oleo struts, armed all three braking systems only at the moment of recorded contact of the left main landing gear assembly with the runway. The ground spoilers and engine reversers will not actuate unless either the shock absorbers are compressed at both main landing gears (the minimum load to compress one shock absorber is 6300 kg) or the wheel speed is above 72 knots at both main landing gears. In an emergency (as noted earlier), the crew is unable to override the braking system lockout and to operate the ground spoilers and engine thrust reversers. (↓E4)

C15 At the initial casual contact, the airspeed was too high to allow compression of the oleo struts (which would activate the ground spoilers). Compression of the oleo struts occurred once the airspeed had decreased to 136.0 knots. The spoilers were deployed to FULL and the reverser systems went into operation. (↓E4, E5)

C16 The wheel brakes, which are dependent on the rotation gain to circumferential speed of 72 knots, did not began to operate until about 4 seconds later. An uneven layer of water up to several millimeters covered the surface of the runway, which along with the high touchdown speed and considerable wear of three of the four tires on the main landing gear, led to hydroplaning (an extreme decrease in friction) and slippage. The hydroplaning delayed activation of the wheel brakes and considerably degraded the braking effectiveness. (↓E4, E6)

C17 By the time the braking systems were fully deployed, the aircraft was already too far along the runway to stop by the runway end, even with fully functioning braking systems. (↓E7)

C18 The embankment at the end of Runway 11 (100 meters from its end and on its centerline) was not described in the AIP (airport information). (↓E7)

C19 Seeing the approaching end of runway and the obstacle behind it, the pilot struggled to deviate the aircraft to the right. The aircraft began to turn right, but its center of gravity movement path did not bend. The aircraft rolled over the end of the runway at a speed of 72 knots and after passing through the next 90 meters, collided with the embankment and started to burn. (↓E7)

C20 The cabin crew responded superbly and all but one passenger got out of the plane safely. That passenger was unconscious in the front corner of the airplane and was unnoticed in the evacuation as the cabin filled with smoke from the fire. Passengers left the aircraft by themselves, assisting each other. Two of the four main cabin attendants, who did not suffer major injuries in the crash, also helped wounded passengers to leave the fire-endangered area. (↓E8)

C21 Because of the character of the fire (aviation fuel and oxygen escaping from onboard systems) and violent fire expansion inside the aircraft, extinguishing the fire was possible only after

an explosion had destroyed part of the cabin roof and fire extinguishing agents could then be delivered through the resulting hole. (↓E8)

Level 3 Systemic Factors

The lack of information and investigation beyond the proximate events makes it difficult to speculate about broad systemic factors (and to learn from the accident). This report, like many for civil aviation accidents, seems more focused on determining who was at fault (i.e., affixing and apportioning blame) than on understanding why the accident occurred. In civil aviation, the pilots are assigned responsibility for safety of the aircraft and the passengers, and therefore, they are by definition responsible if they contribute at all to the accident or could in any way have prevented it or mitigated the consequences. Although we were aware of this standard practice, the participants in the MIT seminar were surprised at the narrow focus of some of the aircraft accident reports we studied.

The deficiencies of the reports became particularly clear in attempting to account for the Level 2 conditions by Level 3 systemic factors. Not only could most of them not be explained, but there was inadequate information provided even to speculate about this. As a result, this section describes the systemic factors that can be reasonably identified and then notes the questions that need to be answered to thoroughly explain this accident and why it occurred.

I. Flaws in the Safety Culture

S1 *Overconfidence in automation*

The Airbus philosophy has been to give final authority to the computer when there is a discrepancy between it and the pilot, for example, the decision not to allow the pilots to override the interlocks in the event of an emergency and to operate the ground spoilers and engine thrust reversers. Although there may be good reasons for making this decision, we are just not at the point where we can build software to account for every possible condition that can arise, and trusting it above human intelligence and flexibility may be a mistake (↓C13, C14, C15, C16).

II. Organizational and communication deficiencies

S2 *Weather-reporting system*

The Polish system for collecting and distributing meteorological information is obviously inadequate. For example, weather is based on information averaged over two hours and therefore does not reflect sudden changes (↓C5).

S3 *Pilot communication and specified procedures*

Perhaps because of the atypical cockpit crew composition, the pilot in command seems to have been too accepting of the pilot in control's decisions. General problems in cockpit crew management and interaction are well-documented and too complex and extensive to cover adequately here (↓C6, C7, C9, C10).

III. Ineffective or Inadequate Technical Activities

S4 *Conflicting and insufficient documentation*

The Aircraft Operations Manual contained conflicting information, such as the appropriate speed to use to counter windshear. In addition, some critical information is missing, such

as how to compute increased landing distance under various landing speeds (including those recommended in the AOM, such as for windshear) (↓C3, C11, C18).

S5 *Inadequate safety engineering and risk control*

Some of the technical inadequacies in the system design arise from the lack of confidence in the human and the overconfidence in the automation discussed above. Even if automation is considered to be more reliable than humans, it may be a mistake not to allow some flexibility in the system for emergencies. For example, the Airbus automation does not allow use of more than 71% of the thrust reversion system even in an emergency. The report does not explain the reason behind this limit, but it most likely is meant to reduce maintenance and stress on the system. However, pilots should probably be given discretion in an emergency situation to stress the system in order to save the occupants and prevent the destruction of the aircraft. Other limitations on the ability of the pilots to override physical interlocks were also involved in this accident (↓C12, C13, C14, C15, C16, C17).

The findings and recommendations in the report do not go beyond those stated above. But in attempting to account for each of the Level 2 conditions, many questions were raised, for example:

- Why did the pilots not notice the atmospheric front on the weather radar screen? Could this be related to the design of the display, pilot workload, or defined pilot procedures? (↓C4)
- Why did the pilots switch the weather radar off once they got visual contact with the airport? Is this common practice? If not, what might be the reasons for highly trained and skilled pilots to do it? (↓C4)
- Why did the pilots not react to the change in direction and velocity of the wind? They knew about it because the report notes that the wind change arose in their conversation. (↓C6)
- Why did the pilots not consider or notice the discrepancy between the EFIS wind display and the air traffic services data? Poor design? Poor procedures? High workload? (↓C7)
- What factors might account for the fact that the pilots did not decide to go around but continued to land when it should have been apparent that a go-around was safer? Why would highly experienced pilots make such a poor decision? Poor training? Complacency? Cognitive fixation? (↓C9)
- Was there a flaw in the instrumentation design that allowed the pilots to think they had landed (the oleo switch came on) when they had not? (↓C13)
- Were the software requirements incomplete in the factors used to activate the braking systems? If so, does this simply reflect the difficulty of thinking of all possible conditions and is therefore a system engineering deficiency? Or is it the result of a tradeoff between the requirements for landing and the danger of inadvertent activation during flight? (↓C13, C14, C15, C16, C17)
- Why were the pilots not warned about water on the runway? (↓C16)
- Were the operational procedures allowing landing with several inches of water on the runway unsafe? (↓C16)
- Why were the tires worn? Did the pilots know this (and therefore know that additional landing distance and the potential for hydroplaning existed)? (↓C16)

- Is having three of the four tires considerably worn acceptable procedure? Was it a maintenance failure? Was it airline policy in order to reduce maintenance expenses? (↓C16)
- Why was the weather report delayed? (↓C5)
- Was it a mistake not to provide alerting functions to the pilot, for example, that they were in a landing configuration but were exceeding the limits specified in the AOM? (↓C11)
- Why was an embankment put at the end of the runway? How was this decision made? (↓C18, C19)

3.7 Nagoya

3.7.1 Background

China Airlines Airbus A300 Flight 140 took off from Taipei International Airport for Nagoya Japan on April 26, 1994. It was an evening departure with an estimated time en route of 2 hours and 18 minutes. While approaching Nagoya for landing, the aircraft crashed into the landing zone close to the E1 taxiway of the airport. The aircraft was on manual control by the First Officer at the time of the accident, there was good communication, and all navigational aids were operational. Of the 271 people onboard, only seven passengers survived, all of whom were seriously injured. The aircraft ignited and was destroyed.

The A300 has an advanced computer-based flight management system (FMS). One part of the FMS design is critical in understanding this accident. When the autopilot is engaged in LAND or GO AROUND modes, movements of the elevators by the autopilot can be overridden by the pilot pushing or pulling the control wheel. Even when the elevator movement by the autopilot is overridden, however, the autopilot autotrim orders are *not* canceled, and the autopilot continues to move the trimmable horizontal stabilizer so as to maintain the aircraft on the scheduled flight path. Under these conditions, the aircraft will eventually go into an out-of-trim condition. A *caution* is provided in the Flight Crew Operating Manual with regard to this hazardous situation.

3.7.2 Chain of Events

E1 After the aircraft took off from Taipei International Airport, at 0854 when the aircraft had passed 1000 ft pressure altitude, autopilot number 2 was engaged during climb, cruise, and descent.

E2 At 1107:14, the aircraft was cleared for ILS approach to Runway 34 and was instructed to contact Nagoya Tower.

E3 At 1107:22, when the aircraft was in the initial phase of approach to Nagoya airport, autopilot number 1 was also engaged.

E4 At 1111:36, both autopilot number 1 and number 2 were disengaged by the first officer. Under manual control, the aircraft continued normal ILS approach.

E5 At 1112:19, the aircraft passed the outer marker and at 1113:39, the aircraft received landing clearance

E6 At 1114:05, while crossing 1070 pressure ft, the first officer inadvertently triggered the GO lever. As a result, the aircraft changed into GO AROUND mode, leading to an increase in

thrust. Both pilots noticed. The captain cautioned the first officer that he had triggered the GO lever and instructed him to "disengage it".

E7 The aircraft leveled off for about 15 seconds at approximately 1040 ft altitude at a point some 5.5. km from the runway.

E8 The captain instructed the first officer to correct the descent path, which had become too high. The first officer acknowledged this command and applied nose down elevator input to adjust the descent path. The aircraft gradually regained its normal glide path. During this period, the captain cautioned the first officer twice that the aircraft was still in GO AROUND mode.

E9 At 1114:18, both autopilots number 1 and number 2 were engaged almost simultaneously when the aircraft was flying at approximately 1040 ft pressure altitude, a position 1.2 dots above the glide slope. Both autopilots were used for the next 30 seconds. There is no record in the CVR (cockpit voice recorder) of either of the crew expressing their intention or calling out to use the autopilot. The GO AROUND mode was still engaged, as were the autopilots.

E10 The first officer continued pushing the control wheel in accordance with the captain's instructions ("push more, push more, push more"), despite its strong resistive force, in order to continue the approach. Accordingly, the elevator moved in the nose-down direction. The THS gradually moved from 5.30 to 12.30, which is close to maximum nose-up limit, to resist the first officer's nose-down input. The conflict between the TRS and the elevator led to an abnormal out-of-trim condition.

E11 At 1114:48, both autopilots were disengaged.

E12 At 1115:02, when the aircraft was passing 510 ft pressure altitude (at a point 1.8 km from the runway), the elevator was close to its nose-down limit. The captain, who had been informed by the first officer that the THR had been latched, told the first officer that he would take over the controls. Around this time, the THR levers had moved forward greatly. Immediately afterwards, however, the THR levers were retarded. In addition, the elevator was moved close to its nose-down limit when the captain took the controls. The AOA (angle of attack) increased and the Alpha Floor function was activated automatically, which increased the thrust to avoid stall. The increased thrust, however, created a large pitch-up moment due to the abnormal out-of-trim situation.

E13 At 1115:11, immediately after the captain called out "Go lever", the THR levers were moved forward greatly once again. The aircraft therefore began to climb steeply. The first officer reported to Nagoya Tower that the aircraft would go around, and Nagoya Tower acknowledged this.

E14 The aircraft started climbing steeply, with a high pitch angle attitude. During this time, the THS decreased from 12.30 to 7.40, and SLATS/FLAPS were retracted.

E15 At 1115:17, the GPWS (Ground Proximity Warning System) warned "glide slope" once, and at 1115:25, the stall warning sounded for approximately 2 seconds.

E16 At 1115:31, after reaching about 1730 ft pressure altitude, the aircraft lowered its nose and began to dive.

E17 At 1115:37, the GPWS warned "Terrain, Terrain" once, and the stall warning sounded from 1115:40 until the time of the crash.

E18 At 1115:45, the aircraft crashed into the landing zone, approximately 110 meters east-northeast of the center of the Runway 34 end of Nagoya Airport.

3.7.3 Identified Cause of Accident

The Aircraft Accident Investigation Committee (AAIC) identified the following cause of the accident:

> While the aircraft was making an ILS approach to Runway 34 at Nagoya Airport, under manual control by the F/O, the F/O inadvertently activated the GO lever, which changed the FD (Flight Director) to GO AROUND mode and caused a thrust increase. This made the aircraft deviate from its normal glide path.
>
> The autopilots were subsequently engaged, with GO AROUND mode still engag ed. Under these conditions, the F/O continued pushing the control wheel in accordance with the captain's instructions. As a result of this, the THS (Trimmable Horizontal Stabilizer) moved to its full nose-up position and caused an abnormal out-of-trim situation.
>
> The crew continued the approach, unaware of the abnormal situation. The AOA (angle of attack) increased, the Alpha Floor function was activated, and the pitch angle increased. It is considered that, at this time, the captain (who had now taken the controls), judged that landing would be difficult and opted for go-around. The aircraft began to climb steeply with a high pitch angle attitude. The captain and the first officer (F/O) did not carry out an effective recovery operation, and the aircraft stalled and crashed.

In addition, the investigation determined that the following factors, as a chain or a combination thereof, caused the accident:

1. The F/O inadvertently triggered the GO lever. The design of the GO lever contributed to this action: Normal operation of the thrust lever creates the possibility of an inadvertent triggering of the GO lever.

2. The crew engaged the autopilots while GO AROUND mode was still engaged, and continued the approach.

3. The F/O continued pushing the control wheel in accordance with the captain's instructions, despite its strong resistive force, in order to continue the approach.

4. The movement of the trimmable horizontal stabilizer conflicted with that of the elevators, causing an abnormal out-of-trim situation.

5. There was no warning and recognition function to alert the crew directly and actively to the onset of the abnormal out-of-trim condition.

6. The captain and F/O did not sufficiently understand the FD (Flight Director) mode change and the autopilot override function. It is considered that unclear descriptions of the AFS (Automatic Flight System) in the FCOM (Flight Crew Operating Manual) prepared by the aircraft manufacturer contributed to this.

7. The captain's judgment of the flight situation while continuing approach was inadequate, control take-over was delayed, and appropriate actions were not taken.

8. The Alpha-Floor function was activated, which was incompatible with the abnormal out-of-trim situation, and generated a large pitch-up moment. This narrowed the range of selection for recovery operations and reduced the time allowance for such operations.

9. The captain's and F/O's awareness of the flight conditions, after the captain took over the controls and during their recovery operation, was inadequate.

10. Crew coordination between the captain and the F/O was inadequate.

11. The modification prescribed in Service Bulletin SB A300-22-602 1 had not been incorporated into the aircraft.

12. The aircraft manufacturer did not categorize the SB A300-22-602 1 as *Mandatory*, which would have given it the highest priority.

3.7.4 Hierarchical Model

The chain of events presented in the report is very detailed, as well as the chain provided as the cause (although some of the events in this chain are not really events). As was found to be common in almost all the reports (see the section on the Titan for a comparison), the chain only included proximate events that occurred on the day of the accident. The report does not delve into events beyond the proximate ones, but one event does have to be added to complete the mapping of events and conditions presented in the report:

E0-1 Because of previous incidents, a change to the flight control computer was issued by Airbus to add an autopilot disengagement function that is activated by applying a force on the control wheel in GO AROUND mode above 400 feet radio altitude. This modification (Service Bulletin SB A300-22-602 1) had not been incorporated into the aircraft involved in the accident.

Level 2 Conditions

C1 The aircraft manufacturer categorized the Service Bulletin to add the autopilot disengagement function as *Recommended* instead of *Mandatory*, which would have given it the highest priority. China Airlines decided to delay this flight control computer (FCC) modification until the FCC needed repair because the installation of the change required sending the FCC to the Sextant Singapore maintenance facility.[17] (↓E0-1)

C2 The airworthiness directive of the nation of design and manufacture (France) did not issue promptly an airworthiness directive pertaining to the Service Bulletin. (↓E0-1)

C3 The GO lever was designed such that inadvertent engagement was possible during normal operation of the thrust lever. (↓E6)

C4 The autopilot is not completely disconnected when the pilot overrides it in LAND and GO AROUND modes (autopilot override does not cancel the autopilot autotrim orders). This can result in an out-of-trim situation and create difficulties in controlling the aircraft. (↓E10, E12)

[17] This information was not in the accident report but was obtained independently.

C5 The movement of the THS conflicted with that of the elevators, causing an abnormal out-of-trim situation. (↓E10)

C6 The captain and first officer did not sufficiently understand the FD (Flight Director) mode change and the autopilot override function. Unclear descriptions of the AFS (Automatic Flight System) in the FCOM (Flight Crew Operating Manual) prepared by the aircraft manufacturer may have contributed to this misunderstanding. (↓E10, E12, E13))

C7 The captain and first officer were not aware of the previous similar incidents within China Airlines and were not warned of the hazards of overriding the autopilots by means of the elevators when the autopilots are in LAND or GO AROUND mode. Airbus did not provide the latest training materials to the simulator training subsidiary company. (↓E10, E12)

C8 The captain's judgment of the flight situation while continuing the approach was inadequate, control takeover was delayed, and appropriate actions were not taken. (↓E9, E10)

C9 Activation of Alpha-Floor function was incompatible with the abnormal out-of-trim situation (and generated a large pitchup moment). This narrowed the range of selection for recovery operations and reduced the time allowance for such operations. (↓E12)

C10 The captain's and first officer's awareness of the flight condition, after the captain took over the controls and during their recovery operation was inadequate. (↓E12, E13, E14)

C11 The captain's and first officer's recovery procedures were improper. (The correct actions would have been to disconnect autothrottle and to turn off the flight director and land under manual control or switch back to LAND mode.)

C12 The crew coordination between the caption and the first officer was poor. The captain initially instructed the first officer instead of taking action (↓E6, E8, E10), the captain issued inappropriate instructions (↓E10), the first officer did not act on the captain's instructions and engaged both autopilots (↓E9), and the first officer took action without communicating the abnormal conditions (e.g., strong force on the control column and the GO lever disengagement not working) (↓E9, E10, E12).

C13 The first officer was very junior in experience compared to the Captain. In the CVR, it appears that the Captain induced competitive behavior in the first officer, which may have encouraged him to continue in manual mode longer than wise and to not report problems:

> Captain: "[other F/O's name] flew very well .... Even I cannot do better .... If he used manual, then it's 100 points. But he only got 90 points because he used auto thrust."

C14 The warning and recognition functions for THS movement were ineffective. There is not a sufficient perceptual alert when the THS is in motion.

C15 There was no warning and recognition function to alert the crew directly and actively to the onset of the abnormal out-of-trim condition.

Level 3 Systemic Factors

Although this accident report has some of the same limitations of other aircraft accident reports, such as focusing on pilot error, assigning blame, and proximate events, it does a good job of investigating and suggesting systemic factors beyond pilot error. The investigation committee had a wide variety of experts involved, including academics and industrial experts without conflicts of interest or industry-specific cultural biases. Law suits associated with the accident have also been a source of additional information and informed speculation as to more systemic factors.

I. Flaws in the Safety Culture

S1 *Overconfidence in automation*

As described under the Warsaw accident, the automation is given final authority by the Airbus engineers. For example, the autopilot does not automatically disconnect if the pilot applies force to the control wheel. One of the recommendations in the report is that Airbus consider design changes allowing autopilot disconnect and manual override functions by which crews can safely control the aircraft, irrespective of flight altitude or phase by applying a force exceeding a certain level on the control column (↓C5).

S2 *Incorrect prioritization of changes to the automation*

Procedures for determining the categorization of changes to the aircraft automation allowed the Service Bulletin to be classified as Recommended rather than required. This is not an isolated incident. The changes that were made to the automation after the Airbus accident while landing at Mulhouse have not been implemented on all Airbus aircraft and were not labeled as mandatory. Political issues occasionally arise even in the issuance of airworthiness directives by government agencies (↓C1, C2 C4).

S3 *Slow understanding of the problems associated with human–automation mismatch within the aviation industry.* (↓C3, C4, C6, C9, C10)

Commercial aviation is the first industry where shared control of safety-critical functions between humans and computers has been widely implemented. As such, they are finding the problems associated with this change from human control to shared control. The very difficult problems that arise, such as those associated with mode confusion and deficiencies in situational awareness, are slow to be recognized. It is easier simply to blame the pilot for an accident than to investigate what aspects of the system design may have led to human errors.

S4 *Ignoring warnings*

Although there had been several serious related incidents, the modification of the FMS was only recommended (not mandatory) and the appropriate governmental authorities did not issue an AD.

II. Ineffective Organizational Structure and Communication Deficiencies

S5 *Inadequate communication of safety information.*

The captain and F/O were not informed of previous similar incidents within China Airlines and elsewhere (e.g., Helsinki 1989, Moscow 1991) that led to the changes implemented in the Service Bulletin. In addition, France did not issue an airworthiness directive promptly to alert airlines to the design flaw in the automation (↓C7, C2).

III. Ineffective or Inadequate Technical Activities

S6 *Inadequate design of feedback to pilot* (↓C14, C15).

Many of the common problems found in human–automation interaction lie in the human not getting appropriate feedback to monitor the automation and to make informed decisions. For example, this accident involved inadequate warning and recognition functions for THS (trimmable horizontal stabilizer) movement—when the THS enters or is close to an out-of-trim situation or when it continues to move for more than a certain period of time, regardless of autopilot engagement or disengagement.

S7 *Inadequate documentation procedures* (↓C6).

These include unclear descriptions of the AFS (Automatic Flight System) in the FCOM (Flight Crew Operating Manual) prepared by the aircraft manufacturer.

S8 *Automation complexity beyond human ability to understand it* (↓C4, C6,).

Accidents, surveys, and simulator studies have all emphasized the problems pilots are having understanding digital automation. Problems are especially found with controls and operations the crews rarely experience in daily flight, such as mode changes and manual overrides during autoflight.

S9 *Inadequate system engineering and safety analysis* (↓C3, C4, C5, C9).

Once again, a system accident occurred where the components acted either correctly or, as with the humans, within reasonable expectation, but the interactions among the components led to an accident. And again, better modeling and analysis tools would help. These tools must include digital components in the models and analyses, for example, identifying the problems in the relationship between the Alpha floor function and the out-of-trim condition or the interaction of the THS and the autopilot.

S10 *Inadequate protection and interlocks for software errors* (↓C5, C9, C15).

There were, for example, no out-of-trim prevention functions or checks when such incorrect states were reached and alerting of the pilots. The accident report suggests that Airbus should consider incorporating functions to prevent an abnormal out-of-trim condition from arising from a prolonged override operation of the autopilot by acting on the pitch axis via the control column, which moves the THS in the opposite direction to the elevator movement.

S11 *Inadequate cognitive engineering (crew resource management)*

There appear (from accident reports) to be problems in mutual confirmation by crews while operating and monitoring the automatic flight system (AFS) mode changes in advanced technology aircraft. For example, it is harder to see what the other crew member is doing when he or she is typing on a keyboard rather than pushing a button or turning a dial on a control panel. In addition, implicit social psychology factors may be preventing optimal crew resource management. There is a need for standardization of terms used for instruction, response, confirmation, and execution of operations in order to ensure that crews can have appropriate situational awareness of the flight; procedures for mutual confirmation by crews operating and monitoring the AFS mode changes of advanced technology aircraft; and reinforcement of standard callout. (↓C12, C13)

3.8 Cali

3.8.1 Background

On December 20, 1995, American Airlines Flight 965, a Boeing 757-223, on a regularly scheduled passenger flight from Miami International Airport to Alfonso Bonilla Aragon International Airport in Cali, Colombia, operating under instrument flight rules, crashed into mountainous terrain while on descent to land at Cali. Alfonso Bonilla Aragon Airport in Cali is located in a long, narrow valley oriented north to south. Mountains extend up to 14,000 feet to the east and west of the valley. Of the 155 passengers, 2 flightcrew members, and 6 cabincrew members on board, 5 passengers survived the accident (all were sitting in the same row), although one of those passengers later died. A dog in a cargo crate was also unharmed.

The aircraft incorporated a flight management system (FMS) that includes a flight management computer (FMC) with a worldwide navigation data base containing radio frequencies, latitude and longitude coordinates of relevant navigation aids, and coordinates of airports capable of B-757 operations. The FMC data base also includes B-757 performance data that, combined with pilot inputs, governs autothrottle and autopilot functions. The FMS monitors the system and engine status and displays the information, as well as airplane attitude, flightpath, navigation, and other information through electronically generated CRT displays.

Pilot input into the FMS can be made in either of two ways: (1) through two keyboards and associated CRTs, known as control display units (CDU), one for the captain and one for the first officer, or (2) through more limited FMS input via controls on the glareshield. The FMS can exercise almost complete flight path control through pilot inputs into the CDU's: Either pilot can generate, select, and execute all or part of a flightpath from origin to destination through CDU inputs. Pilots can also select an instrument approach procedure from the approaches stored in the FMS data base and then either fly the approach manually or direct the FMS to fly it electronically. Retrieving the available approaches and selecting a procedure requires several key strokes on the CDU.

3.8.2 Chain of Events

E1 The flight was delayed 34 minutes in leaving the gate at Miami while waiting for the arrival of connecting passengers and baggage. After leaving the gate, the flight experienced another ground delay of 1 hour and 21 minutes. The flight dispatcher stated the ground delay was related to gate congestion. The flight departed Miami International Airport at 1835, with an estimated time enroute to Cali of 3 hours 12 minutes.

E2 The flight to Cali was uneventful.

E3 The flight plan and preflight briefing called for an ILS approach to Runway 1. After contacting the Cali approach controller, the controller offered and the flightcrew accepted an offer to land on runway 19 instead.

E4 The captain misinterpreted the Cali approach controller's clearance to *proceed to* Cali as a clearance *direct to* Cali. The captain's readback of the latter clearance received an affirmative response from the controller. In changing the FMS programmed flightpath to proceed *direct* to Cali VOR, all fixes were removed between the aircraft's present position and Cali, including Tulua (ULQ), which was the fix the controller had told them to proceed toward and report at.

E5 The flightcrew crossed the initial approach fix ULQ without realizing they had done so and without acknowledging the crossing to the controller.

E6 At 2137:29, the captain asked the controller if AA965 could "go direct to [the non-directional beacon] Rozo and then do the Rozo arrival." The controller agreed but later stated that the question made little sense because Rozo was a beacon located just before the approach end of the runway and not an initial or intermediate approach fix located considerably before the runway.

E7 The crew extended the speedbrakes, which increased the rate of descent.

E8 Either the captain or the first officer selected and executed a direct course through the FMS to the identifier "R" in the mistaken belief that R was Rozo, most likely because R was identified as such on the approach chart. However, instead of selecting Rozo, they had selected the Romeo beacon, located near Bogota, some 132 miles east-northeast of Cali. This course was at approximately a right angle to the published course. In executing a turn toward Romeo rather than Rozo, the flightcrew had the aircraft turn away from Cali and towards mountainous terrain to the east of the approach course, while the descent continued.

E9 At this time, both pilots also attempted to determine the aircraft's position in relation to ULQ, the initial approach fix. Neither flightcrew member was able to determine why the navaid was not where they believed it should be and neither noted nor commented on the continued descent.

E10 The CVR indicates that the flightcrew became confused and attempted to determine their position through the FMS. At 2138:49, the first officer asked "Uh, where are we?" and again, 9 seconds later asked, "Where are we headed?" The captain responded, "I don't know ... what happened here?" The discussion continued as each attempted to determine the position and path of the aircraft relative to the VOR DME 19 approach to Cali.

E11 The interaction with the controller continued at 2140:01, when the captain announced his position and properly interpreted the approach when he asked "... You want us to go to Tulua and then do the Rozo ... to the runway?" While this question demonstrated that the captain understood the appropriate flight path necessary to execute the approach, his position report contradicted his statement because the aircraft had already crossed ULQ and therefore would have to reverse course to comply with his statement. The controller did not question this communication although he later stated that this question made little sense.

E12 At 2140:40, the captain indicated he was having difficulty again, apparently in locating Tulua VOR through the FMS. Over one minute later, the deviation from course was recognized by both pilots and a return to the extended runway centerline was attempted by turning right. However, since they had been flying on an easterly heading for approximately one minute and were now well east of the prescribed course, the direct track back towards "centerline" or Rozo, about 2 miles north of the approach end of runway 19, took the flight towards mountainous terrain, which was then between them and the approach end of the runway.

E13 After turning, they did not stop their descent. There is no evidence the crew recognized the proximity of the terrain (high mountains) to the aircraft's flight path.

E14 At 2141:15, the CVR recorded from the cockpit area microphone the mechanical voice and sounds of the aircraft's ground proximity warning system (GPWS) "terrain, terrain, whoop,

whoop." A sound similar to autopilot disconnect warning began. The mechanical voice and sound continued "pull up, whoop, whoop, pull up."

E15 After the GPWS alerted, the first officer initiated a GO AROUND and correctly followed American Airlines' procedures regarding GPWS escape maneuvers.

E16 The spoilers (speedbrakes) that had been extended during the descent were not retracted. There is debate about whether retraction would have allowed the aircraft to clear the mountain top.

E17 The aircraft entered into the regime of stick shaker stall warning, nose up attitude was lowered slightly, the aircraft came out of stick shaker warning, nose up attitude then increased, and stick shaker was reentered. The CVR ended at 2141:28. The aircraft hit a mountain 250 feet below the top of a ridge.

3.8.3 Identified Causes of the Accident

The investigation "examined the flightcrew actions to determine how a properly trained and qualified crew would allow the airplane to proceed off course and continue the descent into an area of mountainous terrain." The accident report, written by the Aeronautica Civil of the Republic of Colombia concluded that the probable causes were:

1. The flightcrew's failure to adequately plan and execute the approach to runway 19 at SKCL and their inadequate use of automation.

2. Failure of the flightcrew to discontinue the approach into Cali, despite numerous cues alerting them of the inadvisability of continuing the approach.

3. The lack of situational awareness of the flightcrew regarding vertical navigation, proximity to terrain, and the relative location of critical radio aids.

4. Failure of the flightcrew to revert to basic radio navigation at a time when the FMS-assisted navigation became confusing and demanded an excessive workload in a critical phase of the flight.

The report also identifies four contributing factors, although I cannot determine any basis for distinguishing between what they identify as the probable "causes" and these "contributing factors":

1. The flightcrew's ongoing efforts to expedite their approach and landing in order to avoid potential delays.

2. The flightcrew's execution of the GPWS escape maneuver while the speedbrakes remained deployed.

3. FMS logic that dropped all intermediate fixes from the display(s) in the event of execution of a direct routing.

4. FMS-generated navigational information that used a different naming convention from that published in navigational charts.

3.8.4 Hierarchical Model

Level 2 Conditions

Although, like the other aircraft accident reports the investigation focused almost exclusively on pilot behavior, this report stood out from the others in its attempts to hypothesize reasons for the behavior.

C1 One condition related to most of the events was the difference in experience in South American operations between the two cockpit crew members. The Captain had flown into Cali a total of 13 times previously, including twice in the previous 11 days. All of these flights had taken place at night and none had used Runway 19 (Runway 1 was the standard approach with ILS). The first officer, however, had never flown into Cali and may therefore have relied on the captain's experience, relaxed his vigilance, and been less likely to question his actions. For example, at 2133:35, the first officer asked the captain for the transition level at which altimeter settings were to be changed on approach at Cali. Two minutes later, at 2135:44, he asked the captain whether speed restrictions were required as well. Throughout the approach, the captain's experience into Cali appears to have reduced the first officer's otherwise assertive role as the pilot flying. (↓E3-E13)

C2 It is hypothesized that the crew accepted the offer to land on the closer runway—despite the fact that it reduced their preparation time—because they wanted to minimize the effect of the delay in departure from Miami on the flight attendants' FAA-mandated rest requirements. (↓E3)

C3 Approaching Cali from the north, the flight plan and preflight briefing called for an ILS approach to Runway 1, which would require flying past the airport and turning back (which was the usual procedure, even for those flights coming from the north, as wind conditions normally favored that approach). When wind was not a factor, traffic was low, and weather conditions and visibility were good, flights coming from the north were sometimes given the option of a "straight-in" landing on Runway 19. In this case, the crew were offered and accepted the straight-in Runway 19 arrival, which saved them time but also provided less time to prepare for the landing and required an expedited descent using their speedbrakes.

C4 They were too high, too fast, and too close for an unhurried approach to runway 19. This type of situation, however, was not unusual for pilots flying in the U.S. As a result of being hurried, the complete procedure review, briefing, and full reprogramming of the FMS were never accomplished for the new approach.

Experienced persons can quickly make decisions based on cues that they match with those from previous experiences encountered in similar situations. It is likely that when previously faced with similar situations, such as the opportunity to execute an approach that was closer to the airplane's position than the approach anticipated, the pilots of AA965, each of whom had acquired years of experience as air transport pilots, accepted the offers and landed without incident. (↓E3)

C5 No approach control radar was available. Anti-government guerillas in 1992 had destroyed the only radar serving Cali. (↓E6, E8, E11)

C6 The primary means for exchanging information in air-ground communications is the language of the ground stations, which will in most cases be the national language of the country

responsible for the ground station. When English is not the normal language of the station, the English language should be available on request. The air traffic controller stated that he would have asked the pilots more detailed questions regarding the routing and the approach if the pilots had spoken Spanish. (↓E4)

C7 Because of the ability of ATC in the U.S. to provide radar surveillance over most of the airspace and the integration of computer software with radar to alert controllers to aircraft that are descending towards terrain, pilot requests for clearances direct to a fix are often made and often granted. A common problem in pilot-controller communication is one or both of them hearing what they expect to hear rather than what was actually said. With the controller's affirmative response (to the readback of the direct clearance to Cali), the pilot must have assumed that the controller understood his intent and agreed with it. (↓E4)

C8 The air traffic controller said that, in a non-radar environment, it was unusual for a pilot to request to fly from his or her present position to the arrival transition. He also stated that the request from the crew to fly direct to the Rozo VOR, when the flight was 38 miles north of Cali, made no sense to him. He said that his fluency in non-aviation English was limited, and he could not ask them to elaborate on the request. The controller further stated that had the pilots been Spanish-speaking, he would have told them that their request made little sense, and that it was illogical and incongruent. He said that because of limitations in his command of English, he was unable to convey these thoughts to the crew. (↓E6, E11)

C9 The logic of the FMS removed all intermediate waypoints between the aircraft's present position and the "direct to" fix (i.e., the Cali VOR), including Tulua, where the crew was expected by the ATC controller to report their position. This automation behavior compromised the situational awareness of the flightcrew by affecting their awareness of the position of the aircraft relative to critical fixes and navaids necessary for the approach. There is no evidence in the CVR transcript that either pilot recognized that ULQ had been deleted from the display, until they were considerably closer to Cali and were in fact past ULQ.

Since the initial certification of the FMS on the B-757/767, Boeing had developed and implemented a change to the B-757 software that allowed such fixes to be retained in the display. However, this retrofit had not been incorporated into the accident aircraft. (↓E5)

C10 The speedbrakes were necessary because the short distance to Runway 19 required that they expedite their descent. (↓E7)

C11 Both Romeo and Rozo were identified on the charts as "R" (provided in Morse code) and had the same radio frequency, 274 kilohertz. Romeo, a non-directional radio beacon (NDB) in the city of Bogota, and the Rozo NDB both were assigned the letter R as an identifier by the Colombian government. Because the Jeppeson automated database on the B-757 cannot accept duplicate information, the letter R was chosen as the "key" to the Romeo navaid because the Bogota city and airport are bigger than Cali and therefore had the greater number of users. Therefore entering the letter R when in Colombia will call up the Romeo NDB. To retrieve the Rozo NDB, the letters ROZO must be entered into the FMS.

The flightcrew was not informed or aware of the fact that the "R" identifier that appeared on the approach chart for Rozo did not correspond to the "R" identifier (Romeo) that they entered and executed as an FMS command.

In the FMS, all navaids having the same identifier are displayed in descending order of proximity to the aircraft. The one closest is presented first, the second is further from the position,

and so on. Selecting R resulted in a display of 12 NDB's, each of which used the "R" as an identifier. Choosing the first beacon presented in this list resulted from a logical assumption by the pilot. The flightcrew did not know that Rozo, despite its prominent display as "R" on the approach chart and its proximity to the aircraft, was not available for selection as "R' from the FMS (unlike the other navaids having the same identifier) but only retrievable by typing its full name.

Considerable additional differences existed in the presentation of identical navigation information between that of the approach charts and that in the FMS database, despite the fact that the same company supplied the data to both. For example, DME fixes for the Cali VOR DME runway 19 approach that were labeled on the charts as D-21 and D-16 were depicted on the FMS using a different nomenclature entirely, i.e. DF19 and FF19. The company explained that it presented data in the FMS according to a naming convention (ARINC 424) developed for electronic data, while data presented on approach charts met requirements of government civil aviation authorities. (↓E8)

C12 In the rush to start the descent, the captain entered the name of the waypoint (R) and executed instructions without normal verification from the other pilot. The American Airlines policy required that flightcrew members of FMS-equipped aircraft verify coordinates and obtain approval of the other pilot before executing a change of course through the FMS. As noted in C1, the Captain's experience in flying to Cali and the first officer's lack of experience may have led to the first officer relying on the captain's experience and may also partially explain this departure from standard procedures. (↓E8)

C13 Both pilots were experienced in the aircraft and were described as proficient in the use of the FMS by their peers. Most likely because of time pressure, they executed the change of course through the FMS without verifying its effect on the flightpath. The pilots could not have known—without verification with the EHSI (Electronic Horizontal Situation Indicator) display or considerable calculation—that instead of selecting Rozo, they had selected the Romeo beacon. (↓E8)

C14 Either of two reasons could account for the flightcrew's persistence in attempting to land rather than discontinuing the approach: (1) failure to adequately consider the time required to perform the steps needed to execute the approach, and (2) the reluctance of decision makers in general to alter a decision once it has been made. Once they prepared for the approach to runway 19, there is no evidence that the flightcrew revisited the decision, despite increasing evidence that they should have discontinued the approach. (↓E9-E12)

C15 The captain continued unsuccessful attempts to locate Tulua VOR, the initial approach fix, through the FMS rather than using other sources of information such as charts.

Confusion about the FMS presentation, as is true for use of any computer, is often resolved after persistent interaction with it. Thus, it is likely that the captain believed that the confusion he was encountering was related to his use of the FMS and that continued interaction with it would ultimately clarify the situation. The report states that he could not be expected to recognize, because it rarely occurs in regular flight operations, that the fix he was attempting to locate (Tulua VOR) was by this time behind him, and the FMS-generated display did not provide sufficient information to the flightcrew for them to determine this fact.

The actions of the captain are consistent with literature that indicates that under stress, people will tend to narrow their focus of attention. Because of the need to perform multiple

tasks in a limited time and the difficulty in locating a critical navaid, the captain appears to have been under considerable stress, which further compromised his ability to perform in the objective manner needed to develop and maintain good situation awareness. His attention thus narrowed to further repeated unsuccessful attempts to locate ULQ through the FMS. (↓E10, E12)

C16 The evidence indicates that American Airlines, as with other airlines operating FMS-equipped aircraft, emphasized the high reliability of the FMS during training and encouraged pilots to rely on it. Failure of the FMS is so unlikely that if it occurs it is believed most likely to be an electrical system anomaly and not one of the FMS itself. Pilot training for FMS-system failures is generally focused on display or total computer failures and the response suggested is to substitute working displays or computers for non-working ones. As a result, flightcrews have been taught that the FMS is an all-or-none system that will either work or not work and that failures, which are few and far between, will be total. Because the FMS and the electronic displays were functioning, the appropriate pilot assumption was that difficulty in interacting with it was because of pilot input and not something related to the FMS. (↓E9, E10)

C17 Pilots are generally trained to be able to use the capabilities of the FMS, but they are not given much information about the logic underlying much of the performance of the FMS or shown many of the numerous options available to achieve identical goals in the FMS. (↓E9, E10)

C18 At this point, both pilots had lost situational awareness relative to the cleared approach path and terrain: They lost Tulua off the FMS display and they did not have Rozo programmed in. Neither crew member was aware they had passed over the Tulua VOR or that the aircraft was turning left towards Romeo. Cockpit automation has increased the workload during approach, and lots of headdown work programming the computer has been found to reduce situational awareness (in addition to the lack of information provided by the computer in this case). (↓E10)

C19 The clearance clarifications with ATC did not help the situation. Unlike comparable airspace in the U.S. in which the flightcrew had accrued the overwhelming majority of their flight experience, the Cali airspace and controllers were not provided with radar coverage computer software to alert controllers to aircraft deviation from a safe altitude, computer software to enhance the radar image of a particular flight, nor a controller who shared a native language and culture with the flightcrew. Because of these differences, unlike in domestic U.S. airspace, the Cali approach controller was entirely dependent on crew-provided information to determine the location, altitude, airspeed, and climb/descent rate of a flight, and to assess whether that flight required ATC services beyond that provided for in the applicable rules and regulations. In addition, the tools available to U.S. air traffic controllers allow them to monitor the flight for terrain clearance. The accident investigators concluded from the interactions with the controller that the captain and the first officer both incorrectly assumed that a level of redundancy existed in the ability of the Cali controller to provide terrain clearance to the crew when no such ability existed. (↓E4, E6, E11, E13)

C20 Two critical sources of information may have provided some indication to the controller that AA965 was experiencing difficulty: (1) the captain asked the controller two questions regarding the execution of the approach to runway 19 that made little sense, and (2) two of the captain's position reports did not match the time in which they were made. The

accident report suggests that the approach controller was not trained to solicit the information necessary from the flightcrew in order to determine first hand the extent of the difficulty they were experiencing. He also lacked the ability that radar coverage would have provided him to observe the flightpath directly. In addition, he lacked the English language fluency needed to probe the flightcrew, from the subtle hints in the inconsistencies of their responses to him, to learn the extent of their difficulties. According to ICAO (the International Civil Aviation Organization) standards, English language ability by a controller who is not a native English speaker is limited to routine aeronautical communications. The report further argues that it would have been very difficult, in any event, for a Colombian controller to question or critically respond to the statements of an airline captain. (↓E4, E6, E11)

C21 The report argues that the approach controller's experience in Cali was such that the flightcrews of all aircraft arriving from the north recognized the proximity of high terrain to the flight path to the airport. Thus, if a flightcrew was off course and needed assistance from the controller, the controller's natural expectation would be that they would ask for special assistance from him and that they would not continue to descend. It concludes that his training, experience, and guidance under the applicable rules in the non-radar environment of Cali would have made it unlikely for him to solicit the necessary information from the flightcrew of AA965 that would have enabled him or the flightcrew to recognize the precarious nature of their flight path. Consequently, Aeronautica Civil concluded that the Cali controller neither caused nor contributed to the cause of this accident.[18] (↓E11)

C22 The night visual conditions limited the ability to see the terrain. The fact that the captain, the only one of the two flightcrew members to have flown into Cali, had previously landed only at night also limited his appreciation for the presence of the mountains along either side of the approach into Cali. (↓E13)

C23 Evidence shows that the flightcrew did not refer to the local area chart during the flight and only referred to the approach chart. The approach chart does not portray terrain graphically. High points of the terrain were displayed on the chart by several altitude dots and their associated elevations. Although this method presents the necessary information, it takes time to recognize and understand its significance because of the lack of prominence of this information. During a high workload period, or when insufficient time may be available to adequately review the chart, pilots may not be able to assimilate that information to gain a comprehensive view of the aircraft, its flight path, and its adherence to the approach parameters.

Before the accident, Jeppesen Sanderson Company, the supplier of approach charts and navigation information for electronic navigation data bases, began to change the portrayal of terrain to use graphics similar to those used in topographic charts, with colors added to enhance the prominence of terrain and heighten its contrast with other information on the chart. The criteria the company uses to determine whether to display terrain information on approach charts require (1) that terrain is 2000 feet above the airport within 6 miles of

[18] The actions of the controllers and the state of air traffic control in Colombia obviously *did* contribute to the accident. The fact that the report writers (the Colombian Aeronautic Civil) felt the necessity to exclude any possible contribution (while implicating almost everything and everyone else) may be due to the focus on "blame" in accident reports. The report includes recommendations for changes for American Airlines, the U.S. FAA, ICAO, and companies such as Jeppesen-Sanderson, but none for themselves or their facilities. It is interesting also that while defending their controller's actions as following ICAO standards, they at the same time recommend that ICAO standards be changed without any recommendation that their own standards and practices be changed.

the airport and (2) on local area charts, that terrain is elevated more than 4000 feet above the airport. Because neither of these criteria was met in the VOR DME runway 19 approach chart, the chart used by the flight crew did not graphically show the high terrain on either side of the descent into Cali. (↓E13)

C24 Terrain information was also not shown on the electronic cockpit displays. Among the features of an FMS is a map display, which at the option of the pilot graphically displays on the EHSI (the Electronic Horizontal Situation Indicator) the aircraft's present position and future flightpath, as well as the location and relative position of adjacent navigational aids and airports. In FMS-equipped aircraft, the portrayal of flight path (in the Boeing/Honeywell Systems-equipped aircraft by means of a magenta colored line) is so accurate and informative that pilots are permitted to rely on it as the primary means of navigation, believing that they are secure in the knowledge that the aircraft will be maintained along a safe flight path as long as the magenta line is followed. However, unlike charts, the FMA-generated displays do not present associated information, such as terrain, and do not display navaids that are behind the aircraft unless specifically directed to do so by a flightcrew member. As a result, pilots who are accustomed to relying exclusively on FMS-generated displays for navigation can, over time, fail to recognize the relative proximity of terrain and can lose the ability to quickly determine that a fix or beacon is behind them. The evidence suggests that this partially explains the difficulty of the AA965 flightcrew in locating the ULQ waypoint. The history of the flight indicates that the AA965 flightcrew did not effectively use all the navigation information that was available to them and that they relied almost exclusively on their EHSI for navigation. (↓E13)

C25 Although during the descent the flightcrew had no terrain information on the approach chart or the EHSI, American Airlines did provide the flightcrew with written terrain information on the flight plan noting "Critical terrain exists during the descent—Strict adherence to STAR [standard approach route] necessary for terrain clearance." (↓E13)

C26 The flightcrew had become acclimated to the hazards of flying in mountainous terrain. In the years since they received their initial Latin American training, both had operated in South America and the captain had operated into Cali 13 times without incident. Over time, repeated exposure to flight operations into potentially hazardous environments can become routine as pilots acclimate to the environment and their vigilance is diminished. (↓E13)

C27 American Airlines had a list of South American airports (Bogota, Quito, and La Paz) that were critical because of the effects of their high altitudes on aircraft performance. Cali was not on this *hit list.* Pilots were required to meet additional training requirements before being permitted to fly into these cities for the first time. Because Cali was not at high altitude, it was not given "special consideration" and the flightcrew may not have exercised the same level of vigilance when operating into Cali as they would have when operating into the three target airports. (↓E13)

C28 The guidance given in AA written guidance and in training did not have sufficient impact to be recalled in a time of high stress and workload. Unless information is presented regularly in a novel and interesting way, pilots may fail to display the lessons of earlier training when those lessons are most needed. The investigation team suggested that the pilots of AA965 became task saturated and did not recognize the hazards the airline had warned them about as they were encountered during the accident approach. (↓E13)

C29 The Ground Proximity Warning System (GPWS) procedure in the American Airlines Flight Operations Manual addresses the crew actions that must be carried out to attain maximum climb performance of the aircraft in order to overcome the obstacles ahead of its flight path. These pilot actions include the disengagement of autopilot and autothrottle system as well as selecting maximum power and attaining best angle of climb. Retracting the speedbrakes is not explicitly mentioned. Neither the Boeing Operations Manual addressing terrain avoidance nor the American Airlines Operating Manual addressing GPWS escape procedures discuss the need to stow the speedbrakes to extract maximum performance from the aircraft during an escape maneuver. (↓E16)

C30 The speedbrake system in the B-757 consists of overwing control surfaces that extend and retract at the command of the pilot through the aft and forward movement of the speedbrake control lever located in the top left side of the center control stand. In flight, operation of the speedbrake system is manual. Automatic extension and retraction are restricted to the landing phase and is activated upon main wheel touchdown and forward movement of the power levers respectively.

The speedbrake handle may be pulled back to any desired level of spoiler panel deflection. The speedbrake handle also has an armed detent position to allow automatic deployment on landing. When the automatic speedbrake feature is in use and the aircraft is on the ground, advancement of either thrust lever from flight idle will cause any extended spoiler panels to stow. However, advancing the thrust levers in flight has no effect on deployed speedbrakes.

Although for both a controlled flight into terrain (CFIT) and windshear escape maneuver, immediate retraction of the speedbrakes is needed to achieve the maximum climb performance of the aircraft, during periods of high workload flightcrews may not recognize that speedbrakes have remained extended. While the automatic stowing of speedbrakes might provide a significant safety enhancement, automatic retraction of speedbrakes in a go-around maneuver may result in unwanted pitch excursions. So there would need to be enhancements to the pitch control system to compensate for any automatic retraction of the speedbrakes. (↓E16)

C31 There is no evidence that the flightcrew attempted to retract the speedbrakes (spoilers) manually. When a speedbrake fails to retract, an amber center panel speedbrake light goes on as well as an amber Engine Indication and Crew Alerting System (EICAS) SPEED BRAKE EXT message, master caution light, and chime. The speedbrakes remained extended and the CVR did not record the chime.

It is likely that neither pilot recognized that the speedbrakes (which they had deployed earlier to increase the descent rate) remained deployed. Due to the limited aerodynamic effect of the speedbrakes, flightcrews may become unaware that they are in the extended mode. Annunciation of speedbrake deployment becomes activated only in landing configuration and/or below a specified altitude. Neither condition was true in this case, and therefore there were few clues available to alert the pilots to the extended condition.

Boeing's B-757 Flight Crew Training Manual provides one method of monitoring the status of speedbrake deployment. The manual states that "The Captain should keep his right hand on the speedbrake lever whenever they are used in flight. This will preclude leaving the speedbrakes extended." American Airlines does not have a similar procedure. (↓E16)

Level 3 Systemic Factors

I. Flaws in the Safety Culture

S1 *Complacency about software: Underestimating and misunderstanding software-related risks and software "failure" modes.*

The evidence indicates that American Airlines, as with other airlines operating FMS-equipped aircraft, trained its pilots to rely on the FMS. Because of the high reliability of the FMS, failures are assumed to be the result of an electrical system anomaly and not the FMS itself. As a result, flightcrews have been taught that the FMS is an all-or-none system that will either work or not work and that failures, which are few and far between, will be total. Therefore, because the FMS and the electronic displays were functioning, the appropriate pilot assumption was that difficulty in interacting with it was due to pilot input and not something related to the FMS. (↓C13, C5, C16)

S2 *Overconfidence in and dependence on automation*

The history of the flight indicates that the AA965 flightcrew did not effectively use all the navigation information that was available to them and that they relied almost exclusively on their EHSI for navigation.

Pilots who are accustomed to relying exclusively on FMS-generated displays for navigation can, over time, fail to recognize the relative proximity of terrain and can lose the ability to quickly determine that a fix or beacon is behind them. (↓C24)

The FMS is highly reliable and presents navigation information in an easily interpretable manner. Researchers have shown that operators will increase their use of and reliance on an automated system as their trust in the system increases.

Merely informing crews of the hazards of overreliance on automation and advising them to turn off the automation is insufficient and may not affect pilot procedures when it is most needed. In a previous accident involving an FMS-equipped aircraft, the flightcrew of a Thai Airways Airbus A-310 that crashed into the side of a mountain while on approach to Katmandu, Nepal, lost awareness of terrain and of the location of navaids that were in reality behind the aircraft. Investigators of the Thai Airlines crash found that after encountering and correcting a system anomaly during the approach, which was a period of high workload, the pilots lost awareness of the aircraft's course and did not realize that they were headed towards, and not away from, high terrain. The displayed navigation information was confusing to them and they repeatedly attempted to use the FMS to clarify their understanding of the aircraft's position. The aircraft impacted the terrain while both the captain and first officer were interacting with the FMS. Numerous parallels exist between these two accidents: in both, pilots of sophisticated glass cockpit aircraft on approach in mountainous environments relied on the FMS and continued to interact with the FMS in futile efforts to gain situational awareness, at the expense of reference to charts that could have enhanced terrain awareness.

American Airlines uses the Katmandu accident report to inform its crews of the hazards of overreliance on automation. The Cali accident report cites this previous instance of a very similar event but still blames the crew of AA for the same behavior even though there seems to be no evidence that any steps were taken by the automation designers or by anyone else to prevent its reoccurrence except to give the pilots a copy of the previous accident report. (↓C5, C12, C13, C24)

S3 *Incorrect prioritization of changes to the automation*

A retrofit to allow fixes to be retained in the display had not been incorporated on the aircraft. There is no reason given in the report for this omission by American Airlines, but it is not unusual for airlines not to incorporate automation changes even when the change was prompted by a previous accident linked to that automation feature unless the change is required by their civil authorities. (↓C9)

S4 *Complacency*

The accident investigators concluded from the interactions with the controller that the captain and the first officer both incorrectly assumed that a level of redundancy existed in the ability of the Cali controller to provide terrain clearance to the crew when no such ability existed. The flight crew had accrued the overwhelming majority of their flight experience in U.S. airspace where conditions were quite different.

In addition, the crew may have become acclimated to the hazards of flying in mountainous terrain. Potentially hazardous environments can become routine as pilots acclimate to the environment and their vigilance is diminished.

Finally, the first officer may have relied on the captain's experience in operating into Cali and consequently relaxed his vigilance. (↓C5, C12, C13, C24)

S5 *Flawed resolution of conflicting goals*

FAA and airlines make rules for layovers and there is pressure not to be late. The pilots had to make a tradeoff decision between one risk and another and chose to try the quicker landing.

II. Ineffective Organizational Structure and Communication Deficiencies

S6 *Limited communication channels and poor information flow*

There were several types of communication problems that existed between the pilots and the controller. (↓C6, C7, C8, C18)

III. Ineffective or Inadequate Technical Activities

S7 *Conflicting and insufficient documentation*

Numerous important differences existed between the display of identical data on the approach charts and on the FMS-generated displays, despite the fact that the same supplier provided both.

The lack of coordinated standards for the development and portrayal of aeronautical charts and FMS data bases and displays led to a situation where not only were the charts and displays different in appearance, but the basic data are different. The lack of commonality is confusing, time consuming, and increases pilot workload during a critical phase of flight, i.e., the approach phase. The accident report recommended that the FAA and ICAO develop and implement standards. In meantime, they said that FAA should require Jeppesen-Sanderson Company to inform airlines operating glass cockpit aircraft of the presence of each difference in the naming or portrayal of navigation information on FMS-generated and approach chart information, and require airlines to inform their flightcrews of these differences.

A second problem was the fact that the approach charts did not provide terrain information in a form that can be quickly assimilated during a high workload period. The EHSI did not provide terrain information at all. After the accident, Jeppeson-Sanderson began adding terrain contour information to approach charts.

Finally, procedures to avoid CFIT by extracting maximum escape performance are not well developed or documented for pilots. The Boeing Operations Manual and the American Airlines Operating Manual (as well as American Airlines procedures) do not address the need to stow the speedbrakes to extract maximum performance from the aircraft duing an escape maneuver. (↓C11, C23, C24, C29)

S8 *Inadequate safety engineering and risk control*

Standards for commercial aircraft certification still focus on component reliability and redundancy and thus are not effective against these types of system accidents. The software satisfied its specifications and did not "fail" yet it obviously contributed to the flight crew's actions and inactions. (↓C9, C11, C15, C16, C24, C30)

S9 *Inadequate design of feedback to pilot*

Terrain information was not shown on the electronic horizontal situation indicator (EHSI) nor was it graphically portrayed on the approach chart. The FMS also did not display navaids that are behind the aircraft unless specifically directed to do so by a flightcrew member.

Neither pilot recognized that the speedbrakes were extended during the GPWS escape maneuver due to the lack of clues available to alert them about the extended condition. After the accident, studies began regarding mandating auto-spoiler retraction and angle-of-attack cockpit displays. However, they were not mandated. In addition, changes to software that drops waypoints from the FMS display on "direct to" command was also not mandated.

As a result of the accident, EGPWS (Enhanced Ground Proximity Warning Systems (which had been available but were not normally installed) *were* routinely added to new transport aircraft, providing more time for evasive action. (↓C11, C15, C24, C31)

S10 *Automation complexity beyond human ability to understand it*

The flight crew's situation awareness was compromised by a lack of information regarding the rules of logic and priorities of the navigation database in the FMS. In addition, the crew may have been unaware that advancing the thrust levers in flight has no effect on deployed speedbrakes as it does when the aircraft is landing.

This accident demonstrates that proficiency in the use of the FMS, without knowledge of the logic underlying such critical features as the design and programmed priorities of its navigation data base, can lead to its misuse. Such lack of understanding of the logic may partially account for the finding that the use of the FMS often increases workload during periods of already high workload. Numerous studies have shown that pilots are surprisingly uninformed about how the automation works [39, 3]. (↓C17, C30, C31)

S11 *Ineffective or inadequate cognitive engineering*

Cognitive engineering, particularly that directed at the influence of software design on human error, is still in its early stages and much more research is needed. In this accident, there was task saturation and overload, poor situational awareness (inadequate mental models of the automation and the situation), and distraction from appropriate behavior. (↓C9, C18, C23, C28, C30)

Chapter 4

Conclusions

This report had two goals: (1) to compare and evaluate some commonly used accident models and (2) to identify systemic factors in recent aerospace accidents, particularly those involving software-intensive systems. Some basic conclusions can be drawn for each of these goals.

4.1 Comparisons and Evaluation of the Models

In general, the participants in the seminar found that event chains were useful in understanding the mechanism, particularly the physical mechanism, involved in an accident. They were less useful, however, in identifying and understanding the other factors in accidents. The aircraft accident reports, in addition, were found to be focused so much on establishing blame (and making sure the pilots were the focus of that blame) that the events and conditions included were often limited.

While any model reflects a bias, event chain models provide too little guidance to the modeler. In examining the accident reports in Chapter 3, I found tremendous subjectivity and even bias in the events and conditions chosen to explain the accidents. Some of the bias was introduced simply in the wording chosen. For example, the Cali report continually refers to the pilots' "self-induced" time pressure. The frequent addition of the qualifier "self-induced" serves only to remind the reader that the pilots accepted the offer to land on Runway 19 rather than Runway 1 and did not abort the landing to go around again and therefore can be held responsible for all the events that followed. In essence, the words are used to make sure the readers do not miss the investigators' conclusions about where the ultimate blame lies. Although I tried to remove such blame-inducing words from my description of the accident, it turned out to be extremely difficult to eliminate, particularly the more subtle wording biases.

While wording biases were possible (but difficult) to eliminate when creating the models in Chapter 3, bias and filtering that influenced the factors considered in the report and thus the factors investigated was more subtle and impossible to remove. I tried to comment on the most egregious and obvious omissions, but after the fact and without any extra information, it is not possible to determine everything that was omitted from the investigations and thus the reports. In several of the reports, I concluded that the conditions mentioned were carefully chosen to influence the readers' conclusions. In others, the same effect was most likely simply a result of the particular experiences, backgrounds, and mental models of the accident investigators. The farther away from the basic facts one gets, the more this is a factor. My own biases framed the systemic factors I included in the hierarchical model, for example.

Some reports contained more speculation than others. When labelled as such, it was extremely helpful in expanding the readers' understanding of the accident. Because information is always

limited in any after-the-fact accident investigation, a too narrow recitation of the known events alone does not provide enough understanding. Providing alternative explanations of the events is helpful in expanding one's view of possible causal factors.

The stopping factor used in tracing backward along an event chain from the loss event was more important than I had imagined. The problems became obvious when comparing different accidents that had similar mechanisms. For example, the MCO report labels the software units problem as a *root cause* and concentrates most of its detailed evaluation on the events that occurred after the introduction of the error. This resulted in more emphasis in the report on the operator actions that failed to prevent the loss after the error had already been introduced than on the events and conditions that allowed the introduction of the software error itself. In this respect, the MCO report resembles the aircraft accident reports, which also concentrate on operator behavior. The Titan IV-B accident report provides a sharp contrast in that the events and conditions that allowed the "units" (data) error to be introduced were investigated in great detail in addition to why the error was not caught during operations. The resulting recommendations from the Titan/Milstar report are more informative about how to reduce the incidence of accidents with similar etiology in the future.

According to the MCO report, the definitions used come from a NASA standard. NASA might consider reevaluating the approach to accident investigation implied by that standard. The definition of a root cause as an event (causal action or failure to act) in the event chain eliminates the possibility of finding systemic or indirect factors as root causes and can lead the investigation away from what might be the most important factors in terms of preventing future accidents. According to the definitions provided in the MCO report, indirect and systemic factors *can* be labeled as "contributory causes" but not as root causes. Even normal English usage seems to violate this definition of root cause, and using the NASA definition could misdirect accident investigations.

The hierarchical model was an improvement over a basic chain of events model in separating events from the various types of explanations that could be given for those events. In turn, that allowed me to evaluate the explanations in a more objective fashion and to more easily detect omissions and biases. It also helped to identify conditions that indirectly explained the events or explained all or some subset of the events. These complex relationships became clear in the surprising amount of effort it took me to separate the first and second levels of the models using the information contained in the accident reports. I had never expected that writing Chapter 3 would take the enormous amount of time that it ended up taking. Note that only the Titan IV-B and *Challenger* reports included the events I needed to provide a complete hierarchical model (explanation) of the accident. Such omissions became very clear while creating the levels of the hierarchical model and should assist the accident investigators in focusing their attention on the relevant events and conditions.

While omissions and incompleteness in the accident reports became clear from the separation into level 1 accident mechanisms and level 2 explanatory conditions, I also became aware that my abstraction of level 3 systemic factors from the level 2 conditions was filled with my own biases and subjectivity and that the mappings were not nearly as clear as I thought they would be. In the end, I used the categories of accident *root causes* I had identified in my book. These categories were originally derived from studying large numbers of accidents and from the accident theory literature. While providing guidance to accident investigators in identifying systemic factors may have the negative effect of limiting those that are considered, my experience in writing this report convinces me that the positive effect of expanding those factors considered is more likely.

My overall conclusion from this exercise is that better accident models are needed, and that will be the focus in Part 2 of this report.

4.2 Common Factors in the Accidents

Only those factors actually included in accident reports can be considered and not those that are omitted or filtered out. Given the incompleteness in the accident reports (as described in the previous section), any analysis of common factors is very limited in the conclusions that can be reached. Some factors that may be common to all the accidents may only have been investigated by one of the investigation boards—but that does not mean it did not play an important role in the other accidents.

Having stated these limitations, however, there are many common factors in the reports that do not seem to stem from obvious biases in the reports themselves. I have separated the spacecraft and aircraft accidents because the factors investigated were so different.

4.2.1 Systemic Factors in Spacecraft Accidents

On the surface, the events and factors involved in the five spacecraft accidents appear to be very different except that software played a role in four of the accidents. But a more careful, detailed analysis of the systemic factors in these accidents shows striking similarities. More might have been found if all the accident reports had been as thorough as Challenger and Titan in their investigation of the sources of the design flaws.

The systemic factors are again grouped into the three categories: (1) flaws in the safety culture, (2) ineffective organizational structure, and (3) inadequate or ineffective technical activities.

I. Flaws in the Safety Culture

Overconfidence and complacency

Success is ironically one of the progenitors of accidents when it leads to overconfidence and cutting corners or making tradeoffs that increase risk. Petroski, in his book *To Engineer is Human*, describes this common phenomenon. It is not new, and it is extremely difficult to counter when it enters the engineering culture in an organization.

The Rogers Commission investigating the *Challenger* accident noted that Shuttle flights had become routine: safety margins were relaxed over time and risks were tolerated because they had been experienced before—no adequate attempt was made to eliminate them. NASA and its contractors accepted escalating risk (although they probably did not realize it was escalating) because they had gotten away with it previously. The MCO report noted similarly that because JPL's navigation of interplanetary spacecraft had worked well for 30 years, there was widespread perception that "orbiting Mars is routine" and inadequate attention was devoted to risk management. A similar culture apparently permeated the MPL project.

Complacency can manifest itself in a general tendency of management and decision makers to discount unwanted evidence of risk. In an analysis of an accident in the Moura mine in Australia, Hopkins describes a general phenomenon he calls a *culture of denial* in which it is generally believed that there is no significant risk and production pressures lead to dismissing any evidence to the contrary.

A recommendation common to several of the spacecraft reports was to pay greater attention to risk identification and management. The investigators found that the project management teams appeared primarily focused on meeting mission cost and schedule objectives and did not adequately focus on mission risk. As an example, the MCO report recommended that:

> Risk management should be employed throughout the life cycle of the project, much

> the way cost, schedule, and content are managed. Risk, therefore, becomes the "fourth dimension" of project management—treated equally as important as cost and schedule.

A report on the MPL loss concludes that the pressure of meeting the cost and schedule goals resulted in an environment of increasing risk in which too many corners were cut in applying proven engineering practices and in the checks and balances necessary for mission success.

While management may express their concern for safety, true priorities are shown during resource allocation. By the time of the fatal Challenger flight, reductions had been made in the safety engineering functions that essentially made those functions ineffective. MCO and MPL were developed under very tight "Faster, Better, Cheaper" budgets. The Titan Program office had cut support for monitoring the software development and test process by 50% since 1994 and had greatly cut the number of engineers working launch operations. Although budget decisions are always difficult when resources are reduced (and budgets are almost always less than is desirable), the first things to be cut are often system safety, system engineering, quality assurance, and operations, which are assigned a low priority and assumed to be the least critical parts of the project.

Underestimating and Not Understanding Software Risks

Accidents involving software often occur within an engineering culture that has unrealistic expectations about software and the use of computers. The common misperceptions (or myths) take two forms: 1) Software does not fail and all errors will be eliminated through testing and (2) software can be handled using the same techniques as hardware.

The first myth stems from an underestimation of the complexity of most software and an overestimation of the effectiveness of testing. Even software engineers can fall prey to these misbeliefs, as in the Ariane 5 software where it led to unprotected variables, lack of assertions and range checks, etc. The Ariane 5 report notes that software was assumed to be correct until it was shown to be faulty. This form of complacency also plays a part in the common proliferation of software functionality and in unnecessary design complexity.

In the Titan accident, there apparently was no checking of the correctness of the software after the standard testing performed during development. For example, on the day of the launch, the attitude rates for the vehicle on the launch pad were not properly sensing the earth's rotation rate but no one had the responsibility to specifically monitor that rate data or to perform a check to see if the attitude filters were correctly sensing the earth's rotation rate. In fact, there were no formal processes to check the validity of the filter constants or to monitor attitude rates once the flight tape was actually loaded into the INU at the launch site. Potential hardware failures are usually checked up to launch time, but it may have been assumed that testing discovered all software errors and no further checks were needed. Even right before launch, the programmed rate check used a default set of constants to filter the measured rate rather then the actual constants loaded on the Centaur.

The second myth is that the same techniques used to make electromechanical systems safe or reliable will work in software-intensive systems. This myth leads to ineffective and inadequate technical activities directed toward safety. The recommended approach in the Mars Polar Lander report is an example. It and other reports suggest using FMECA or FTA along with appropriate redundancy to eliminate failures. But software and digital systems bring a totally different game to engineering practice. Some classically trained engineers have difficulty appreciating the new and very different engineering environment created by the introduction of software and the new mindset and approaches required. Not only are the failures not random (if the term "failure" makes any sense when applied to something that is pure design separate from the physical realization of that

design), but the complexity of most software precludes examining all the ways it could "misbehave." In addition, the failure modes are very different than for physical devices.

Throughout the accident reports, there is an emphasis on failures as the cause of accidents. The contribution of software to accidents is very different than that of hardware and engineering activities must be augmented to reflect this. Almost all software-related accidents can be traced back to flaws in the requirements specification and not to coding errors—the software performed exactly as specified, but the specification was incorrect.

Understanding these differences between software and other engineering "materials" and implementing alternative approaches that will be effective for software has been slow in engineering. For example, Figure 6-1 (page 22) of the JPL report on the MPL loss contains a figure that shows the "potential failure modes for the [entry, descent, and landing] sequence." Potential hardware failure or misbehavior, such as *Propellant line rupture* and *Excessive horizontal-velocity causes lander to tip over at touchdown*, is identified for each stage except for software. Instead, a statement *Flight software fails to execute properly* is identified as common to all phases. The problem with this is that it provides no useful information—it is equivalent to simply substituting a single statement for all the other hazards identified in the figure with *Hardware fails to execute properly.* Singling out the JPL engineers is unfair here because I find the same types of useless statements about software in almost all the fault trees and failure analyses I see in industry, and this practice is not limited to aerospace.

Software by itself is never dangerous—it is an abstraction without the ability to produce energy and thus to lead directly to a physical loss. Instead, it contributes to accidents through issuing (or not issuing) instructions to other components in the system. In the case of the identified probable factor in the MPL loss, the dangerous behavior was *Software prematurely shuts down the descent engines.* Such an identified unsafe behavior would be much more helpful during development in identifying ways to mitigate the risk than the general statement *Software fails to execute properly*, which provides no guidance to the system or software designers and reviewers. In fact, software probably should not appear in this figure at all—it should instead be identified in a later (more detailed) analysis step in terms of potential specific undesired software behaviors that could lead to many of the hardware failure modes that are listed.

Because of these fundamental differences, changes are necessary in the way complex, software-intensive systems are designed and engineered. Researchers will need to create new engineering techniques that accomplish the goals of the old ones but account for the difference in the components from which complex systems are being built.

Accidents are changing their character. This change is not solely the result of using digital components, but it is made possible because of the flexibility of software. Most of the accidents investigated in this report displayed at least some features of *system* accidents, where all or most of the components implicated in the accident work as specified but the combined behavior of the components lead to disastrous system behavior. Not only did each component in isolation work correctly (i.e., they satisfied their specifications), but, in fact, many of the system design features that contributed to the accident involved standard recommended practice. Protecting against software "failures" will do nothing to protect against system accidents. What we will need in order to deal with system accidents is discussed below, but the first required step is for the engineering culture to recognize the need for change and to recognize that safety and reliability are *different* qualities for software—one does not imply the other. Although confusing reliability with safety is common in engineering, it is perhaps most unfortunate with regard to software as it encourages spending most of the effort devoted to safety on activities that are likely to have no effect.

Overrelying on Redundancy

Redundancy is commonly used to reduce component failures and increase system reliability. The *Challenger* accident is typical. The engineers and managers relied on redundancy without properly evaluating whether the redundancy provided adequate protection. Once the original evaluation had been completed, they continued to believe in the independence of failures of the redundant O-rings long after that independence assumption had been shown to be incorrect. While some of this mistaken reliance had a basis in inadequate communication and documentation procedures, mistaken reliance was placed on redundancy even by those who *knew* the criticality rating had been increased from 1R (highest criticality but risk mitigated by redundancy) to 1 (highest criticality).

Redundancy usually has a greater impact on reliability than safety. System accidents, for example, will not be decreased at all by the use of redundancy. In fact, the added complexity introduced by redundancy has frequently resulted in accidents. In addition, redundancy is most effective against random wearout failures and least effective against requirements and design errors—the latter being the only type found in software. The Ariane report notes that according to the culture of the Ariane program, only random failures are addressed and they are primarily handled with redundancy. This approach obviously failed when both the Inertial Reference System computers shut themselves down (exactly as they were designed to do) in response to the same unexpected input value.

To cope with software design errors, "diversity" has been suggested in the form of independent groups writing multiple versions of software with majority voting on the outputs. This approach is based on the assumption that such versions will fail in a statistically independent manner, but this assumption has been shown to be false in practice and in a large number of carefully designed experiments (see, for example, [17]). Common-cause (but usually different) logic errors tend to lead to incorrect results when the various versions attempt to handle the same unusual or difficult-to-handle inputs. In addition, such designs usually involve adding to system complexity, which can result in failures itself. A NASA study of an experimental aircraft with two versions of the control system found that all of the software problems occurring during flight testing resulted from errors in the redundancy management system and not in the control software itself, which worked perfectly [28].

Assuming Risk Decreases over Time

In the Milstar satellite loss, the Titan Program Office decided that because the software was "mature, stable, and had not experienced problems in the past," they could use the limited resources available after the initial development effort to address hardware issues. In several of the accidents, quality and mission assurance as well as system engineering was also reduced or eliminated during operations because it was felt they were no longer needed or the resources were needed more elsewhere. In MCO, the operations group did not have a mission assurance manager. The *Challenger* accident report also noted cuts in the operational safety activities (the "Silent Safety" program). It is very common to assume that risk is decreasing after repeated successes.

In fact, risk usually increases over time, particularly in software-intensive systems. The Therac-25, a radiation therapy machine that massively overdosed five patients due to software flaws, operated safely thousands of times before the first accident. Industrial robots operated safely around the world for several million hours before the first fatality. Risk may increase over time because caution wanes and safety margins are cut, because time increases the probability that the unusual conditions will occur that trigger an accident, or because the system itself or its environment changes. In some cases, the introduction of an automated device may actually change the environment in

ways not predicted during system design. For example, as operators became more familiar with the Therac-25 operation, they started to type faster, which triggered a software error that had not surfaced previously. Software also tends to be frequently changed and "evolves" over time, but changing software without introducing errors or undesired behavior is much more difficult than building correct software in the first place. The more changes that are made to software, the more "brittle" the software becomes and the more difficult it is to make changes without introducing errors.

Thus we have the rather surprising conclusion that *as system error rates decrease and reliability increases, the risk of accidents may actually be increasing.*

Ignoring Warning Signs

Warning signs almost always occur before major accidents. For example, all the aircraft accidents considered had had precursors but priority was not placed on fixing the causal factors before they reoccurred. For *Challenger*, warning signs existed but were ignored and concerned engineers were unable to draw attention to the O-ring problems. In several of the other spacecraft accidents, there were warning signs that the software was flawed but they went unheeded. Engineers noticed the problems with the Titan/Centaur software after it was delivered to the launch site, but nobody seemed to take them seriously. The problems experienced with the MCO software during the early stages of the flight also did not seem to raise any red flags.

II. Ineffective Organizational Structure and Communication

The Ariane 5 report was strangely silent about organizational and communication issues. However, it does include a recommendation to " consider a more transparent organization of the cooperation among partners" and concludes that "close engineering cooperation, with clear cut authority and responsibility, is needed to achieve system coherence, with simple and clear interfaces between partners." No other information is provided so little can be learned about organizational factors from this accident. The other accident reports, however, all described classic management factors related to accidents.

Diffusion of Responsibility and Authority

In many of the accidents, there appeared to be serious organizational and communication problems among the geographically dispersed partners. Responsibility was diffused without complete coverage and without complete understanding by anyone about what all the groups were doing. Roles were not clearly allocated. Both the Titan and NASA Mars '98 programs were transitioning to process "insight" from process "oversight." Just as the MPL reports noted that "Faster, Better, Cheaper" was not defined adequately to ensure that it meant more than simply cutting budgets, this change in management role seems to have been implemented as a reduction in personnel and oversight responsibility without assurance that anyone was responsible for specific necessary tasks.

The MCO report concludes that project leadership did not instill the necessary sense of authority and accountability in workers that would have spurred them to broadcast problems they detected so that those problems might be "articulated, interpreted, and elevated to the highest appropriate level, until resolved." The Titan accident also shows some of these same symptoms.

Inadequate transition from development to operations played a role in several of the accidents. Engineering management sometimes has a tendency to focus on development and to put less effort into planning the operational phase of any project or system. The operations teams (in those accidents that involved operations) also seemed isolated from the developers. The MCO report

notes this isolation and provides as an example that the operators did not know until long after launch that the spacecraft sent down tracking data that could have been compared with the ground data, which might have identified the software error while it could have been fixed. The operations crew for the Titan/Centaur also did not detect the obvious problems, partly because of a lack of the knowledge required to detect them. Better communications and involvement of the developers in the launch operations might have avoided the losses.

Most important, responsibility for safety does not seem to have been clearly defined outside of the quality assurance function in any of these programs. As noted in the Challenger accident report, safety was originally identified as a separate responsibility by the Air Force during the ballistic missile programs of the 50s and 60s to solve exactly the problems seen in these accidents—to make sure that safety is given due consideration in decisions involving conflicting pressures and that safety issues are visible at all levels of decision making. Having an effective safety program cannot prevent errors of judgment in balancing conflicting safety, schedule, and budget constraints, but it can at least make sure that decisions are informed and that safety is given due consideration. It also ensures that someone is focusing attention on what the system is not supposed to do, i.e., the hazards, and not just on what it is supposed to do. Both perspectives are necessary if safety is to be optimized. Placing safety *only* under the assurance umbrella instead of treating it as a central engineering concern is not going to be effective, as has been continually demonstrated by these and other accidents.

Having an effective safety program cannot prevent errors in judgement in balancing conflicting requirements of risk, schedule, and cost, but it can at least make sure that decisions are informed and that risk is given due consideration.

Low-Level Status and Inappropriate Organizational Placement of the Safety Program

The reports on the recent accidents involving NASA projects are surprisingly silent about their safety programs. One would think that the safety activities and why they had been ineffective would figure prominently in the reports. In fact, the only time system safety is mentioned is with respect to quality assurance. Is system safety engineering being performed at all on these projects? Perhaps it is only considered necessary for missions involving humans or safety is being confused with reliability. The MPL and MCO accident reports both lament the lack of "what-if" analysis, which is the hallmark of system safety engineering. It is very effective in preventing general losses, not just those involving human life.

More information is needed to determine why this type of analysis is not being used. System safety engineering techniques may not have been used on these projects or they may have been ineffective and system safety marginalized by being limited to the quality assurance program: While safety is certainly one property among many that needs to be assured, it cannot be engineered into a design through after-the-fact assurance activities alone.

Limited Communication Channels and Poor Information Flow

In the Titan, Challenger, and Mars Climate Orbiter accidents, there was evidence that a problem existed before the loss occurred, but there was no communication channel established for getting the information to decision makers and to those who could understand it or, alternatively, the problem-reporting channel was ineffective or unused.

All the accidents involved one engineering group not getting the information they needed from another engineering group. The MCO report cited deficiencies in communication between the project development team and the operations team. The MPL report noted inadequate peer com-

munication and a breakdown in intergroup communication. The Titan accident also involved critical information not getting to those who could use it. For example, the tests right before launch detected the zero roll rate but there was no communication channel established for getting that information to those who could understand it.

In addition, system engineering on several of the projects did not keep abreast of test results from all areas and communicates their findings to other areas of the development project: Establishing and implementing this type of intergroup technical communication is one of the primary roles for system engineering.

Communication is one of the most important functions in any large, geographically distributed engineering project and must be carefully planned and fostered.

III. Ineffective or Inadequate Technical Activities

Although the actual technical errors made were different in each of the accidents, common flaws in the engineering activities led to these errors.

Inadequate System Engineering

For any project as complex as those involved in these accidents, good system engineering is essential for success. In some of the accidents, system engineering resources were insufficient to meet the needs of the project. In others, the process followed was flawed, such as in the flowdown of system requirements to software requirements or in the coordination and communication among project partners and teams. In the Titan project, there appeared to be nobody in charge of the entire process, i.e., nobody responsible for understanding, designing, documenting, controlling configuration, and ensuring proper execution of the process.

Preventing system accidents falls into the province of system engineering—those building individual components have little control over events arising from dysfunctional interactions among components. As the systems we build become more complex (much of that complexity being made possible by the use of computers), system engineering will play an increasingly important role in the engineering effort. In turn, system engineering will need new modeling and analysis tools that can handle the complexity inherent in the systems we are building. Appropriate modeling methodologies will have to include software, hardware and human components of systems. Such modeling and analysis techniques are currently only in their infancy.

Flawed Review Process

General problems with the way quality and mission assurance are practiced were mentioned in several of the reports. QA often becomes an ineffective activity that is limited simply to checking that the appropriate documents are produced without verifying the quality of the contents. The Titan accident report makes this point particularly strongly.

General review processes (outside of QA) are also described as flawed in the reports but few details are provided to understand the problems. The Ariane 5 report states that reviews included all the major partners in the Ariane 5 program, but no information is provided about what types of reviews were held or why they were unsuccessful in detecting the problems. The MCO report recommends that NASA "conduct more rigorous, in-depth reviews of the contractor's and the team's work," which it states were lacking for the MCO, Consistent with the report's emphasis on operations, it concludes that "the operations team could have benefited from independent peer reviews to validate their navigation analysis technique and to provide independent oversight of the

trajectory analysis." There is no mention, however, of software quality assurance activities or the software review process.

In the MPL case, reviews were held and the two software errors could have been caught there. Those apparently attending the review of the descent engine control software did not include anyone familiar with the potential for the spurious Hall Effect sensor signals. In general, software is difficult to review and the success of such an effort is greatly dependent on the quality of the specifications. However, identifying *unsafe* behavior, i.e., the things that the software should *not* do and concentrating on that for at least part of the review process, helps to focus the review and to ensure that critical issues are adequately considered. The fact that specifications usually include only what the software should do and omit what it should *not* do makes this type of review even more important and effective in finding serious problems. Such unsafe (or mission-critical) behavior will be (or should be) identified in the system engineering process before software development begins. The design rationale and design features used to prevent the unsafe behavior should have been documented and can be the focus of such a review. This presupposes, of course, a system safety process to provide the information, which does not appear to have existed for these projects.

None of the reports but Titan mentions any independent verification and validation (IV&V) review process (beyond normal system testing) by a group other than the developers. The Titan program did have an independent IV&V process, but it used only default values for the filter rate constants and never validated the actual constants used in flight.

Inadequate Specifications

Several of the reports refer to inadequate specification practices. The Ariane accident report refers in several places to inadequate specification practices and notes that the structure of the documentation obscured the ability to review the critical design decisions and their underlying rationale. Inadequate documentation of design rationale to allow effective review of design decisions is a very common problem in system and software specifications [26]. The Ariane report recommends that justification documents be given the same attention as code and that techniques for keeping code and its justifications consistent be improved.

The MCO report does not mention anything about specifications (or other system and software development artifacts) but clearly good specifications might have helped in educating the operators in the areas where their lack of knowledge about the engineering features prevented them from noticing problems or taking appropriate action. Also, poor specifications may have led to the use of the wrong units in the software.

The MPL report notes that the system-level requirements document did not specifically state the failure modes the requirement was protecting against (in this case possible transients) and speculates that the software designers or one of the reviewers might have discovered the missing requirement if they had been aware of the rationale underlying the requirements. The small part of the requirements specification shown in the accident report (which may very well be misleading) seems to avoid all mention of what the software should not do. In fact, some standards and industry practices even forbid such negative requirements statements. The result is that software specifications often describe nominal behavior well but are very incomplete with respect to required software behavior under off-nominal conditions and rarely describe what the software is *not* supposed to do. Most safety-related requirements and design constraints are best described using such negative requirements or design constraints.

In general, the vast majority of software-related accidents can be traced to flawed requirements rather than coding errors. Either (1) the requirements are incomplete or contain wrong assumptions about the operation of the controlled system or the required operation of the computer or (2) there

are unhandled controlled-system states and environmental conditions. This experiential evidence leads directly to a need for better specification review and analysis—the system and software specifications must be reviewable and easily understood by a wide range of engineering specialists.

Complete and understandable specifications are not only necessary for development, but they are critical for operations and the handoff between developers, maintainers, and operators. In the Titan accident, nobody other than the control dynamics engineers who designed the roll rate constants understood their use or the impact of filtering the roll rate to zero. When discrepancies were discovered right before launch, nobody understood them. The MCO operations staff also clearly had inadequate understanding of the automation and therefore were unable to effectively monitor its operation. Good specifications that include requirements tracing and design rationale are critical for long-lived systems and may be able to improve the effectiveness of the operations personnel [26].

The Titan report stressed the lack of a different type of specification: formal documentation of the overall process flow. The Centaur software process was developed early in the Titan/Centaur program and many of the individuals who designed the original process are no longer involved in it due to corporate mergers and restructuring and the maturation and completion of the Titan/Centaur design and development. Much of the system and process history was lost with their departure and therefore nobody knew enough about the overall process to detect that it omitted any testing with the actual load tape or knew that the LMA test facilities had the capability of running the type of test that could have caught the error.

Violation of Basic Safety Engineering Practices in the Digital Parts of the System Design

Although system safety engineering textbooks and standards include principles for safe design, software engineers are almost never taught them. As a result, software often does not incorporate basic safe design principles, for example, separating critical functions, eliminating unnecessary functionality, and designing error-reporting messages such that they cannot be confused with critical data (see the Ariane accident), and reasonableness checking of inputs and internal states. Consider the MPL loss: The JPL report on the accident states that the software designers did not include any mechanisms to protect against transient sensor signals nor did they think they had to test for transient conditions and they also apparently did not include a check of the current altitude before turning off the descent engines. Runtime reasonableness and other types of checks should be part of the design criteria used for any real-time software.

Another basic design principle for mission-critical software is that unnecessary code or functions should be eliminated or separated from mission-critical code and its processing. In the case of MPL, code was executing when it was not needed. The same was true for Ariane and for Titan. I am sure that each of these decisions was considered carefully, but the tradeoffs may not have been made in an optimal way and risk may have been discounted with respect to other properties.

Inadequate Software Engineering

As a person trained in software engineering, I found the reports very frustrating in their lack of investigation of the practices that led to the introduction of the software flaws and the related lack of recommendations to fix them. In some cases, software processes were declared to be adequate when all evidence pointed to the fact that they were not. Only the Titan investigation board and to a lesser extent the Ariane 5 investigators looked at the software in enough detail to determine the conditions and systemic factors related to the software engineering process (or at least included this information in the accident report).

Not surprising, the interfaces were a source of problems. It seems likely from the evidence in several of the accidents that the interface documentation practices were flawed. The MPL report includes a recommendation that in the future "all hardware inputs to the software must be identified ... The character of the inputs must be documented in a set of system-level requirements." This information is usually included in the standard interface specifications, and it is surprising that it was not. In the case of MCO, either the MCO programmers were incompetent (unlikely) or there is a more likely explanation for the units error such as the common practice of writing interface and other specifications after coding has begun, poorly designed specifications that make retrieving information error prone, software programmers without the necessary science background to understand the importance of the units used, inadequate information provided to the programmers for them to understand the role of the data in the larger system, etc.

There also appear to be problems in software reviews and perhaps also in implementing the special practices unique to real-time, embedded software. Why were unhandled cases not detected? Testing only against the requirements specification will not uncover errors in the specification. What type of requirements validation was performed? Why were defensive programming and exception-handling practices not used more effectively?

Without further investigation, we cannot learn enough about the software engineering practices involved in these accidents to prevent future reoccurrences. That is very unfortunate.

Flawed or Inadequate Analysis of Software Functions

Limitations in how thoroughly software can be tested make simulation and other types of analysis critical for software-intensive systems. The two identified MPL software errors involved incomplete handling of software states and are both examples of very common specification flaws and logic omissions. As such, they were potentially detectable by formal and informal analysis and review techniques. Software hazard analysis and requirements analysis techniques exist (and more should be developed) to detect this type of incompleteness.

The Ariane report also says that the limitations of the inertial reference system software were not fully analyzed in reviews, and it was not realized that the test coverage was inadequate to expose such limitations. The assumption by the Ariane developers that it was not possible to perform a complete system integration test made simulation and analysis even more important, including analysis of the assumptions underlying any simulation. Executable and easy to read formal requirements specifications should theoretically be able to help here. Unfortunately, most of the formal specification languages that have been proposed use obscure and obtuse notations that are inappropriate for most industry projects.

Software Reuse without Appropriate Safety Analysis

Reuse and the use of commercial off-the-shelf software (COTS) is common practice today in embedded software development. It is widely believed that because software has executed safely in other applications, it will be safe in the new one. This misconception arises from a confusion between software reliability and safety. Most accidents involve software that is doing exactly what it was designed to do, but the designers misunderstood what behavior was required and would be safe.

The blackbox (externally visible) behavior of a component can only be determined to be safe by analyzing its effects on the system in which it will be operating, that is, by considering the specific operational context. The fact that software has been used safely in another environment provides no information about its safety in the current one. In fact, reused software is probably less safe because the original decisions about the required software behavior were made for a different

system design and were based on different environmental assumptions. *Changing the environment in which the software operates makes all previous usage experience with the software irrelevant for determining safety.* The same software flaw that led to overdosing five patients by the Therac-25 existed in the Therac-25 software (which was reused on the Therac-25), but the flaw had no noticeable effects within the different Therac-20 system design.

A reasonable conclusion to be drawn is not that software cannot be reused, but that a safety analysis of its operation in the new system context is mandatory: Testing alone is not adequate to accomplish this goal. For complex designs, the safety analysis required stretches the limits of current technology. For this analysis to be technically and financially feasible, reused software must contain only the features necessary to perform critical functions. As noted earlier, both the Ariane 5 and the Titan software contained unnecessary functions that led to the losses and the MPL loss involved a necessary function operating when it was not necessary. Note, however, that COTS software is often constructed with as many features as possible to make it commercially useful in a variety of systems. Thus there is a tension between using COTS and being able to perform a safety analysis or have confidence in the safety of the system. This tension must be resolved in management decisions about risk—ignoring it only leads to accidents and potential losses that are greater than the additional cost of designing and building new components instead of buying them.

If reuse and the use of COTS software are to result in acceptable risk, then system and software modeling and analysis techniques must be used to perform the necessary safety analyses. This process is not easy or cheap. Introducing computers does not preclude the need for good engineering practices, and it almost always involves higher costs, despite the common myth that introducing automation saves money.

Inadequate System Safety Engineering

Judging only from the information (or lack of information) included in the accident reports, none of these projects appear to have involved adequate system safety engineering.

For example, several of the reports recommend reconsidering the definition they used of critical components, particularly for software. Unfortunately, not enough information is given about how the criticality analyses were performed (or even *if* they were done) to determine why they were unsuccessful. Common practice throughout engineering, however, is to apply the same techniques and approaches that were used for electromechanical systems (e.g., FMEA and FMECA) to the new software-intensive systems. This approach will be limited because the contribution of software to accidents, as noted previously, is different than that of purely mechanical or electronic components.

There appear to be several instances of flawed risk tradeoff decisions. For example, in the Ariane accident, there was a lack of effective analysis to determine which variables should be protected during execution. Unfortunately, the accident reports describe flawed decisions, but not the process for arriving at them. Important information that is missing includes how the analyses and trade studies were performed and what additional information or additional analysis techniques could have allowed better decisions to be made.

Providing the information needed to make safety-related engineering decisions is the major contribution of system safety techniques to engineering. It has been estimated that 70-90% of the safety-related decisions in an engineering project are made during the early concept development stage [11]. When hazard analyses are not performed, are done only after the fact (for example, as a part of quality or mission assurance), or are done but the information is never integrated into the engineering decision-making environment, they can have no effect on these decisions and the safety effort reduces to a cosmetic and perfunctory role.

The Titan accident provides an example of what happens when such analysis is not done. The

risk analysis, in that case, was not based on determining the steps critical to mission success (a traditional hazard analysis) but instead considered only the problems that had occurred in previous launches. Software constant generation was considered to be low risk because there had been no previous problems with it. Not only is such an approach inadequate for complex systems in general, but considering only the specific events and conditions occurring in past accidents is not going to be effective when new technology is introduced into a system. Computers are, in fact, introduced in order to make radical changes in functionality and design. In addition, software is often used precisely because it is (seemingly) easy to change for each mission and throughout operations—the system being flown today is often not the same one that existed yesterday. Proper hazard analysis that examines all the ways the system components (including software) or their interaction can contribute to accidents needs to be performed and used in decision making.

At the same time, system-safety techniques, like other engineering techniques, need to be expanded to include software and the complex cognitive decision making and new roles played by human operators [24]. Existing approaches need to be applied, and new and better ones developed. When appropriately modified system safety techniques have been used, they have been successful. If system hazard analysis is performed prior to software implementation (not just prior to testing, as is recommended in the MPL report), requirements can be analyzed for hazardous states and protection against potentially hazardous behavior designed into the logic from the beginning.

The MCO report and the Challenger report were the only ones to recommend instituting a classic system safety engineering program, i.e., continually performing the system hazard analyses necessary to explicitly identify mission risks and communicating these risks to all segments of the project team and institutional management; vigorously working to make tradeoff decisions that mitigate these risks in order to maximize the likelihood of mission success; and regularly communicating the progress of the risk mitigation plans and tradeoffs to project, program, and institutional management. The other reports, when changes were suggested, instead described classic reliability engineering approaches that are unlikely to be effective for system accidents or software-intensive systems.

One of the benefits of using system-safety engineering processes is simply that someone becomes responsible for ensuring that particular hazardous behaviors are eliminated if possible or their likelihood reduced and their effects mitigated in the design. Almost all attention during development is focused on what the system and software are supposed to do. A system safety engineer or software safety engineer is responsible for ensuring that adequate attention is also paid to what the system and software are *not* supposed to do and for verifying that hazardous behavior will not occur. It is this unique focus that has made the difference in systems where system safety engineering successfully identified problems not found by the other engineering processes.

Unnecessary Complexity and Software Functions

One of the most basic concepts in engineering critical systems is to "keep it simple." The seemingly unlimited ability of software to implement desirable features often, as in the case of most of the accidents examined in this paper, pushes this basic principle into the background: *Creeping featurism* is a common problem. As stated, the Ariane and Titan accidents clearly involved software that was not needed, but surprisingly the decision to put in or to keep (in the case of reuse) these unneeded features was not questioned in the accident reports. The MPL accident involved software that was executing when it was not necessary to execute. In the case of the Titan IVB-32, the report explains that the filter was not needed but was kept in for "consistency." The exact same words are used for the Ariane software. Neither report explains why consistency was assigned such high priority. In all these projects, tradeoffs were obviously not considered adequately, perhaps

partially due to complacency about software risk.

The more features included in software and the greater the resulting complexity (both software complexity and system complexity), the harder and more expensive it is to test, to provide assurance through reviews and analysis, to maintain, and to reuse in the future. Engineers need to start making these hard decisions about functionality with a realistic appreciation of their effect on development cost and eventual system safety and system reliability.

Test and Simulation Environments that do not Match the Operational Environment

It is always dangerous to conclude that poor testing was the "cause" of an accident. After the fact, it is always easy to find a test case that would have uncovered a known error, but it is usually difficult to prove that the particular test case would have been selected beforehand, even if testing procedures were changed. By definition, the cause of an accident can always be stated as a failure to test for the condition that was determined, after the accident, to have led to the loss. However, in these accidents, there do seem to be omissions that reflect poor decisions related to testing, particular with respect to the accuracy of the simulated operational environment.

A general principle in testing aerospace systems is to *fly what you test and test what you fly.* This principle was violated in all the spacecraft accidents. The test and simulation processes must reflect the environment accurately. Although following this process is often difficult or even impossible for spacecraft, no reasonable explanation was presented in the reports for some of the omissions in the testing for these systems. An example is the use of Ariane 4 trajectory data in the specifications and simulations of the Ariane 5 software when the Ariane 5 trajectory was known to be different.

In both the Ariane 5 and Mars '98 projects, a conclusion was reached that the components implicated in the accidents could not be tested and simulation was substituted. After the fact, it was determined that such testing was indeed possible and would have had the ability to detect the design flaws. The same occurred with the Titan accident, where default and simulated values were used in system testing although the real roll rate filter constants could have been used. Like Ariane, the engineers incorrectly thought the rigid-body simulation of the vehicle would not exercise the filters sufficiently. Even the tests performed on the Titan right before launch (because anomalies had been detected) used default values and thus were unsuccessful in detecting the error. After wiring errors were discovered in the MPL testing process, for undisclosed reasons the tests necessary to detect the software flaw were not rerun.

Better system testing practices are needed for components containing software (almost everything these days), more accurate simulated environments need to be used in software testing, and the assumptions used in testing and simulations need to be carefully checked. Another common problem with testing, particularly software testing, is inadequate emphasis on off-nominal and stress testing.

Deficiencies in Safety-Related Information Collection and Use

Researchers have found that the second most important factor in the success of any safety program (after top management concern) is the quality of the hazard information system. Both collection of critical information as well as dissemination to the appropriate people for action is required. The Challenger report noted that what had once been an outstanding hazard information system had broken down amid cost-cutting and complacency. The situation at NASA today does not appear to be any different. The MCO report concludes that lack of discipline in reporting problems and insufficient followup was at the heart of the mission's navigation mishap. In the Titan mishap, the use of voice mail and email implies there either was no formal anomaly reporting and tracking

system (none is mentioned in the report) or the formal reporting procedure was not known or used by the process participants for some reason. The report states that there was confusion and uncertainty as to how the roll rate anomalies should be reported, analyzed, documented and tracked because it was a "concern" and not a "deviation." There is no explanation of these terms.

In all the spacecraft accidents, the existing formal anomaly reporting system was bypassed (in Ariane 5, there is no information about whether one existed) and informal email and voice mail was substituted. The problem is clear but not the cause, which was not included in the reports and perhaps not investigated. When a structured process exists and is not used, there is usually a reason. Some possible explanations may be that the system is difficult or unwieldy to use or it involves too much overhead. Such systems may not be changing as new technology changes the way engineers work. There is no reason why reporting something within the problem-reporting system should be much more cumbersome than adding an additional recipient to the email. The Raytheon CAATS (Canadian Automated Air Traffic System) project implemented an informal email process for reporting anomalies and safety concerns or issues that reportedly was highly successful [12].

Operational Personnel not Understanding the Automation

Neither the MPL nor the Titan mission operations personnel understood the system or software well enough to interpret the data they saw as indicating there was a problem in time to prevent the loss. Complexity in the automation combined with poor documentation and training procedures are contributing to this problem, which is becoming a common factor in aircraft accidents.

4.2.2 Systemic Factors in the Aircraft Accidents

The aircraft accident reports differed significantly from those for the spacecraft losses. They were very detailed in what they studied but the focus was almost entirely on the physical evidence and on what the pilots had done to cause or contribute to the accident. Therefore, the range of systemic factors identified is much narrower than for the spacecraft accidents.

I. Flaws in the Safety Culture

Overconfidence in Automation

In the aircraft accidents, overconfidence in automation played a slightly different role in the events than in the spacecraft losses. It both encouraged engineers to trust software over humans and put final authority in the computer rather than the pilot, and it encouraged pilots to trust their computer-based decision aids beyond the point where they should have.

For example, the A320 autopilot does not automatically disconnect if the pilot applies force to the control wheel. One of the recommendations in the Nagoya accident report is that Airbus consider design changes allowing autopilot disconnect and manual override functions by which crews can safely control the aircraft, regardless of flight altitude or phase, by applying a force exceeding a certain level on the control column. The latter is the approach used in Boeing aircraft. A similar factor in the Warsaw accident involved the inability of the pilots to override the interlocks and operate the ground spoilers and thrust reversers because the computer did not think the airplane was on the ground.

The problems are not limited to the A320, however. The Cali accident report noted that American Airlines, as with other airlines operating FSM-equipped aircraft, trained its pilots to rely on the FMS. The assumption is made that any potential problems will result from an electrical system anomaly (not the FMS itself) and flightcrews have been taught that FMS failures will be

total. In the Cali accident, the FMS and electronic displays were functioning, and the appropriate assumption (according to their training) was that difficulty in interacting with it was due to pilot input and not something related to the FMS.

Research has shown that operators of highly reliable automated systems (such as the FMS) will increase their use of and reliance on the automation as their trust in the system increases. In addition, pilots who are accustomed to relying exclusively on FMS-generated displays for navigation can, over time, fail to recognize the relative proximity of terrain and can lose the ability to determine quickly that a fix or beacon is behind them.

At the same time, merely informing crews of the hazards of overreliance on automation and advising them to turn off the automation is insufficient and may not affect pilot procedures when it is most needed. In a previous accident involving an FMS-equipped aircraft, the flightcrew of a Thai Airways Airbus A-310 that crashed into the side of a mountain while on approach to Katmandu, Nepal, lost awareness of terrain and of the location of navaids that were in reality behind the aircraft. The incorrect pilot behavior, in that case, was very similar to that of the Cali pilots: in both accidents, pilots of sophisticated glass cockpit aircraft on approach in mountainous environments relied on the FMS and continued to interact with it in futile efforts to gain situation awareness, at the expense of reference to charts that could have enhanced terrain awareness. American Airlines uses the Katmandu accident report to inform its crews of the hazards of overreliance on automation, but that obviously was not effective in this case. There is no evidence in the report that any steps were taken by the automation designers or by anyone else to prevent its reoccurrence except to give the pilots a copy of the previous accident report.

The European Joint Aviation Authorities' Future Aviation Safety Team has identified "crew reliance on automation" as the top potential safety risk in future aircraft [19].

Slow Understanding of the Problems Associated with Human–Automation Mismatch within the Aviation Industry

Commercial aviation is the first industry where shared control of safety-critical functions between humans and computers has been widely implemented. The very difficult problems that result, such as those associated with mode confusion and deficiencies in situational awareness, are slow to be recognized and acknowledged. It is more common to simply blame the pilot for the accident than to investigate the aspects of system design that may have led to the human error(s). The Cali accident report, while being the most overtly focused on blaming the pilots, was also the most complete of the three in examining the human-automation issues. Perhaps the two factors are related in this case—the in-depth discussion in the report of the human–computer automation factors tended to excuse the pilots and the authors of the report wanted to ensure that the reader did not come to the "wrong" conclusion.

Incorrect Prioritization of Changes to the Automation

Two of the three aircraft accidents involved problems for which software fixes had been created but for various reasons had not been installed on the specific airplane involved. The reasons for this omission are complicated, and sometimes involved politics and marketing (and their combination) as much as complacency or cost factors. It is not unusual for airlines not to incorporate automation changes even when the change was prompted by a previous accident linked to that automation feature unless the change is required by their civil authorities. Optional changes may be delayed or never incorporated.

Ignoring Warnings and Near Misses

Each of the aircraft accidents involved factors that had contributed to prior incidents, but the responses had not been adequate to prevent the future loss. For example, although there had been several serious prior incidents related to the factors involved in the Nagoya accident, the modification of the FMS was only recommended (not mandatory) and France did not issue an airworthiness directive (AD) to alert airlines to the flaw in the automation. In addition, the crew were not informed of previous similar incidents within China Airline and elsewhere. In the Cali accident, the response of the airline was simply to provide the pilots with information about previous incidents and accidents and to tell them not to make the same mistake, a much less effective response than changing the confusing or misleading automation design.

II. Ineffective Organizational Structure and Communication

Limited Communication Channels and Poor Information Flow

In the aircraft accidents, the nature of the reports limited any investigation of communication problems to those between pilots and controllers, to the dissemination of previous accident information to pilots, and to the distribution of operational information such as weather reports. All of these factors were important in the accidents. For example, in the Nagoya accident, the captain and first officer had not been informed of previous similar incidents within China Airlines and elsewhere that led to the changes implemented (but not yet installed) in the service bulletin. The French authorities also did not issue an airworthiness directive promptly to alert airlines to the design flaw in the automation. The Cali accident involved several types of communication problems between the pilots and the controller.

Communication problems between the crew members (part of what is called cockpit resource management) was cited in all the accident reports.

III. Ineffective or Inadequate Technical Activities

Conflicting and Inadequate Documentation

The aircraft accidents focused on operations and thus the only specifications discussed were those provided to the pilots about the automation. However, all three aircraft accident reports noted inadequate, conflicting, or poorly designed documentation of the automatic flight systems.

In the Warsaw accident, the Aircraft Operations Manual contained conflicting or inadequate information about critical factors, such as the appropriate speed to use to counter windshear. In addition, some information was missing, such as how to compute increased landing distance under various landing speeds (including those recommended in the AOM for windshear). The Nagoya accident report cites unclear descriptions of the Automatic Flight System in the Flight Crew Operating Manual prepared by the aircraft manufacturer. With respect to the Cali accident, procedures to avoid CFIT by extracting maximum escape performance are not well developed or documented for pilots. The Boeing Operations Manual and the American Airlines Operating Manual (as well as American Airlines procedures) do not address the need to stow the speedbrakes to extract maximum performance from the aircraft during an escape maneuver.

In the Cali accident, numerous important differences existed between the display of identical data on the approach charts and on the FMS-generated displays, despite the fact that the same supplier provided both. The airline had not warned the pilots about these differences. The discrepancies were not merely the fault of the supplier but reflected inconsistencies between international standards. The lack of coordinated standards for the development and portrayal of aeronautical

charts and FMS data bases and displays led to a situation where not only were the charts and displays different in appearance, but the basic data are different. The lack of commonality is confusing, time consuming, and increases pilot workload during a critical phase of flight, i.e., the approach phase.

A second factor in the Cali accident was that the approach charts did not provide terrain information in a form that can be quickly assimilated during a high workload period. The EHSI (Electronic Horizontal Situation Indicator), which the pilots relied on for navigation, did not provide terrain information at all. Because the portrayal of flight path on the EHSI is so accurate and informative, pilots are permitted to rely on it as the primary means of navigation.

Ineffective or Inadequate Cognitive Engineering

Cognitive engineering, particularly that directed at the influence of software design on human error, is still in its early stages. Human factors experts have written extensively on the potential risks introduced by the automation capabilities of glass cockpit aircraft. Among those identified are: over reliance on automation; shifting workload by increasing it during periods of already high workload and decreasing it during periods of already low workload; being "clumsy" or difficult to use; being opaque or difficult to understand; and requiring excessive experience to gain proficiency in its use. In the Cali accident, for example, the accident report noted task saturation and overload, poor situation awareness (inadequate mental models of the automation and the situation), and distraction from appropriate behavior.

With the introduction of advanced fly-by-wire aircraft, researchers have suggested that pilots can lose awareness of the current aircraft flight mode or exhibit other forms of mode confusion.

Flightcrews not Understanding the Automation

Accidents, surveys, and simulator studies have emphasized the problems pilots are having in understanding digital automation. Not enough information is provided by the report to determine whether this occurred in the Warsaw accident, but it played a prominent role in both the Cali and Nagoya crashes. The Nagoya report concludes that the flight crew did not sufficiently understand the flight director mode changes and the autopilot override function (unclear descriptions in the operations manuals are cited as contributing to this). The Cali pilots' situation awareness was compromised by a lack of information regarding the rules of logic and priorities of the FMS navigation database. In addition, the crew may have been unaware that advancing the thrust levers in flight when the speedbrakes are deployed does not have the same effect as it does when the aircraft is landing.

These accidents demonstrate that proficiency in the use of the FMS without adequate knowledge of the logic underlying critical features, such as the design and programmed priorities of its navigation data base or autopilot override functions, can lead to its misuse. Such lack of understanding of the logic may partially account for the finding that the use of the FMS often increases workload during periods of already high workload. Numerous studies have shown that pilots are surprisingly uninformed about how the automation works [39, 3].

Problems are especially found with controls and operations the crews rarely experience in daily flight, such as mode changes and manual overrides. Either the design of the automation needs to be simplified so it is understandable or new training methods are needed for those who must deal with the clumsy, unpredictable, and inconsistent automation we are designing, or both.

Inadequate System and Safety Engineering

Standards for commercial aircraft certification still focus on component reliability and redundancy and thus are not effective against system accidents. In the aircraft accident reports, the software satisfied its specifications and did not "fail" yet it obviously contributed to the flight crews' actions and inactions.

Some of the technical inadequacies in aircraft system design stem from lack of confidence in the human and overconfidence in the automation. Even if automation is considered to be more reliable than humans, it may be a mistake not to allow flexibility in the system for emergencies and allowance for pilots to override physical interlocks, such as the inability of the pilots to operate the ground spoilers and engine thrust reversers in the Warsaw A-320 accident. Although there may be good reasons for making this decision, we are just not at the point where we can have high confidence software will do the right thing for every possible condition that can arise, and trusting software above operators (even though operators make mistakes too) may be unjustified at this point in time.

At the same time, some accident reports (such as Nagoya) mention the lack of automated protection against or nonalerting of the pilots to unsafe states (such as out-of-trim conditions). In the Cali accident, the pilots were not alerted to the fact that the speedbrakes were extended although such an alert is provided under other flight conditions.

These accidents all fit the definition of system accidents where the components acted either correctly or, as with the humans, within reasonable expectation given the context, but the interactions among the components led to an accident. Again, better modeling and analysis tools would help the system engineers and human-interface designers. These tools must include digital and human components in the models and analyses.

Inadequate Design of Feedback to Pilot

Many of the problems found in human–automation interaction lie in the human not getting appropriate feedback to monitor the automation and to make informed decisions. For example, the Nagoya accident pilots did not get adequate warning and information about the trimmable horizontal stabilizer movement when it threatened the control of the aircraft. The Cali accident report notes the lack of terrain information on the electronic situation indicator, the removal of navaids behind the aircraft from the display, and the lack of cues about the extension of the speedbrakes.

4.2.3 Summary

Except for the Titan and Ariane accidents, the spacecraft reports focused most of their attention on management, communication, and to a lesser extent culture problems. The aircraft accident reports concentrated on pilot error although the recommendations (but not the identified causes) often had a broader scope. The issues raised in all the reports are important, but not enough attention was placed on the technical and process deficiencies and, in particular, there were almost no recommendations for research on and development of new engineering methods and tools.

The culture and management problems identified in the reports are well known in the engineering and safety community and have been documented as contributing to accidents for the past 50 years. It is time we stopped recommitting the same mistakes and learned from them. System safety engineering has identified and created procedures and management techniques to reduce them, but they have not been widely applied outside the defense community.

In contrast, most of the technical factors involved in these accidents were related to new technology and the changes it is bringing to engineering processes. Here there is more need for research

and development of new techniques and tools to assist system and software engineers. The primary focus in computer science departments has been on commercial software and business environments, not the embedded, real-time, safety-critical software found in these systems: The same software engineering processes and design approaches are just not effective in both environments.

4.3 The Next Steps

While much important information for reducing risk can be gleaned from these accident reports, more is possible. Any model reflects a bias. But chain-of-events models provide too little guidance in the selection of events to include in the accident explanation and, more importantly, in the selection of conditions to investigate. We need a way to provide more scientific modeling of accidents that produces a better understanding of *why* the accident occurred and how to prevent future ones. Such an approach will have to handle system accidents and the new types of accident mechanisms that are starting to occur in software-intensive systems. Accident models based on control theory have the potential to satisfy these goals. Part II of this report describes and evaluates such models.

If a control-theoretic accident model proves to be useful in explaining accidents that have already occurred, then future steps will involve creating and evaluating new hazard analysis methods and risk assessment techniques based on the new model.

Bibliography

[1] Robert U. Ayres and Pradeep K. Rohatgi. Bhopal: Lessons for technological decision-makers. *Technology in Society*, 9:19–45, 1987.

[2] William Bogard. *The Bhopal Tragedy.* Westview Press, Boulder, Colo., 1989.

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

[4] Council for Science and Society. *The Acceptability of Risks (The Logic and Social Dynamics of Fair Decisions and Effective Controls).* Barry Rose Publishers Ltd., London, United Kingdom, 1977.

[5] Department of Employment. *The Flixborough Disaster: Report of the Court of Inquiry.* Her Majesty's Stationery Office, London, United Kingdom, 1975.

[6] Paul Eddy, Elaine Potter, and Bruce Page. *Destination Disaster.* Quandrangle/N.Y. Times Book Co., New York, 1976.

[7] M. Edwards. The design of an accident investigation procedure. *Applied Ergonomics*, 12:111–115, 1981.

[8] William Haddon, Jr. The prevention of accidents. In Duncan W. Clark and Brian MacMahon, editors, *Preventive Medicine*, page 595, Little, Brown, and Company, Boston, 1967.

[9] Willie Hammer. *Product Safety Management and Engineering.* Prentice-Hall, Inc., Englewood Cliffs, N.J., 1980.

[10] Helicopter Accident Analysis Team. newblock Final Report. newblock NASA 1998.

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

[12] Jeffrey Joyce. Personal Communication.

[13] John G. Kemeny. *Report of the President's Commission on Three Mile Island (The Need for Change: The Legacy of TMI).* U.S. Government Accounting Office, Washington,D.C., 1979.

[14] Urban Kjellen. An evaluation of safety information systems at six medium-sized and large firms. *Journal of Occupational Accidents*, 3:273–288, 1982.

[15] Urban Kjellen. Deviations and the feedback control of accidents. In Jens Rasmussen, Keith Duncan, and Jacques Leplat, editors, *New Technology and Human Error*, pages 143–156, John Wiley & Sons, New York, 1987.

[16] Trevor Kletz. *An Engineer's View of Human Error.* Institution of Chemical Engineers, Rugby, Warwickshire, U.K., 1991.

[17] John C. Knight and Nancy G. Leveson. An Experimental Evaluation of the Assumption of Independence in Multi-Version Programming. *IEEE Transactions on Software Engineering*, Vol. SE-12, No. 1, January 1986, pp. 96-109.

[18] John Ladd. Bhopal: An essay on moral responsibility and civic virtue. Department of Philosophy, Brown University, Rhode Island, January 1987.

[19] David Learmount. Flight Safety Foundation's European Aviation Safety Seminar. *Flight International*, March 20-26, 2001, p. 17.

[20] Jacques Leplat. Accidents and incidents production: Methods of analysis. In Jens Rasmussen, Keith Duncan, and Jacques Leplat, editors, *New Technology and Human Error*, pages 133–142, John Wiley & Sons, New York, 1987.

[21] Peter Lewycky. Notes toward an understanding of accident causes. *Hazard Prevention*, pages 6–8, March/April 1987.

[22] 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.

[23] Jacques Leplat. Some observations on error analysis. In Jens Rasmussen, Keith Duncan, and Jacques Leplat, editors, *New Technology and Human Error*, pages 311–316, John Wiley & Sons, New York, 1987.

[24] Nancy G. Leveson. *Safeware: System Safety and Computers.* Addison Wesley, 1985.

[25] Nancy G. Leveson. A Taxonomy and Evaluation of Accident Models. , in preparation

[26] Nancy G. Leveson. Intent Specifications: An Approach to Building Human-Centered Specifications. *IEEE Transactions on Software Engineering*, Vol. SE-26, No. 1, January 2000.

[27] Nancy Leveson et.al. An Assessment of Space Shuttle Flight Software Development Processes. National Research Council, 1993.

[28] 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.

[29] Malcolm McConnell. *Challenger: A Major Malfunction.* Doubleday and Company, Inc., Garden City, N.J., 1987.

[30] G.F. McCormick. When reach exceeds grasp. Unpublished essay.

[31] C.O. Miller. The broader lesson from the Challenger. *Hazard Prevention*, pages 5–7, January/February 1987.

[32] Elisabeth Pate-Cornell. Organizational Aspects of Engineering System Safety: The Case of Offshore Platforms. *Science*, vol. 250, November 30, 1990, pp. 1210-1217.

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

[34] Charles Perrow. The habit of courting disaster. *The Nation*, pages 346–356, October 1986.

[35] 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.

[36] Jens Rasmussen. Risk Management in a Dynamic Society. *Third Annual Symposium on Human Interaction with Complex Systems*, August 1996.

[37] William P. Rogers. *Report of the Presidential Commission on the Space Shuttle Challenger Accident.* U.S. Government Accounting Office, Washington, D.C., 1986.

[38] Veikko Rouhiainen. The quality assessment of safety analysis. Technical Report Publications 61, Technical Research Center of Finland, Espoo, Finland, 1990.

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

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

[41] Juoko Suokas. The role of management in accident prevention. In *First International Congress on Industrial Engineering and Management*, Paris, June 11–13 1986.

[42] Juoko Suokas. Evaluation of the quality of safety and risk analysis in the chemical industry. *Risk Analysis*, 8(4):581–591, 1988.