

Framing analysis of software failure with safety cases

William S. Greenwell*, John C. Knight

Department of Computer Science, University of Virginia, P.O. Box 400470, Charlottesville, VA 22904-4740

Abstract

Failures of digital systems arise from design faults that are introduced during system development or maintenance, and the complexity and tight coupling of these systems can lead to accidents involving interactions of multiple failed components. These factors complicate the analysis of digital system failures, particularly with respect to framing an analysis and issuing recommendations that are relevant and practicable. We present a systematic approach to the analysis of digital system failures based upon the concept of safety cases. Failures of safety-related digital systems typically indicate the presence of fallacies in their underlying safety arguments. Using our approach, investigators elicit evidence from a failure to discover fallacies in the safety argument that might have contributed to the failure and then develop recommendations for addressing the fallacies they discover, producing a revised safety argument. Our approach assists investigators in framing an analysis by determining what evidence should be elicited, and it ensures that investigators' recommendations address the problems that they identify. We report our results from applying the approach to a sequence of accidents involving a low-altitude warning system and compare them to the results of the official investigations. We then contrast the features of our approach with those of other methods.

Keywords: Forensic software engineering; Software failure analysis; Evidence elicitation; Safety cases

1. Introduction

Leveson defines failure in the context of system safety as “the nonperformance or inability of the system or component to perform its intended function for a specific time under specified environmental conditions” [1]. Thus, any safety-related system whose behavior deviates from what was intended has failed, even if it complies with its specification. Safety-related digital systems may generally be divided into two categories: those that control potentially hazardous operations and those that monitor for hazardous conditions. The failure of a control system might cause the system to take actions that directly compromise safety; for example, the series of radiation overdoses administered by the Therac-25 treatment apparatus that caused the deaths of six patients [1]. The failure of a monitor system could cause the system to display false information or not take the appropriate actions when a hazardous condition arises; such was the case with the Minimum Safe Altitude Warning system (MSAW), which we consider later in this paper. The severity of a failure depends upon the context in which it occurs and the range of hazards it exposes, but in many cases failures may cause or contribute to accidents. Thus, it is necessary to analyze these failures to prevent them from recurring.

In this paper we present a systematic approach to the analysis of digital system failures based upon the concept of system safety cases. The safety case for a system documents the complete argument and evidence that the system is safe to operate. Failures of safety-related digital systems typically indicate the presence of fallacies in their underlying safety arguments. Our approach, *Pandora*, elicits evidence from a failure to discover fallacies in the safety argument that might have contributed to the failure. *Pandora* documents fallacies as lessons and develops recommendations for addressing them, yielding a revised safety argument that is free of the fallacies that were discovered. We apply *Pandora* to a sequence of accidents involving the MSAW system and compare our results to those of the official investigations into the accidents.

The analysis of a digital system failure is complicated by several factors. First, failures of digital systems typically occur due to design faults [2], which may be introduced during specification, design, implementation, or

* Corresponding author. Tel.: +1 434 982 2292; fax.: +1 434 982 2216.
E-mail addresses: greenwell@cs.virginia.edu (W. Greenwell), knight@cs.virginia.edu (J. Knight).

maintenance [3]. Investigators must therefore be prepared to review a system's complete development and maintenance history in order to conduct a comprehensive analysis. Second, digital systems often implement radically new functions, and so the analysis of a new system cannot benefit much from knowledge of previous designs [2]. Finally, digital systems typically exhibit interactive complexity and tight coupling not only within their internal designs but also in their interactions with other systems. Perrow argues that such systems are prone to accidents involving "the unanticipated interaction of multiple failures" of components, which he calls *system accidents* [4]. Not only are system accidents nearly inevitable, but the interactions between failures may hinder attempts to diagnose them.

Existing software engineering techniques are not well-suited to the analysis of digital system failures. The complexity and coupling noted above can lead to accidents caused by interacting failures of multiple components; however software engineering tends to treat components in isolation because of the application of functional decomposition [5]. Debugging quickly becomes infeasible as system complexity grows [6]. More sophisticated analysis tools such as formal verification can be used to represent or confirm a theory of system failure, but they cannot attest to the completeness of that theory [7]. Model checking has been shown to be effective at detecting potentially hazardous states in complex systems, but it has not been applied to forensic analysis of failure [8].

Aside from the inapplicability of existing software engineering techniques, Johnson identifies the following additional factors that complicate the forensic analysis of digital system failures:

- The lack of an accepted stopping rule for framing the analysis;
- The assessment of the developer's intentions;
- The influence of contextual factors on development; and
- The challenge of issuing relevant and practicable recommendations [7].

Framing an analysis refers to determining which aspects of a system's development and maintenance history might have contributed to a failure as well as the evidence that would be required to confirm or exclude them as contributory. If the scope of an investigation is too shallow, it might only address superficial issues that are symptomatic of deeper, more fundamental problems. Investigators may also wish to consider the intentions behind development decisions—why those decisions were believed to enhance or not adversely affect safety—as well as contextual factors that might have influenced the developer's actions. Finally, investigators should be confident that the recommendations they issue address the problems discovered during the analysis and are feasible to implement. We present an approach to analyzing digital system failures that helps to address these issues.

This paper is organized as follows. We begin with an overview of safety cases in section 2 followed by a description of the enhanced safety-case lifecycle in section 3. We then present our approach, *Pandora*, in section 4. In section 5 we report on a case study in which we applied the approach to failures of the MSAW system mentioned earlier to compare its performance to that of official investigations. In section 6 we compare Pandora to related techniques, and we summarize our conclusions in section 7.

2. The System Safety Case

A safety case is a "documented body of evidence that provides a convincing and valid argument that a system is adequately safe for a given application in a given environment" [9]. The system safety case documents the safety requirements for a system, the evidence that the requirements have been met, and the argument linking the evidence to the requirements. Bishop and Bloomfield decompose the safety case into claims, evidence, argument, and inferences. Claims are simply propositions about properties of the system supported by evidence. Evidence may either be factual findings from prior research or scientific literature or sub-claims that are supported by lower-level arguments. The safety argument is the set of inferences between claims and evidence that leads the reader from the evidence forming the basis of the argument to the top-level claim—typically that the system is safe to operate in its intended environment. Thus, a safety argument may be viewed as a directed acyclic graph (DAG) in which the root node is the top-level claim, the internal nodes are sub-claims, and the leaves are the basis of the argument. Producing a safety case does not impose specific process requirements on the developer; rather, he is free to develop a system as he chooses provided he can submit evidence accompanied by a compelling argument that the system

meets its safety requirements. System safety is argued through satisfaction of the requirements, which are then broken down further into more specific goals that can be satisfied directly by evidence.

Graphical notations for depicting safety arguments formalize the structure of an argument—that is, the relationships between claims, evidence, and accompanying contextual information—and present it diagrammatically. Formalizing the structure of an argument also reinforces the DAG structure and makes the argument more amenable to analysis. Adelaar’s Claims-Argument-Evidence (ASCAD) and Kelly’s Goal Structuring Notation (GSN) are two popular notations [11], [12]. Tool support for constructing and analyzing safety arguments is available for both notations [13], [14].

An additional advantage of representing the structure of safety arguments formally is that successful arguments may be reused easily provided the context is compatible. This observation leads to the notion of safety-case patterns, which capture solutions to common problems in safety argumentation much as design patterns do for software development [15]. Similar to the concept of safety-case patterns, AntiPatterns “communicate weak and flawed arguments – such that they may be recognized and avoided in future developments” [16]. They are “the antithesis of patterns, but also include an approach for re-factoring the problem, which has negative consequences, into an acceptable solution” [17]. AntiPatterns can assist detection of fallacious inferences in system safety arguments, and analysis of failed systems may reveal new AntiPatterns.

3. The Enhanced Safety-Case Lifecycle

An important feedback loop exists between safety-critical system development and failure analysis. Systems are deployed with unknown faults, faults are discovered through observed failures, and lessons are obtained that are ultimately incorporated into development practices for future systems. The system safety case plays a central role in this feedback loop because it attests to the safety of a system before the system is deployed, and if the completeness of the safety case is undermined by a failure, the safety case must be repaired in order to determine what design or procedural changes should be made to prevent the failure from recurring.

Kelly and McDermid developed a process for updating the system safety case in response to a recognized challenge to its validity that they refer to as the *safety-case lifecycle* [18], [19]. Challenges may arise in the form of changes to regulatory requirements, system design changes, modified assumptions, and counter-evidence obtained from operational experience including observed failures. They assume that the challenges to a safety argument have been recognized prior to the application of their process, and they do not address the problem of deriving challenges from an observed failure. To complete their lifecycle, we added an additional step in which failure analysis is performed on the safety case to discover the fallacies in the safety argument that contributed to a failure so that the argument may then be recovered. This refined process is the *enhanced safety-case lifecycle* [9].

The enhanced safety-case lifecycle, illustrated in Figure 1, augments the core lifecycle so that the safety case guides failure analysis and serves as the medium through which lessons and recommendations from the analysis are communicated. As a safety-related system is developed, a safety argument is prepared that shows: (1) why confidence should be placed in the system’s ability to meet its safety requirements; and (2) why those requirements entail safe operation. Once the system and its safety argument are accepted, the system is put into operation, but faults may still exist in the system or in its operational or maintenance procedures if the safety argument is incomplete. If triggered, these faults may lead to a failure, in which case an analysis will be conducted to discern the nature of the failure. The failure might be *random*—within the fault probabilities specified by the safety requirements—or it might be *systemic*; that is, due to a design fault that was introduced when the system was developed or serviced. Failures arising from systemic faults must be thoroughly analyzed to prevent recurrences.

The *pre-failure safety case*—that is, the safety case for the system as it existed just prior to the failure—can guide the analysis by leading investigators through the original argument describing why the system was thought to be safe. Using the evidence obtained from the failure, investigators correct flaws in the argument as they discover them, which are documented as lessons and recommendations, and ultimately produce a revised safety case for the system called the *post-failure safety case*. Implementing the post-failure safety case may require developers to make changes to the system or its associated procedures as well as changes to development practices so that future systems will not exhibit similar failures. Through this iterative loop in which the safety case plays a central role, the system development process changes over time to reflect the accumulated experience gained from observing operational systems.

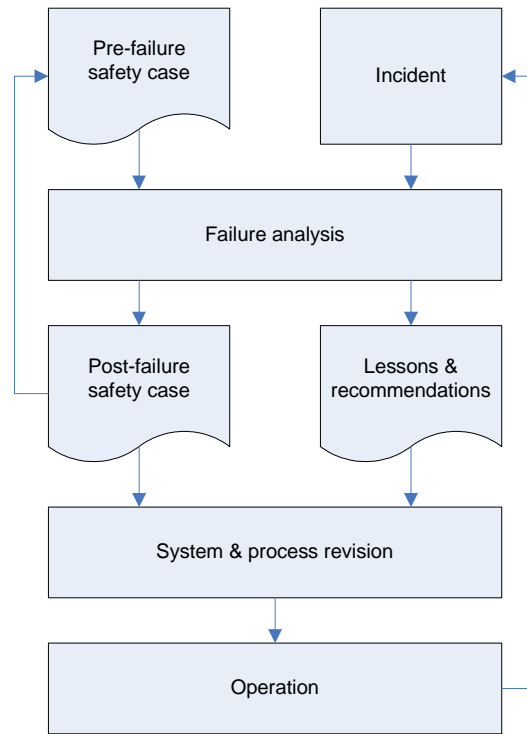


Figure 1: The enhanced safety-case lifecycle

4. Pandora

Pandora is an approach to analyzing digital system failures that is grounded in the system safety case and implements the failure analysis process specified by the enhanced safety-case lifecycle. It assumes that safety-related failures result from systemic faults, which exist in operational systems due to fallacies in their underlying safety arguments.

Based upon counter-evidence obtained from a failure, Pandora identifies fallacious inferences in a system’s pre-failure safety case that might have allowed the failure to occur. It then develops lessons and recommendations for removing the fallacies from which a post-failure safety case is produced. Through its processes of eliciting evidence, detecting fallacious inferences, and developing lessons and recommendations—all of which are centered on the system safety case—Pandora provides a systematic analysis of safety-related system failures so that greater confidence may be placed in both the rigor of investigations and the effectiveness of their findings.

Figure 2 presents the core Pandora process. Pandora begins with a pre-failure safety case and an observed system failure. For each claim in the safety argument and starting with the top-level claim, Pandora first considers the relevance of that claim’s premises. Premises that do not support the claim are not considered further. This pruning process is repeated for each of the remaining premises, leading to a depth-first recursion through the safety argument. Once the relevance of the supporting premises has been established, the premises are taken together and the sufficiency of the argument supporting the claim is evaluated. If premises are missing or if counter-evidence elicited from the failure refutes the claim, then the argument is insufficient, and Pandora invalidates the claim. When Pandora discovers an insufficient argument, it repairs the argument by either amending the premises or, if the claim is unsupported, by modifying or deleting the claim and making reparations elsewhere. It then resumes consideration of the consequent (parent) claim in the argument. In its final step, Pandora considers the sufficiency of the top-level claim of the safety argument after it has either affirmed or restored the logical validity of each of its premises.

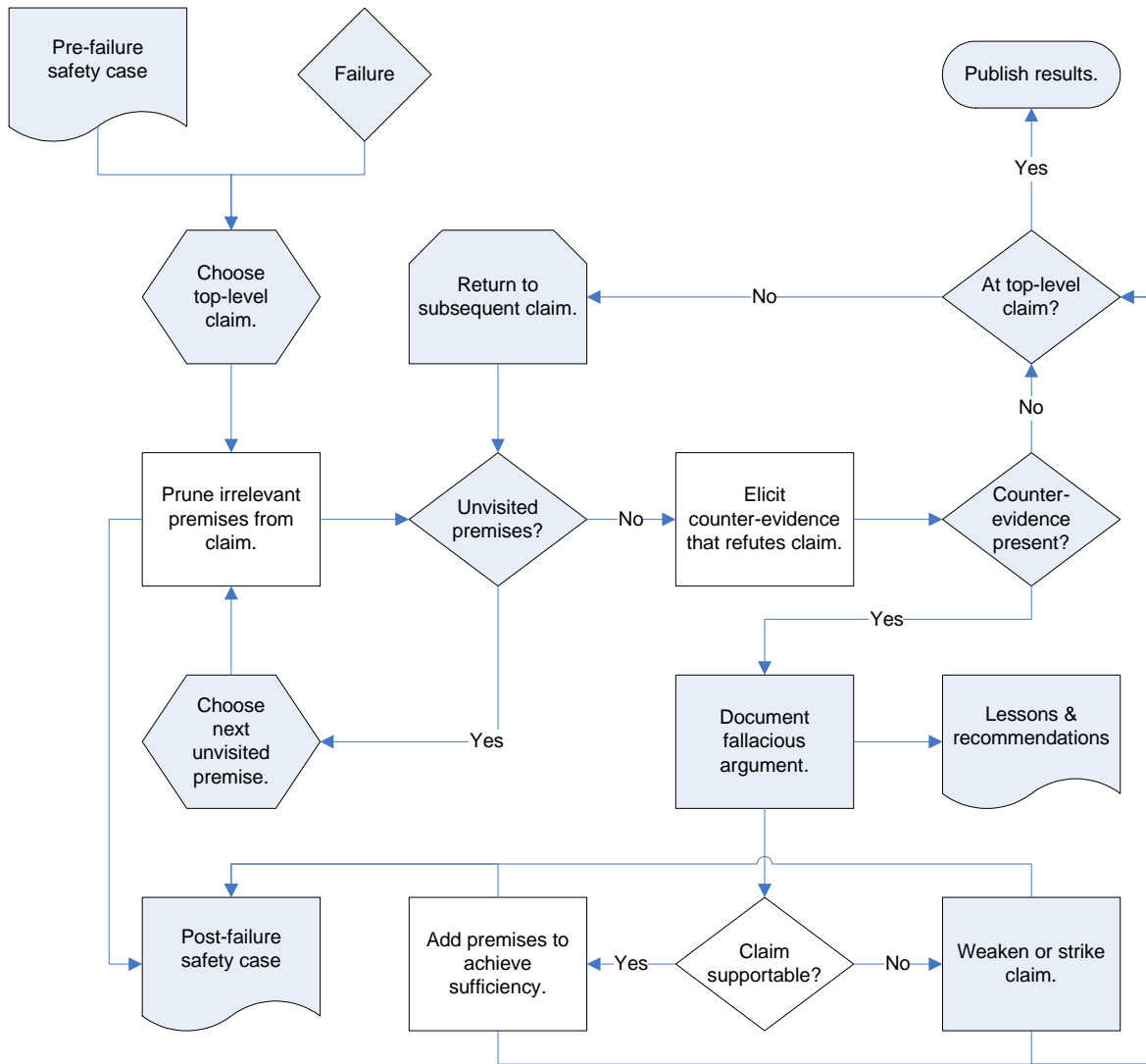


Figure 2: The Pandora failure-analysis process

Upon concluding its traversal through the safety argument, Pandora has produced a set of lessons documenting the fallacies in the pre-failure safety case that might have contributed to the failure, a set of recommendations for removing those fallacies from the argument, and the post-failure safety case: a prototype of the pre-failure case into which the recommendations have been incorporated. With some revisions, the post-failure safety case is intended to become the new operational safety case for the system. Implementing the post-failure safety case may impose new requirements on the system or otherwise entail changes to the system’s design or associated operational or maintenance procedures.

The following sections explain Pandora’s methods of detecting fallacious reasoning in safety arguments and repairing arguments when fallacies are discovered.

4.1. Detecting Fallacious Reasoning

A safety case might contain premises that are irrelevant to the claims it supports. Irrelevant premises are those from which the truth of the claim does not follow from the truth of the premise; they are sometimes referred to as non sequiturs [20]. To use an irrelevant premise in an argument is to commit a relevance fallacy. The mere existence of a relevance fallacy in a safety argument cannot contribute to a failure; rather, these fallacies might mislead a developer or reviewer into accepting an insufficient argument, which, in turn, may contribute to a system failure. Since irrelevant premises cannot themselves contribute to a failure, however, there is no need to consider them as

part of an investigation, and so Pandora prunes them from the safety argument early in the process. Even if the arguments supporting the pruned claims are further flawed, repairing them is futile because they ultimately support inconsequential conclusions. Pruning irrelevant premises from the safety argument improves the efficiency of Pandora because it reduces the amount of evidence investigators must collect.

While relevance fallacies may hinder one's ability to evaluate an argument, safety-related failures ultimately arise from insufficient arguments in a system's safety case. An argument is insufficient if it provides too little evidence in support of its claim, if the evidence is biased, or if it fails to provide evidence that would be expected for its particular claim [20]. To see why the former statement is true, consider a typical top-level claim of a safety case that a system is acceptably safe to operate in its intended context. If the system fails, and the failure is assumed to be systemic, then clearly the top-level claim has been violated. It is then the case that at least one of the premises supporting the top-level claim was not satisfied or that the premises together do not provide adequate support for the claim. The same can be said of each premise for which counter-evidence is obtained.

In the Pandora process, the investigator considers the sufficiency of the argument supporting a claim after he has established the relevance of the claim and its premises, and either affirmed or restored the validity of the arguments supporting the premises. Thus, when investigators consider the sufficiency of a particular argument within the safety case, they have already affirmed or restored the validity of its premises and their relevance to the claim. Sufficiency is evaluated in two ways: first by checking whether the argument fits a known pattern (or AntiPattern) and has omitted expected evidence; and second by eliciting counter-evidence from the failure suggesting that the claim was not satisfied despite the truth of its premises. Investigators are asked to consider, for a given claim, whether there exists evidence from the failure that the claim was unsatisfied and, if so, whether this occurred because the premises, taken together, provided inadequate support for the claim. If the argument fails either of these checks, then it is insufficient and must be repaired.

If the argument being considered fits a known pattern, then investigators can use the pattern to determine whether the argument has provided the appropriate types and volume of evidence to satisfy the claim. To support this step, we have developed a taxonomy of common safety-argument fallacies that we adapted from several general-purpose taxonomies taken from philosophical literature based on an assessment of real-world safety arguments [21]. The taxonomy, which is summarized in Table 1, organizes fallacies into categories according to the types of arguments in which they typically appear. This organization allows investigators to quickly determine the set of fallacies that might pertain to the arguments they consider. Each entry in the taxonomy consists of the name of the fallacy, a brief definition, examples of the fallacy, and in some cases a detailed discussion of how the fallacy is likely to appear in safety arguments.

Even if an argument properly instantiates a pattern or if it does not fit any known patterns, it is still possible that the argument is insufficient. To determine whether this is the case, investigators must search for counter-evidence indicating that the claim of the argument was not satisfied even though its premises were. If counter-evidence is discovered, then there must exist additional premises that were unsatisfied because a true antecedent cannot imply a false consequent. The counter-evidence may be documented as an AntiPattern showing how the argument failed to satisfy its claim, which can then be used to evaluate future arguments as part of the first sufficiency check. This second method of sufficiency evaluation is key to Pandora's ability to learn from incidents because it allows investigators to discover new forms of fallacious reasoning in safety arguments and model them as AntiPatterns to ease future detection.

4.2. Repairing the Safety Argument

Once a fallacious argument has been discovered in a safety case, the argument must be repaired. The Pandora process performs repair in two phases: it first documents the fallacious argument as a lesson and then develops a recommendation as to how the argument should be amended in order to satisfy its claim. If a claim is found to be unsupported, then the recommendation would weaken the claim or strike it altogether. Investigators using Pandora derive lessons and recommendations systematically by requiring that each lesson describe an identified fallacy in the safety case and that each recommendation address one or more lessons, greatly reducing if not eliminating the number of superfluous lessons and recommendations produced. If the lessons and recommendations are sufficiently general, they may be applied to other Pandora investigations or even to similar safety cases for systems that have not yet failed.

Table 1: The safety argument fallacy taxonomy

| | |
|---|---|
| <p>Circular Reasoning <i>Circular Argument</i> <i>Circular Definition</i></p> | <p>Anecdotal Arguments <i>Correlation Implies Causation</i> <i>Damning the Alternatives</i> <i>Destroying the Exception</i> <i>Destroying the Rule</i> <i>False Dichotomy</i></p> |
| <p>Diversions Arguments <i>Irrelevant Premise</i> <i>Verbose Argument</i></p> | <p>Omission of Key Evidence <i>Omission of Key Evidence</i> <i>Fallacious Composition</i> <i>Fallacious Division</i> <i>Ignoring Available Counter-Evidence</i> <i>Oversimplification</i></p> |
| <p>Fallacious Appeals <i>Appeal to Common Practice</i> <i>Appeal to Improper/Anonymous Authority</i> <i>Appeal to Money</i> <i>Appeal to Novelty</i> <i>Association Fallacy</i> <i>Genetic Fallacy</i></p> | <p>Linguistic Fallacies <i>Ambiguity</i> <i>Equivocation</i> <i>Suppressed Quantification</i> <i>Vacuous Explanation</i> <i>Vagueness</i></p> |
| <p>Mathematical Fallacies <i>Faith in Probability</i> <i>Gambler's Fallacy</i> <i>Insufficient Sample Size</i> <i>Pseudo-Precision</i> <i>Unrepresentative Sample</i></p> | |
| <p>Unsupported Assertions <i>Arguing from Ignorance</i> <i>Unjustified Comparison</i> <i>Unjustified Distinction</i></p> | |

A lesson in Pandora is an AntiPattern describing how an insufficient argument in the safety case either failed to provide expected evidence in support of its claim or failed to address counter-evidence elicited from the system failure. Lessons are generated whenever insufficient arguments are discovered. If the fallacy was discovered by matching the argument to an existing pattern or AntiPattern, then the lesson will show how the argument fails to meet that pattern or conforms to the AntiPattern. If it was discovered based upon counter-evidence obtained from the failure, then the lesson will be a new AntiPattern altogether. Documenting lessons as AntiPatterns helps investigators to describe precisely the nature of the fallacy that was committed and where the fallacious reasoning appears in the safety case.

From each lesson a recommendation is developed as to how the argument should be revised so that it either conforms to the pattern or addresses the counter-evidence. Just as lessons take the form of AntiPatterns, a recommendation is a pattern showing how the argument might be modified to address the fallacies identified by the lesson and restore validity. The recommendation is a suggested strategy for repairing the argument—it does not have to encompass all possible solutions nor must the system developer implement the particular strategy contained in the recommendation. A recommendation may address multiple lessons, but it should correspond to at least one lesson to prevent superfluous recommendations from being issued. If investigators find that a claim cannot be supported by adding additional premises to the argument, then they may recommend that it be weakened or stricken from the argument, which could impact the sufficiency evaluation of consequent (parent) claims.

5. MSAW Case Study

We applied Pandora to a series of accidents involving a low-altitude warning system for aircraft to compare its lessons and recommendations to those of the official investigations into the accidents. The system in question is the Minimum Safe Altitude Warning system (MSAW), which has been in use since the early 1970s. MSAW provides a good case study because it was initially developed, and has since been improved, in response to a series of accidents. We briefly describe MSAW and the accidents surrounding its development, and then we move on to discuss the

methodology of our case study, the lessons we obtained from applying Pandora, the results of the case study, and the threats to experimental validity.

5.1. Overview of the Minimum Safe Altitude Warning System (MSAW)

The Minimum Safe Altitude Warning system (MSAW) is a software system intended to help prevent controlled flight into terrain (CFIT), which typically occurs when a flight crew loses situational awareness and pilots a serviceable aircraft into terrain or an obstacle. MSAW is a function of the Automated Radar Terminal System (ARTS) family of air traffic management systems deployed by the U.S. Federal Aviation Administration (FAA) in the early 1970s and the Standard Terminal Automation Replacement System (STARS), which is replacing ARTS. The FAA developed MSAW in response to a 1972 commercial aviation accident near Miami, Florida and began to deploy the system in 1977.

MSAW alerts air traffic controllers when an aircraft descends, or is predicted by the system to descend, below a predetermined minimum safe altitude (MSA). Upon receiving a MSAW alert, a controller is required to identify the aircraft that triggered the alert and issue an advisory to the flight crew or to the controller handling the aircraft. An MSAW alert consists of a visual indication beside the aircraft's data block on a controller's radar display and an aural alarm that sounds in the control room.

MSAW detects MSA violations via two mechanisms: general terrain monitoring and approach path monitoring. General terrain monitoring applies to most aircraft that are equipped with altitude-encoding transponders and detects violations by comparing the altitude reported by an aircraft's transponder to the MSA for the aircraft's location. If the reported altitude is below the MSA, then MSAW will alert the controller. Approach path monitoring only applies to aircraft on final approach (where the risk of CFIT is greatest) and predicts MSA violations by calculating an aircraft's descent path and then comparing it to the standard descent path for the approach course. If MSAW determines the aircraft's approach to be too steep, it alerts the controller.

Each MSAW installation is customized to the local facility at which it will operate. Site adaptation variables specify the locations and length of runways and the dimensions of approach capture boxes, which MSAW uses to determine whether an aircraft is flying a final approach course. MSAW computes MSA values from a local terrain database that provides elevation data for the surrounding environment. If a MSAW installation generates excessive false alarms, areas known as *inhibit zones* may be defined temporarily to exclude problematic regions of airspace from MSAW processing until adjustments are made to other site adaptation variables.

Since its initial development, several accidents have occurred in which the failure of MSAW to alert a controller to a MSA violation was cited as a contributory factor. The National Transportation Safety Board (NTSB) investigated each of these accidents, and in section 5.3 we compare their findings to those we obtained by applying Pandora.[†] The accidents, beginning with the 1972 accident that motivated the development of MSAW, are summarized below.

- Eastern Air 401 (December 29, 1972) – Although the controller noticed the aircraft's low altitude and queried the flight crew, he concluded that the flight was in no immediate danger after receiving a positive response. This accident motivated the development of the MSAW system.
- USAir 105 (September 8, 1989) – The aircraft began a premature descent, but MSAW failed to alert the controller because the descent occurred inside an inhibit zone. In response, the FAA revised its site adaptation guidance “to minimize the extent of MSAW inhibited areas” [22].
- TAESA Learjet 25D (June 18, 1994) – The aircraft generated no MSAW alerts because the site variables, including those specifying the runway layout for the airport, were incorrect. In response, the FAA conducted a national review of MSAW site adaptation variables at all its facilities.
- Beechcraft A36 (January 29, 1995) – MSAW issued four visual alerts concerning the accident aircraft, but the controller did not notice them. This accident prompted the FAA to begin installing aural alerts at all its MSAW-equipped facilities.
- Piper PA-32-300 (October 2, 1996) – An inspection of the ATC facility involved in the accident revealed that the MSAW aural alarm speaker had been muted. Consequently, the FAA began requiring facility supervisors to inspect the speakers at the beginning of each shift.

[†] We did not participate in the NTSB's investigations of the accidents that we included in the case study.

- Korean Air 801 (August 6, 1997) – MSAW did not generate any alerts for the aircraft because an inhibit zone had been defined that encompassed almost the entire radar service area. The FAA recertified all of its MSAW installations, revised its training and site adaptation procedures, and established a centralized configuration management program for MSAW.
- Gates Learjet 25B (January 13, 1998) – The aircraft crashed east of the runway, but MSAW did not generate any alerts because the site adaptation variables for the runway’s final approach course were incorrect. This accident occurred as the FAA began revising its configuration management program for MSAW in the wake of the Korean Air 801 accident.

5.2. Methodology

We applied Pandora to each of the accidents listed above beginning with the USAir 105 accident on September 8, 1989 and ending with the Korean Air 801 accident on August 6, 1997. We excluded the Eastern Air 401 accident from the study because no safety case for MSAW existed prior to this accident, and we excluded the Gates Learjet accident because the findings of the investigation into the accident were encompassed by those from previous MSAW-related accidents. For each accident, we developed a pre-failure safety argument for MSAW based on the safety rationale for the system at the time of the accident. We then applied Pandora using the evidence obtained from the official investigation to identify flaws in the safety argument that could have contributed to the accident. Finally, we corrected the flaws Pandora identified to produce a post-failure safety argument and then compared our post-failure argument to the findings and recommendations from the official investigation.

We were unable to locate an official safety case for the MSAW system, so we derived each of the pre-failure arguments from National Transportation Safety Board’s (NTSB) final report on the Korean Air 801 accident [22]. The report contains an extensive review of MSAW and the FAA’s management of the system, and it includes an overview of MSAW, a chronology of the system’s development and the accidents surrounding it, and the findings pertaining to MSAW from the Korean Air accident. The chronology lists the lessons and recommendations from the investigations into each accident as well as the corrective actions taken by the FAA.

Attempting to construct the pre-failure safety arguments directly from the NTSB’s final report on the Korean Air 801 accident would have introduced a high risk of hindsight bias. To reduce this risk, we split the relevant sections from the report into individual statements, which yielded 289 statements. We then categorized each statement according to the earliest accident prior to which the information contained in the statement would have been known by the FAA. For example, the Beechcraft A36 accident in 1995 revealed the importance of an aural alert, so we assumed that this information was unknown prior to the accident. In general, findings from one accident investigation were assumed to be known by the FAA prior to the subsequent accident, and dated statements (i.e.; “In a May 31, 1977 letter, the FAA advised the Safety Board that...”) were assumed to be known prior to the earliest accident that occurred after the date. In some cases, however, dated statements were moved later in the timeline because their significance would not have been recognized until after a particular accident (e.g.; the modifications that were made to the MSAW system at Guam prior to the Korean Air accident).

After categorizing the statements from the report, we constructed a candidate pre-failure MSAW safety argument for the USAir 105 accident. We extracted the statements that were known prior to the accident and eliminated those that did not pertain to the safety rationale for MSAW. This process yielded nine statements, including the high-level safety requirement that MSAW “visually and aurally [alerts] a controller whenever an IFR-tracked target with an altitude encoding transponder (Mode C) descends below, or is predicted by the software to descend below, a predetermined safe altitude” [22]. The other statements discussed the monitoring and visual alerting capabilities of MSAW, controller training, and that MSAW may be inhibited to prevent nuisance alerts. These nine statements constituted the factual basis for our pre-failure safety argument.

We constructed the pre-failure safety argument in the Goal Structuring Notation (GSN) [12]. A portion of the argument is presented in Figure 3. To construct the argument, we represented the information from our basis as nodes and then inferred evidentiary relationships between the nodes. The complete argument contains 23 nodes including goals, context, and solutions. Eighteen nodes correspond directly to information contained in the factual basis of the argument. Three of the remaining nodes are goals that were added to enhance the validity of the argument (nodes G01, G06, and G07 in Figure 3), and the other two are context nodes that define unfamiliar terms (nodes C01 and C02). We inferred the top-level claim of the argument (G01) from the claims that MSAW will detect altitude violations and alert a controller (G02) and that controllers will relay MSAW alerts to flight crews (G03).

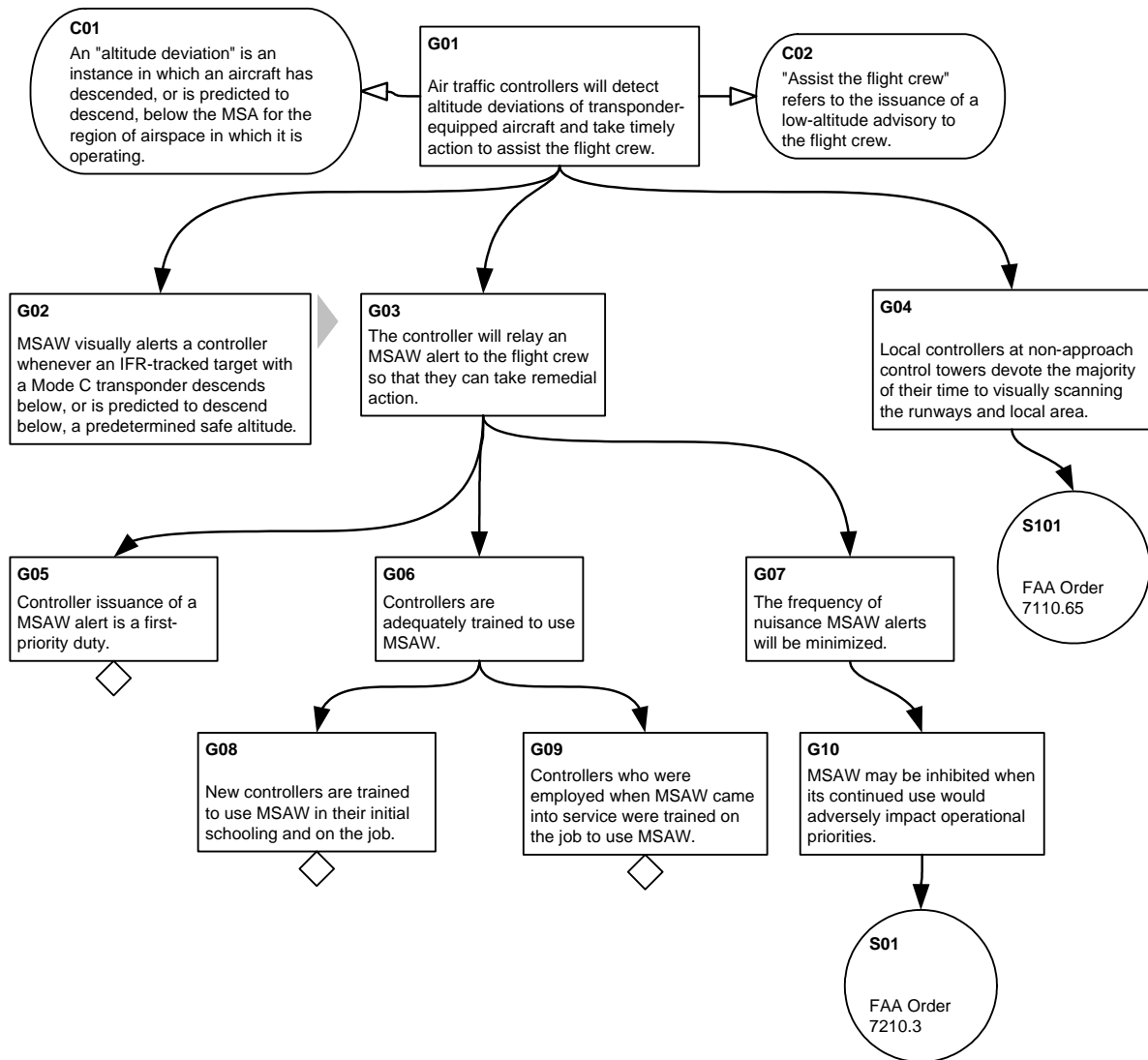


Figure 3: The top-level pre-failure MSAW safety argument for the USAir 105 accident

Using the USAir 105 argument as a baseline, we constructed pre-failure arguments for each of the subsequent accidents by adding the additional information that was known prior to each accident. By the time of the Korean Air accident in August 1997, the FAA had made significant changes to MSAW and its configuration management program for the system, so we rebuilt the pre-failure argument from the basis as we did for USAir 105. Since the basis for our pre-failure arguments might not reflect the actual safety rationale for MSAW at the time, our arguments are hypothetical; however, we were unable to locate any additional safety-related information concerning MSAW that was considered as part of the official investigations into the accidents.

We applied Pandora to each of the pre-failure arguments to produce post-failure safety arguments. When Pandora elicited counter-evidence refuting a claim, we used the findings of the official investigations to determine whether such evidence existed. We noted the fallacies that Pandora discovered in each of the arguments and then corrected those flaws in the post-failure arguments. This process was also subject to hindsight bias since we were aware of the actual recommendations of the official investigations and the corrective actions the FAA took, so we constrained our changes to those that were required to address the counter-evidence elicited by Pandora in order to restore the validity of the arguments. We present the fallacies Pandora identified and the changes made in the post-failure arguments in the next section.

5.3. Lessons Learned From Pandora

The first fallacy Pandora identified in the USAir 105 pre-failure safety argument occurred in goal G02 in Figure 3. Upon evaluating this claim, Pandora posed the question, “Is there evidence from the accident that MSAW did not visually alert a controller when an IFR-tracked a target descended below a predetermined safe altitude?” MSAW did not generate any alerts regarding USAir 105 because the MSA violation occurred in a region of airspace in which MSAW processing had been inhibited. Therefore, Pandora invalidated goal G02. We corrected this fallacy in the post-failure safety argument by weakening the claim: “MSAW visually alerts a controller whenever an IFR-tracked target descends below, or is predicted to descend below, a predetermined safe altitude *unless the violation occurs in an inhibited region of airspace.*” This change consequently invalidated the top-level claim of the argument, goal G01, because the possibility of MSAW inhibitions created instances in which a controller might not be alerted to a MSA violation. To remedy this problem, we added a new claim to the argument that the risk of a MSA violation occurring in MSAW-inhibited airspace has been sufficiently reduced. How to support this claim would be left to the discretion of the FAA, but we suggested three convergent approaches:

- Showing that the creation of inhibit zones has been minimized to instances in which continued use of MSAW would adversely impact operational priorities;
- Showing that the creation of inhibit zones is a temporary measure; and
- Showing that alternate methods for detecting MSA violations in inhibited airspace have been implemented.

We cannot conclude that these approaches would have been suggested if Pandora had been applied at the time of the accident because of the risk of hindsight bias.

The official investigation into the USAir 105 accident recommended that the FAA “provide site adaptation guidance to encourage modification of Minimum Safe Altitude Warning parameters, as appropriate, to minimize the extent of inhibit areas.” This recommendation is more specific than the risk-reduction claim we added to our post-failure argument, but it is consistent with the first of our suggested approaches for satisfying the claim.

The Pandora analysis of the TAESA Learjet accident focused on the argument supporting goal G02 in Figure 3. This argument is shown in Figure 4 with post-failure revisions indicated by a bold outline. Pandora invalidated goals G13 and G16 based upon evidence from the accident that the MSAW site variables were incorrect, and it invalidated goal G15 based upon evidence that MSAW did not generate an alert despite receiving radar returns indicating that the aircraft had violated its MSA. To restore validity to the argument, we added claims that MSAW site variables are verified to be correct (goals G201 and G204) and recommended that the FAA show either that MSAW will issue an alert when it receives a radar return indicating a violation (goal G202) or that the risk of MSAW discarding a legitimate return indicating a violation has been sufficiently reduced (goal G203). Supporting these claims would again be left to the FAA. The official investigation recommended that the FAA “conduct a complete national review of all environments using MSAW systems. This review should address all user-defined site variables for the MSAW programs that control general terrain warnings, as well as runway capture boxes, to ensure compliance with prescribed procedures.”

In both the Beechcraft A36 and Piper PA-32-300 accidents, controllers testified that they did not notice any MSAW alerts for the accident aircraft even though facility logs indicated that alerts were generated. In the Beechcraft accident, the controller did not observe the visual alerts because he was attending to other duties, and the ATC facility was not equipped with an aural alarm. In the case of the Piper accident, NTSB investigators found that the aural alarm speakers had been muted due to frequent nuisance alerts. Pandora elicited this evidence when it considered the claim that “the controller will relay an MSAW alert to the flight crew so that they can take remedial action” (goal G03 in Figure 3) in the pre-failure arguments for both accidents. In these accidents, the controllers did not relay the alerts because they were unaware of them. Therefore, Pandora invalidated the claim, and we added a new premise to repair the arguments: “The controller will recognize an MSAW alert when one occurs.” Supporting this claim would require the FAA to show that MSAW alerts are sufficiently conspicuous to attract a controller’s attention, which could entail design changes. Moreover, because the alarm speaker had been muted at the facility involved in the Piper accident due to frequent nuisance alerts, Pandora invalidated goal G07: “The frequency of nuisance MSAW alerts will be minimized.” We repaired the argument supporting this claim by adding a new premise calling on the FAA to routinely review nuisance alerts and make system design and configuration changes to reduce them while minimizing the extent of inhibit zones. The official investigations into the Beechcraft and Piper accidents recommended that the FAA install aural alarms at all its MSAW-equipped facilities and that it conduct routine inspections of the aural alarm speakers to ensure that they are not muted, respectively.

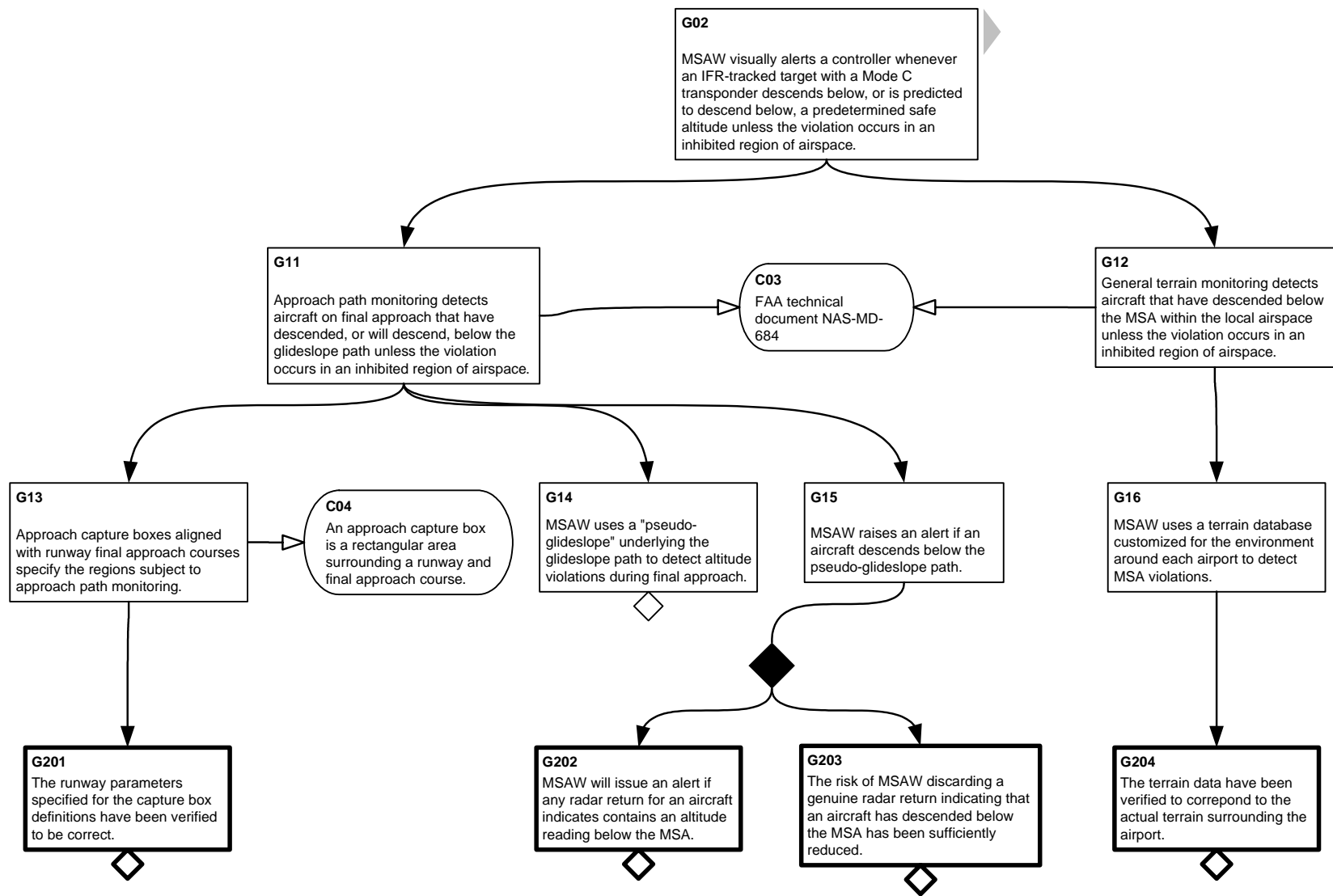


Figure 4: Pre-failure MSAW alerting argument for the TAESA Learjet accidents. A bold outline denotes nodes added in the post-failure argument.

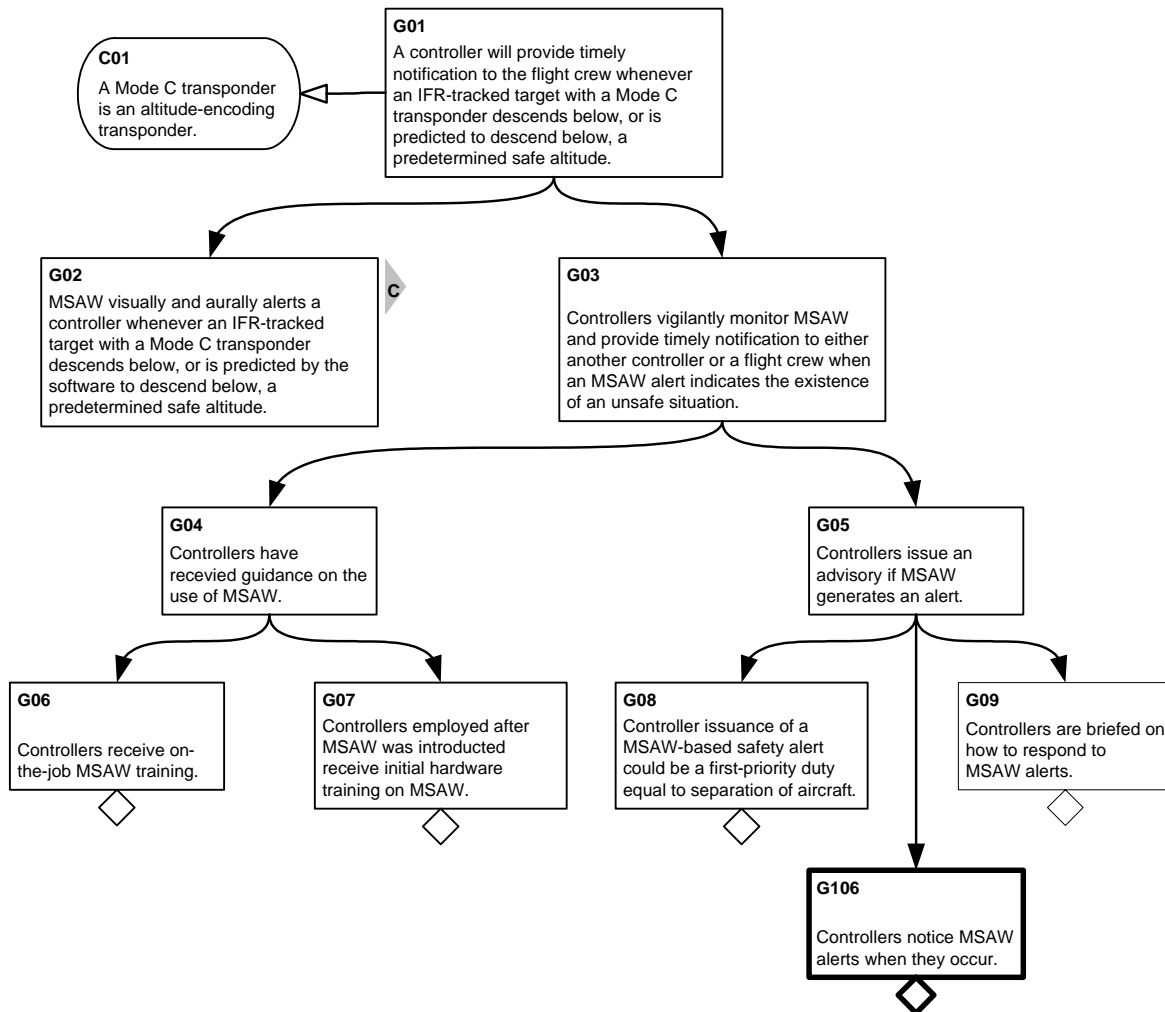


Figure 5: Top-level pre-failure argument for the Korean Air 801 accident. A bold outline denotes nodes added in the post-failure argument.

The final accident we considered, the Korean Air flight 801 crash into Guam on August 6, 1997, was a major accident with 228 fatalities. By 1997, the FAA had made several design changes to MSAW and its configuration management program for the system, and it had issued more statements regarding the safety rationale behind MSAW. To account for this new information, we rebuilt the pre-failure safety argument for MSAW from the factual basis as we did for the USAir 105 argument. Figure 5 presents the revised top-level argument, and Figure 6 depicts the argument that MSAW inhibit zones have been minimized (nodes with bold outlines were added in the post-failure argument).

Pandora's analysis of the pre-failure safety argument for the Korean Air 801 accident noted a minor fallacy in the argument supporting goal G02 in Figure 5, but this fallacy was a precursor to more significant problems that would be discovered later in the analysis. Part of the argument supporting this goal claims that MSAW will generate an alert if an aircraft descends below the normal descent path for a final approach course. Pandora invalidated this claim because the MSAW system at Guam was inhibited at the time of the accident and thus did not produce an alert when Korean Air 801 descended prematurely. We repaired the argument by adding a contextual note that MSAW would not generate an alert if it is inhibited. This step in the analysis was important, however, because it caused Pandora to elicit evidence pertaining to the inhibit zone at Guam.

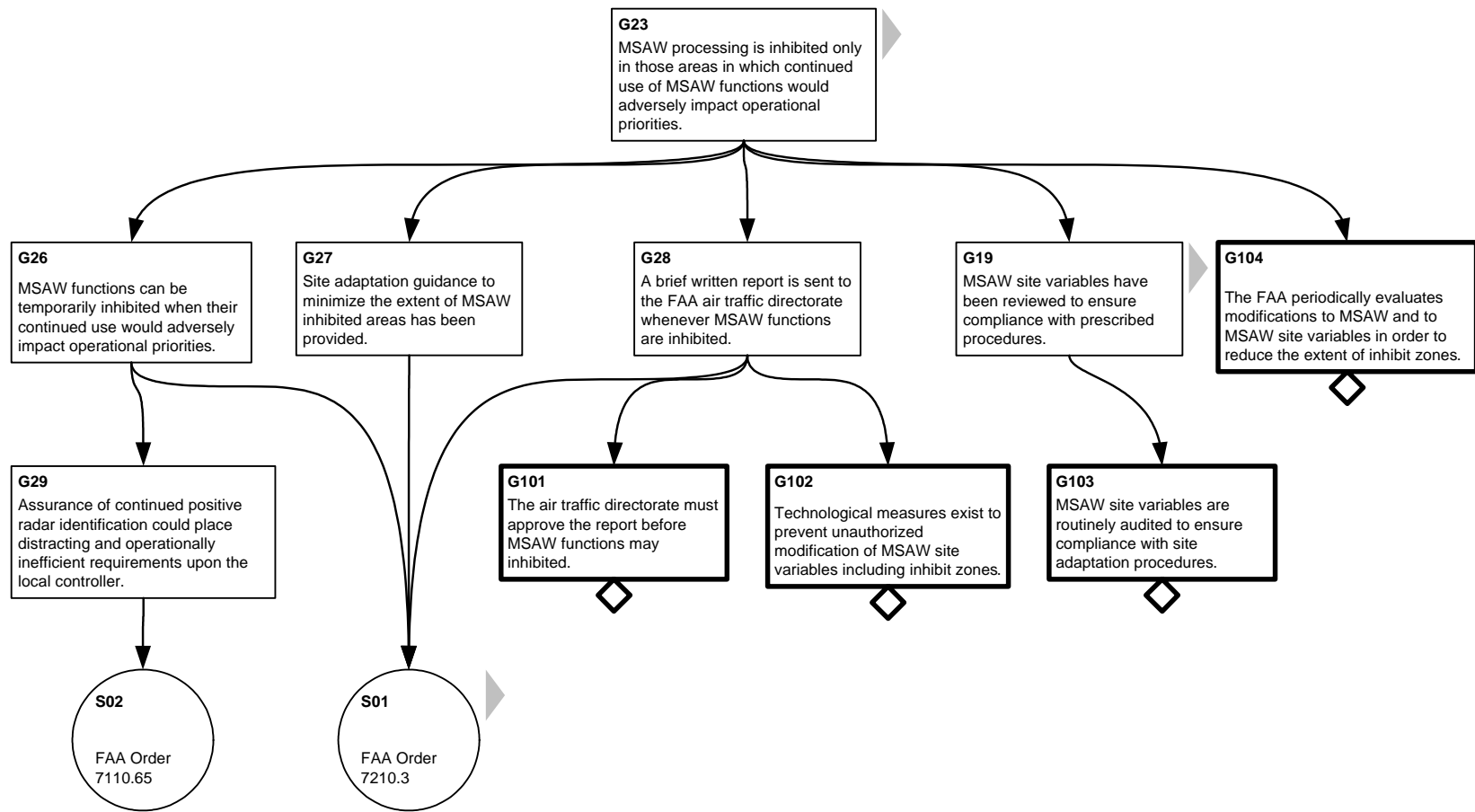


Figure 6: Inhibit zone minimization argument for the Korean Air 801 accident. A bold outline denotes nodes added in the post-failure argument.

When Pandora arrived at the argument supporting goal G23 in Figure 6, which claims that “MSAW processing is inhibited only in those areas in which continued use of MSAW functions would adversely impact operational priorities,” it had already elicited evidence of the large inhibit zone at Guam. Upon considering goal G28, Pandora called for evidence of the report that should have been sent to the FAA regarding the inhibit zone. Since the FAA could not produce this report as part of the NTSB’s investigation, Pandora invalidated goal G28, and we added two new premises to strengthen assurance that reports would be sent whenever MSAW functions are inhibited (goals G101 and G102). Pandora subsequently invalidated goal G19, which claims that “MSAW site variables have been reviewed to ensure compliance with prescribed procedures,” because the extent of the inhibit zone at Guam contradicted the intent of the FAA’s program for minimizing inhibit zones.

Pandora discovered fallacies in the argument supporting two other claims in the safety argument. The first concerned the claim that MSAW visually and aurally alerts a controller when it detects or predicts a MSA violation. Pandora invalidated this claim because the facility at which MSAW was installed lacked an aural alarm. The second fallacy was discovered in goal G05 in the top-level argument in Figure 5 because MSAW generated a visual alert, but the controller did not notice the alert.

The NTSB’s investigation into the Korean Air flight 801 accident found that the MSAW system at Guam had been inhibited and that it would have generated an alert at least one minute prior to the accident had it been functioning properly. Consequently, the NTSB cited the inhibition of the MSAW system at Guam as a contributory factor in the accident. Although the NTSB did not issue any recommendations concerning MSAW in its report, the FAA took several corrective actions on its own initiative. These included a revised training program for the operation and maintenance of MSAW, the development of uniform site adaptation parameters for MSAW, and centralized configuration management of the system.

5.4. Results of the Case Study

Pandora’s analysis of the accidents included in this case study was fairly consistent with the NTSB’s official investigations, but there were some important differences. We summarize these observations below and discuss them in the following paragraphs. We discuss limitations of the case study that may affect our observations in section 5.5. The key findings of our application of Pandora to the sequence of accidents associated with MSAW were:

- Pandora identified all the defects in the MSAW assurance case that the NTSB identified in its investigations of the accidents.
- Pandora’s recommendations tended to be more general than those of the NTSB.
- The evidence Pandora elicited spanned the breadth of MSAW-related evidence considered by the NTSB, but in some cases the NTSB considered evidence in more depth than did Pandora.
- Pandora unnecessarily considered claims that were known to have been true during the failure.
- Pandora did not consider similar systems that might be afflicted by the problems it identified.

We must be extremely cautious in interpreting these findings for many reasons. We note, however, that Pandora identified all the defects in the MSAW assurance case that the NTSB cited in its final report. Specifically, it addressed the need to minimize MSAW inhibit zones (USAir 105), the need to verify site adaptation variables to ensure correct operation of MSAW (TAESA Learjet 25D), the failure of MSAW alerts to attract controllers’ attention (Beechcraft A36 and Piper PA-32-300), and the lack of centralized configuration management of MSAW to prevent unauthorized modifications to site adaptation variables and inhibit zones (Korean Air 801). Pandora was able to arrive at a similar set of findings through a systematic traversal of the safety argument rather than an exhaustive search for evidence.

The recommendations we obtained by applying Pandora tended to be more general than those issued by the NTSB. Some of the NTSB’s recommendations called for the FAA to take a specific course of action to address a particular finding without presenting compelling evidence that the course of action would prevent similar accidents from occurring. For example, when the NTSB learned that the controller involved in the Beechcraft A36 accident was unaware of the visual alerts that MSAW generated for the aircraft, it recommended that the FAA add an aural alert to the system. Without evidence that an aural alert would have attracted the controller’s attention and led him to contact the flight crew, we cannot be certain that this action would have prevented the accident. Moreover, alternate

corrective measures, such as enhancing the conspicuity of visual alerts, might also have adequately addressed the problem. Pandora produced a more general recommendation that the FAA should provide greater assurance that controllers will recognize MSAW alerts when they occur. This recommendation gives the FAA greater flexibility in choosing corrective actions while requiring it to justify the sufficiency of whichever measures it takes. The addition of an aural alert could still be suggested as a possible means of satisfying the recommendation.

Pandora's ability to elicit evidence is dependent upon the level of detail in the pre-failure safety argument and the aggressiveness with which the process is applied. When the NTSB discovered that the MSAW system at Guam had been inhibited, it probed the complete modification history of the Guam system beginning with its initial installation in 1990. When we applied Pandora to this accident, it elicited evidence of the inhibit zone, but we did not probe into the modification history because Pandora did not explicitly call for such evidence. Thus, Pandora might not elicit all of the evidence that is necessary to understand a digital system failure; rather, it identifies which aspects of the system's safety argument investigators should focus on in their search for evidence. It is important for investigators to follow up on evidence that Pandora elicits in order to discover details that are not directly related to the pre-failure safety argument.

Pandora sometimes considered claims that were known to have been true during the failure because they were either vacuously true or there was evidence that affirmed them. In the pre-failure argument for USAir 105 shown in Figure 3, for example, goal G03, which claims that the controller will relay an MSAW alert to the flight crew, was vacuously true because MSAW did not generate any alerts. It was therefore unnecessary to evaluate the argument supporting it. Revising Pandora to skip claims that are vacuously true or for which overwhelming supporting evidence exists would improve the efficiency of the process.

Finally, Pandora did not consider similar systems that might exhibit the fallacies it identified in the MSAW safety arguments. During several of its investigations, the NTSB expressed concern that the problems it discovered in the ARTS implementations of MSAW would also manifest in STARS, the replacement system for ARTS. It issued several safety recommendations asking the FAA to ensure that MSAW functions in STARS would afford the same level of protection as they did in ARTS and to exploit new features in STARS, such as color displays, to further enhance MSAW. Pandora did not consider STARS in its analysis because none of the MSAW installations involved in the accidents were STARS-based. Broadening the scope of Pandora to include related systems in its analysis is an area for future work.

While performing this case study we also learned that the Goal Structuring Notation (GSN) does not support counter-evidence in safety arguments. GSN was developed to assist readers in understanding the structure of safety arguments and, in particular, the evidence, context, justifications, and assumptions supporting each of an argument's claims. Counter-evidence does not support a claim but rather weakens or refutes the claim. In the case of MSAW, many of the safety features of the system were introduced to address counter-evidence that was discovered from prior accidents. It is essential for a reader to be familiar with this counter-evidence in order to evaluate the soundness of an argument. The only facility GSN provides for expressing counter-evidence is the context node, which assists the reader in evaluating a claim but does not directly support it. Context nodes are often used to clarify unfamiliar terms, state facts that are not claims, or make references to other documents, and so their meaning in a safety argument is ambiguous. The use of a context node to express counter-evidence might confuse the reader because nodes in GSN generally support the claims to which they are attached. Therefore, we suggest that GSN be extended to support counter-evidence as first-class entities.

5.5. Threats to Validity

The threats to the internal validity of this case study are hindsight bias and the restriction of evidence that was available to Pandora. The threats to external validity are the use of a single system for the case study and the fact that the analyses were all performed by one person.

Hindsight bias poses the most significant threat to the validity of this case study. The experimenter who performed the case study was aware of the chronology of MSAW at the outset, and the accident history of MSAW was mixed with its safety rationale in the NTSB's final report on the Korean Air flight 801 accident. There is a risk that the experimenter inadvertently used information learned from later accidents when he constructed the pre-failure safety arguments, applied Pandora, and produced the post-failure arguments. To reduce the risk, we split the information in the report into individual facts and then categorized each fact according to the earliest accident by which it was known by the FAA. We then constrained the factual basis for each of our pre-failure arguments to what was known prior to the corresponding accident, and we enforced traceability from the nodes in the pre-failure

arguments to the basis. We also constrained the evidence that was available to Pandora to that which was uncovered during the official investigation (this created an additional threat, which we describe below). Finally, we intentionally chose a naïve approach to repairing the fallacies that Pandora identified in the pre-failure arguments in order to avoid exploiting insight gained from subsequent investigations or evidence that Pandora did not explicitly elicit. Rather than making detailed revisions to the arguments, we simply added premises to the claims Pandora invalidated in order to address the specific counter-evidence it elicited. Despite these measures, there is still a risk that our analysis benefited from hindsight.

Because we restricted the evidence that was available to Pandora to that which was elicited during the official investigation of each accident, it is possible that Pandora validated some claims for which it would have discovered counter-evidence if the evidence had been available to it. The fallacies that Pandora identified might be a subset of the fallacies it would have discovered in a real-world setting. Moreover, we refrained from following up on the evidence Pandora elicited (for example, by exploring the modification history of MSAW when Pandora discovered evidence of the Guam inhibit zone) to preserve traceability from the evidence Pandora elicited to the claims in the pre-failure safety arguments. A more aggressive approach to gathering evidence might have revealed additional fallacies in the pre-failure arguments or produced superior recommendations for addressing fallacies.

The external threats to validity are legitimate criticisms of this case study. The MSAW system and the accidents surrounding it were amenable to analysis because we were able to derive pre-failure safety arguments from the documented safety rationale, and there was sufficient evidence for us to evaluate the arguments; this might not be true of systems or accidents in general. Moreover, the fallacies and recommendations obtained by applying Pandora to a given failure may vary across investigators. Additional case studies are needed to address these issues and improve the generality of our results.

6. Related Work

In addition to Pandora, other techniques exist for repairing safety arguments and analyzing system failures. The most notable of these are safety-case maintenance, Why-Because Analysis, STAMP, and PARCEL. We discuss each of these techniques and contrast them with Pandora.

6.1. *Why-Because Analysis*

Why-Because Analysis (WBA), developed by Ladkin and Loer, is a method of developing rigorous, highly formal event-chain models of accidents and incidents based on Lewis's counter-factual reasoning [6]. WBA begins with a semi-formal graphical reconstruction of an accident's event sequence, which is then translated into temporal, counter-factual logic and verified for logical consistency and sufficiency. The result of this analysis is a highly rigorous causal explanation as to why an event occurred. WBA does not address the issues of how the event sequence is initially determined or what evidence is needed to establish the event sequence, nor does it suggest where recommendations should be issued to prevent the causal chain from recurring (although one might infer some of this information from the graphical depiction of the event sequence). Thus, WBA is useful for those who have a hypothesis as to why an accident occurred and wish to verify its soundness. Pandora is designed to facilitate the development of an accident hypothesis based upon fallacies discovered in the system safety argument as well as recommendations for repairing the system that address the deficiencies expressed in the hypothesis. Nevertheless, those desiring the rigor afforded by WBA could develop an event chain of an accident based upon the lessons derived from a Pandora investigation and then use that event chain to produce a WBA graph.

6.2. *STAMP*

Leveson's Systems-Theoretic Accident Model and Processes (STAMP) is an accident model grounded in systems theory that views accidents as the result of undesired interactions between system components [23]. STAMP models systems and accidents as control and informational feedback loops in order to depart from event-chain models and seek out management, organizational, governmental, and other social-technical factors that might have influenced the event sequence preceding an accident. STAMP classifies accidents as arising from either inadequate enforcement of constraints on system behavior, inadequate execution of control actions, or inadequate or missing feedback.

STAMP is primarily a model, not a process. Expressing the relationships between management and organizational structures and the system itself can provide insight into how those structures might have influenced the system's development or operation and possibly contributed to an accident, but STAMP does not define a rigorous process for detecting dysfunctional interactions or determining where in the social-technical structure remedies should be applied. By focusing on the system safety case, Pandora is able to examine the safety requirements, evidence of safety, and contextual details of a system's development while providing a systematic process for detecting faults and removing them. That said, Pandora and STAMP are compatible, and both techniques may be used in an investigation. This compatibility exists because safety constraints from higher-level control systems in a socio-technical structure function as safety obligations that lower-level systems must conform to by implementing their own lower-level constraints. Safety constraints appear in a system's safety argument either as goals or contextual information that frames the argument, and so they may be analyzed by Pandora. Likewise, the lessons from a Pandora investigation may be expressed in STAMP since they correspond to inadequate or inadequately-enforced safety constraints.

6.3. PARCEL

Johnson and Bowell's PARCEL (Programmable Electronic Systems Analysis for Root Causes and Experience-based Learning) is an accident analysis technique specifically developed for investigating failures of programmable systems that "traces the causes of adverse events back to the lifecycle phases a common requirements of the IEC 61508 standard" [24]. PARCEL is actually pair of causal analysis techniques. The first is a lightweight flowcharting technique that can be applied relatively quickly to suggest typical potential causal factors; this approach might be applied to low-cost accidents or incidents. The second approach involves a more complex events and causal factors (ECF) analysis that could be reserved for incidents with a higher risk of recurrence. Both of the techniques map possible causal factors in the accident or incident to lifecycle phases in system development as specified in the IEC 61508 standard. Casual factors are classified according to a taxonomy developed from the same standard.

PARCEL identifies causal factors in a digital system failure by searching for deviations between the system's development lifecycle and the lifecycle prescribed by the standard to which it was developed. Although PARCEL is based upon the IEC 61508 standard, Johnson envisions adapting it to other popular standards such as DO-178B and DS 00-55. It is not clear, however, whether PARCEL would be able to discover a flaw in the standard itself. Most digital system standards, including those mentioned, are prescriptive, and Weaver has argued that prescriptive standards do not assure safety because they over-emphasize the importance of following an unproven development process [17]. Thus, even if a system were developed in compliance with standards it is possible for the system to exhibit a safety-related failure due to a systemic fault. Whether PARCEL would be able to detect such a fault is questionable because the fault would not correspond to any deviation from the standard. Pandora does not make assumptions about the manner in which a system was developed, and so it does not suffer from this limitation; however it does assume that the system has a pre-failure safety case (or that one may be derived retroactively), and the extent of the lessons Pandora is able to derive from the safety case could be limited by the comprehensiveness of the safety case itself. Nevertheless, if the safety case is flawed, Pandora will be able to derive some lessons from it and provide at least an incremental improvement to the safety of the system.

Although PARCEL is intended to be used as the driving analysis technique in an investigation, Johnson notes that other techniques may be used to perform the causal analysis instead of ECF such as STAMP or WBA. To make the process compatible with Pandora, the requirements imposed by IEC 61508 could be expressed as a safety-case template from which the safety argument for the system under investigation is derived. Pandora could then analyze the argument, and the lessons and recommendations arising from Pandora could then be incorporated into PARCEL. Thus, investigators could choose which causal analysis techniques they wished to apply according to the nature of the system and the needs of the investigation: STAMP to analyze interactions between subsystems, WBA to produce a rigorous proof of causal arguments, or Pandora to detect fallacious reasoning in safety arguments.

7. Conclusions

We have introduced Pandora, an approach to analyzing failures of digital systems in which the analysis is framed around a system's safety case. The basic concept is that faults that lead to failures—and the development errors that introduced those faults—manifest themselves as fallacies in the underlying safety argument. Pandora detects these fallacies by systematically reviewing the safety argument and eliciting counter-evidence from the failure that refutes

its claims. When Pandora discovers a fallacy, it modifies the argument to address the evidence, yielding a new safety argument that is free of the fallacies that were identified.

Framing failure analysis around the safety case provides important benefits. Pandora may be used to scope the analysis and drive the collection of evidence because it systematically elicits the evidence that is necessary to evaluate the safety argument. Second, each recommendation that Pandora produces is stated in the context of the original safety argument and closely tied to the fallacy it is intended to address. This ensures that recommendations are justified and improves their practicability. Our application of Pandora to a sequence of related commercial aviation accidents involving a low-altitude warning system shows that it is a promising approach to analyzing digital system failures. Pandora successfully identified all the system defects that were cited by the official investigations.

Despite its benefits, we note that there are several important limitations to Pandora. In order to apply Pandora to a digital system failure, one must obtain a safety case for the system involved, and most digital systems currently in operation do not have documented safety cases. Second, Pandora's ability to elicit evidence from a failure is dependent on the completeness of the pre-failure safety case. If the safety case omits major aspects of the system's safety rationale, Pandora is unlikely to elicit detailed evidence in these areas. Finally, although Pandora is systematic, it is a manual process, and so its application will not necessarily be consistent across multiple investigators.

The future work regarding Pandora addresses the generality of our case study results, the applicability of Pandora to systems for which no documented safety case exists, and the inclusion of related safety arguments as part of the analysis. Additional case studies involving other systems and spanning multiple investigators are needed to ensure that Pandora is applicable to a wide range of digital systems and that its application is fairly consistent across investigators. Since documented safety cases do not exist for most safety-related digital systems currently operation, a method of deriving the safety cases retroactively is needed to extend Pandora's applicability to these systems. We were able to derive the safety arguments for our case study because we had access to the complete factual basis for the correct operation of the MSAW system. Finally, broadening Pandora's recommendations from a failure to include safety arguments that share the same fallacious reasoning would allow investigators to address defects in related systems before they contribute to failures.

Acknowledgements

We would like to thank Michael Holloway of NASA Langley Research Center and Jacob Pease of the University of Virginia for their assistance in developing the safety argument fallacy taxonomy. This work was funded in part by NASA Langley Research Center under grant number NAG-1-02103.

References

- [1] Leveson NG. *Safeware: System safety and computers*. Boston: Addison-Wesley, 1995.
- [2] Littlewood B, Strigini L. Validation of ultrahigh dependability for software-based systems. *Communications of the ACM* 1993;36(11):69780.
- [3] Storey N. *Safety-critical computer systems*. Harlow: Prentice Hall, 1996.
- [4] Perrow C. *Normal accidents: living with high-risk technologies*. Princeton: University Press, 1999.
- [5] Johnson CW. *Failure in safety-critical systems: a handbook of incident and accident reporting*. Glasgow: U Press, 2003.
- [6] Ladkin P, Loer K. *Why-because analysis: formal reasoning about incidents*. 1998.
- [7] Johnson CW. Forensic software engineering: are software failures symptomatic of systemic problems? *Safety Science* 2002;40(9):8357847.
- [8] Rushby J. Using model checking to help discover mode confusions and other surprises. In: *Proc. 3rd Workshop on Human Error, Safety, and Systems Development*. Belgium, 1999.
- [9] Greenwell WS, Strunk EA, Knight JC. Failure analysis and the safety case lifecycle. In: *Proc. 7th Working Conf. on Human Error, Safety, and Systems Development (HESSD)*. Toulouse, 2004.
- [10] Bishop P, Bloomfield R. A methodology for safety case development. In: *Proc. Safety-critical Systems Symposium*. Birmingham, 1998.
- [11] Adelard. *The Adelard safety case development manual*. 2005.

- [12] Kelly T, Weaver R. The Goal Structuring Notation – a safety argument notation. In: Proc. Dependable Systems and Networks Workshop on Assurance Cases. Florence, 2004.
- [13] Adelard. The Assurance and Safety Case Environment – ACSE. 2005.
- [14] CET Advantage. GSNCaseMaker: Goal Structuring Notation Tool.” 2004.
- [15] Kelly T, McDermid J. Safety case construction and reuse using patterns. In: Proc. 16th Intl. Conf. Computer Safety, Reliability, and Security (SAFECOMP’97). New York, 1997.
- [16] Kelly TP. Arguing safety - a systematic approach to managing safety cases. Diss., U of York, 1998.
- [17] Weaver RA. The safety of software - constructing and assuring arguments. Diss., U of York, 2004.
- [18] Bishop P, Bloomfield R. A methodology for safety case development. In: Proc. Safety-critical Systems Symposium. Birmingham, 1998.
- [19] Kelly T, McDermid J. A systematic approach to safety case maintenance. Reliability Engineering and System Safety 2001;71(3):271-284.
- [20] Damer TE. Attacking faulty reasoning: a practical guide to constructing fallacy-free arguments. 5th ed. Belmont: Wadsworth, 2005.
- [21] Greenwell WS, Holloway CM, Knight JC. A taxonomy of fallacies in system safety arguments. 2004.
- [22] NTSB. Controlled flight into terrain Korean Air flight 801, Boeing 747-300, HL7468, Nimitz Hill, Guam August 6, 1997 DCA97MA058. Washington, D.C., 2000.
- [23] Leveson N. A new accident model for engineering safer systems. Safety Science 2004;42(4):237-270.
- [24] Johnson CW, Bowell M. PARCEL: using software development standards to guide the investigation of computer-related incidents and accidents. 2004.