COMPUTING SCIENCE

On the Safety Analysis of Requirements Specifications

R. de Lemos, A. Saeed and T. Anderson


TECHNICAL REPORT SERIES

No. 493 October, 1994
On the Safety Analysis of Requirements Specifications

R. de Lemos, A. Saeed and T. Anderson

**Abstract**

For safety-critical systems, it is most efficient to consider safety concerns as early as possible during system development in order to ensure that safety problems do not propagate through to subsequent phases of development. In this paper, we present an approach for the safety analysis of requirements specifications that complements a framework for the analysis of safety requirements. The aim of the safety analysis is to assess, in parallel with the requirements analysis, if the risk associated with requirements specifications is acceptable.
Bibliographical details

DE LEMOS Rogério Sergio Neves

On the Safety Analysis of Requirements Specifications
[By] R. de Lemos, A. Saeed and T. Anderson

(University of Newcastle upon Tyne, Computing Science, Technical Report Series, no. 493)

Added entries

UNIVERSITY OF NEWCASTLE UPON TYNE.
SAEED, Amer
ANDERSON, Thomas

Abstract

For safety-critical systems, it is most efficient to consider safety concerns as early as possible during system development in order to ensure that safety problems do not propagate through to subsequent phases of development. In this paper, we present an approach for the safety analysis of requirements specifications that complements a framework for the analysis of safety requirements. The aim of the safety analysis is to assess, in parallel with the requirements analysis, if the risk associated with requirements specifications is acceptable.

About the author

Mr. R. De Lemos is a Research Associate in the Departments of Computing Science and Chemical and Process Engineering at the University of Newcastle upon Tyne.

Dr. A Saeed is a Research Associate in the DCSC, Department of Computing Science at the University of Newcastle upon Tyne.

Professor T. Anderson is a Professor in the Department of Computing Science at the University of Newcastle upon Tyne.

Suggested keywords

FORMAL TECHNIQUES
REQUIREMENTS ANALYSIS
SAFETY ANALYSIS
SAFETY ANALYSIS TECHNIQUES

Suggested classmarks (primary classmark underlined)
Dewey (18th): 001.6425 658.47
U.D.C. 681.322.06 519.718
On the Safety Analysis of Requirements Specifications

Rogério de Lemos¹², Amer Saeed³ and Tom Anderson¹³
¹Department of Computing Science
²Department of Chemical and Process Engineering
³BAe Dependable Computing Systems Centre
University of Newcastle upon Tyne, NE1 7RU, UK

Keywords
Safety Analysis, Requirements Analysis, Formal Techniques, Safety Analysis Techniques.

Abstract

For safety—critical systems, it is most efficient to consider safety concerns as early as possible during system development in order to ensure that safety problems do not propagate through to subsequent phases of development. In this paper, we present an approach for the safety analysis of requirements specifications that complements a framework for the analysis of safety requirements. The aim of the safety analysis is to assess, in parallel with the requirements analysis, if the risk associated with requirements specifications is acceptable.

1. Introduction

In order to ensure that a system is adequately safe (i.e. the risk of operating the system is acceptable), attention must be paid to safety concerns from the very earliest phases of the system development process. The requirements phase plays a vital role in software development, since defective requirements specifications may result in errors which propagate to subsequent phases of the development. This phase is, therefore, the appropriate level at which to initiate an examination of safety concerns. Furthermore, experiences with safety—critical systems and the analysis of software—related errors have shown that mistakes made during this phase can easily introduce faults which subsequently lead to accidents /Lutz 93/. In this paper, we present an approach for the safety analysis of requirements specifications that complements a framework for the analysis of safety requirements. This framework partitions the overall analysis of safety requirements of process control systems into smaller phases of analysis and defines an ordering relation between the phases. The effect of the framework is to produce a sequence of safety specifications that are progressively refined from the safety requirements imposed on the behaviour of a system /de Lemos 94/.

Safety analysis is conducted in parallel with each phase of requirements analysis, and has the aim of ensuring that the risk associated with the safety specifications produced during each phase is acceptable. In the case where the risk associated with a safety specification is not deemed acceptable the safety specification must be modified. The overall approach to safety analysis considers both qualitative analysis and quantitative analysis /Saeed 93/,
here we focus on qualitative analysis. The safety analysis to be performed for each phase of the process of requirements analysis is restricted to the information scoped by the level of abstraction of that phase, thereby limiting the effort required during in the safety analysis. The safety analysis for safety specifications, apart from analysing the normal behaviour must examine the failure behaviours of the safety specifications to determine their impact on safe behaviour. Our concern is with devising a systematic approach to examine failure behaviours of safety specifications, based on the principles that underpin traditional safety analysis techniques.

The rest of this paper is organised as follows. The next section overviews frameworks for requirements analysis and safety analysis. In section 3, we present the safety specification graph (SSG) as a means for recording the results of the requirements analysis and safety analysis, and relating the results from the two types of analysis. In section 4, we discuss the safety analysis of the requirements specifications in relation to the safety analysis currently conducted for safety-critical systems. In section 5, we discuss the relationship of the approach for safety analysis to current trends in standards. Finally, section 6 contributes some concluding remarks.

2. Systemising the Analysis of Safety Requirements

The approach to systemising the analysis of safety requirements is based upon two interrelated frameworks: a framework for requirements analysis that systemizes the production of safety specifications, and a framework for safety analysis that assesses whether or not the risk associated with the specification is acceptable. These frameworks structure the analysis by partitioning the overall analysis into smaller phases and defining the relationships between the phases. The aim is to conduct the safety analysis of the safety specifications in tandem with the phases of requirements analysis that produce the safety specifications. The safety specifications obtained from a phase of requirements analysis will feed into the safety analysis and any defects identified by the safety analysis will be used to modify the safety specifications before proceeding to the next phase. The main benefits of conducting the analysis at each phase, rather than at the end of the requirements analysis, are that it minimises the effects of faults propagating through the sequence of safety specifications, considers respecification as early as possible (the preferred method for the reduction of risk /MoD 91/), and reduces the complexity of the safety analysis by scoping it into smaller phases.

The relationship between a phase of requirements analysis and the safety analysis is depicted by a SADT diagram /Ross 80/ in figure 1. If the result of the safety analysis of a safety specification is that the risk associated with the specification is acceptable evidence that can be used in a safety case must be produced. On the other hand, if the safety analysis concludes that the risk is unacceptable the result of the analysis is to identify the defects in
the safety specification. Because of the iterative nature of the requirements and safety analysis, the overall process should be thought of as cyclic rather than sequential.

2.1. Framework for Requirements Analysis

The framework for requirements analysis is obtained from the structure of process control systems by identifying the key components and their interactions which establishes the domains of analysis and their inter-relationships, respectively. This process is conducted recursively, with each decomposition leading to a lower level of abstraction. From the viewpoint of safety, the framework is defined by associating its phases with the domains of analysis and the ordering of the phases with the identified inter-relationships between the domains. Each phase of requirements analysis consists of the following notions:

- **Input specifications** are the safety specifications produced from the previous phase of the framework, they will be refined in order to produce the **output specifications**. The latter characterize safe behaviour over the entities of the domain.

- **Requirements**, which are imposed on the entities of the domain, can either be functional or non-functional.
• **Models** of the domain support the analysis in order to identify the entities of the domain, capture their behaviour and describe their interactions; the models input to the requirements analysis will be refined and become part of the output specifications.

• **Safety integrity level** is an indicator of the likelihood of a safety specification achieving its required safety features under all stated conditions within a stated measure of use /MoD 91/. New safety integrity levels are associated with the output specifications produced during the analysis.

• **Defects** can either represent output specifications that do not comply with input specifications or output specifications that do not attain the established safety integrity levels.

• **Standards and guidelines** impose non-negotiable constraints on the safety specifications, and may prescribe **methods and techniques** to be employed during requirements analysis.

• For each output specification, specific **criteria** are defined to provide the basis to check whether or not the safety integrity levels are attained. Essentially, this criteria relates to the functionality of the safety specifications.

Within the context of the framework, formalisms to support the requirements analysis are grouped into two classes: property-oriented and operational /de Lemos 93/. A **property-oriented** formalism specifies the behaviour of a domain in terms of axioms (representing system properties) over a model of the domain, whereas an **operational formalism** is used to model the activities and interactions between the entities of a domain. The degree of application of one class of formalism versus the other is related to the level of abstraction being considered: at higher levels of abstraction there is a natural tendency to use property-oriented formalisms, whereas at lower levels operational formalisms predominate.

**2.2. Framework for the Safety Analysis**

The purpose of the safety analysis of the safety specifications is to increase the assurance that the contribution of a specification to the overall system risk is acceptable. The results of a safety analysis are used during requirements analysis for **risk reduction** and subsequently for **certification** of the system. For risk reduction, the safety analysis must identify the defects in the safety specifications that make the greatest contribution to the risk, providing guidance for the inclusion of additional safety precautions. For certification, the safety analysis must provide evidence that can be used in the preparation of a safety case.
A framework for the safety analysis of safety specifications, from qualitative and quantitative perspectives was presented in earlier work [Saeed 93]. In the rest of this paper, we will be focusing on the qualitative analysis, and the reasons for that include the following.

- Most current standards subject software to safety targets that are expressed in qualitative terms [Brazendale 94].

- A detailed quantitative analysis should not be conducted too early, particularly when there is limited confidence in the data on which the quantitative analysis will be based – the case during the requirements phase.

For safety specifications, the qualitative safety analysis must confirm that under normal circumstances safety specifications will prevent the system to enter a hazard state, identify possible failure behaviours of the safety specifications and analyse their impact on safe behaviour. Consequently the qualitative analysis is partitioned into two activities, preliminary analysis and vulnerability analysis. The relationship between the preliminary and vulnerability analysis is depicted by the SADT diagram in figure 2 that represents the decomposition of the safety analysis.

**Preliminary Analysis**

The preliminary analysis is concerned with ensuring that absence of hazards is maintained by the safety specifications. Preliminary analysis aims to increase the assurance in the safety specifications by verification and validation. Verification is the process of checking that the safety specifications are internally consistent (i.e. the requirements captured by the specifications are not contradictory), and that they conform to the specifications stipulated at the previous layer. Validation is the process of checking that a safety specification accurately reflects the requirements and constraints imposed on the system by the user, or some other authority, and does not conflict with the mission requirements. Each phase of preliminary analysis consists of the following notions:

- The output specifications are verified against the input specifications of the requirements analysis and validated against the user and mission requirements.

- The verification and validation of the output specifications are performed in order to confirm that the respective safety integrity levels have been attained, following general criteria established by the standards and guidelines, and specific criteria associated to a particular safety specification defined during the requirements analysis. Either the risk associated with an output specification is acceptable and evidence has to be produced to support this, or defects are found.

- The methods and techniques employed in the verification and validation of the output specifications should be part of the evidence that the specifications are able to attain their associated safety integrity levels. Formal techniques can be
Figure 2. Relationships between preliminary and vulnerability analysis

employed, together with formal proofs, for the verification, and animation techniques can be employed for the validation.

- Depending on the safety integrity levels, the standards and guidelines impose, or suggest, the methods and techniques to be used for the preliminary analysis, and the evidence that has to be produced.

The formal refinement process should not be used as the only evidence that an output specification is able to preserve the safety integrity levels dictated from the input specifications. It has also to be shown that a output specification does not violate the safe behaviour of the system, in particular influencing hazards from which it is not derived.

**Vulnerability Analysis**

The vulnerability analysis is concerned with examining failure behaviours of the safety specifications and determining their impact on safe behaviour. Vulnerability analysis probes the safety specifications to identify circumstances which can affect the safe behaviour of the system, and once such circumstances are identified the safety
specifications can be modified to reduce their contribution to the system risk. The intent behind the vulnerability analysis is similar to that for a traditional safety analysis.

The notions associated with a phase of vulnerability analysis are similar to those defined for the preliminary analysis. However, because the type of evidence that is sought from the vulnerability analysis is distinct from the one obtained from the preliminary analysis, different methods and techniques have to be employed in order to identify different defects, namely, safety specifications that violate system safe behaviour (hazard states). The features of the vulnerability analysis will be discussed in more detail in section 4.

3. Safety Specification Graph

A crucial concern in the development of safety-critical systems is to provide quality documentation that records the results of the analysis in a well structured format. Poor, or incomplete, documentation is often a major obstacle to the certification of a system. For the proposed approach we provide a Safety Specification Graph (SSG) as a means to record the results of the requirements analysis and safety analysis, and their interrelationships. The structure provided by the SSG supports the selection of information to be recorded, and records the information in a format that facilitates further analysis.

An SSG is represented as a linear graph, in which a node represents a safety specification and an edge denotes a relationship between a pair of safety specifications. For those systems whose behaviour is described through modes of operation, a separate SSG is constructed for each mode. For a mode of operation with \( p \) accidents, the SSG consists of \( p \) component graphs.

3.1. SSG Properties

An SSG has the following properties.

- The derivation of the safety specifications starts from the identification of the accidents, hence each accident represents a root from which other safety specifications are derived. This is reflected in the SSG by the construction of a distinct component graph for each accident.

- Each component graph is an evolutionary graph (a graph whose nodes and edges are not fixed, but evolve by addition or removal of subgraphs to/from terminal edges). New nodes, representing the safety specifications produced, are added to the graph by means of edges connecting them to the previous terminal nodes.

- A component graph organizes the safety specifications into layers.

- Since every safety specification is obtained from refinements performed on safety specifications from previous layers, the graph is directed and acyclic.

3.2. SSG Structure
3.2.1. Safety Specifications (Nodes)

The nodes of an SSG record the information associated with individual safety specifications, and the relationship between safety specifications of the same layer. For each node, this involves recording the following notions:

- The safety specification represented by the node, comprising of some of the following three elements:
  - *Safety strategy* - a scheme for maintaining the safe behaviour, defined as a set of conditions imposed on controllable factors;
  - *Assumptions* - hypotheses on the behaviour of the system at the level of abstraction being considered;
  - *System states* - states of the system considered unsafe for the level of abstraction being considered.

- The *safety integrity level* associated with the safety specification.

Relative weights can be associated to the safety specifications of the same layer in order to provide a notion of relative risk associated with alternative safety specifications. These relative weights are similar to those employed by FMECA techniques to obtain the risk priority numbers that provide a notion of the relative risk associated with a particular failure /DoD 84/. However, the relative weights do not give the necessary assurance that a particular safety specification, or set of safety specifications, are able to meet the required safety integrity level.

3.2.2. Relationships between Safety Specifications (Edges)

The edges of an SSG encode the relationships between safety specifications at consecutive layers of component graphs. Each edge records the relationship between two safety specifications, the specification defined at the originating node (the input specification) and the specification defined at the terminating node (the output specification). For each edge, this involves recording the following elements:

- The *evidence* produced by the safety analysis to confirm that the output specification has attained the required integrity level and complies with the input specification.
- The *criteria* used to judge that the output specification complies with the input specification and related standards.
- The specific *standard and guidelines* used to select techniques, establish integrity levels etc.

When more than one specification is related to a specification in a previous layer, then either the specifications are exclusive alternatives and a choice will have to be made in later
phases of development to select and refine a single safety specification, or the safety specifications complement each other and all are needed to attain sufficient confidence that the safety integrity levels can be achieved. These two situations are distinguished by annotating the edges with a “⊕” in the exclusive case, and a “⊙” in the complementary case.

3.3. SSG Role

The structured record of the safety specifications and their relationships, embodied in an SSG, will provide assistance for subsequent certification and maintenance activities. An SSG can also support a number of specific tasks during the requirements analysis and safety analysis.

- A key concern when safety specifications are modified is traceability, that is the ability to trace back from a specification to its origins and to trace forwards to the specifications which are derived from it. Support for traceability is provided by constructing reachability matrices and adjacency matrices of an SSG, determining respectively the set of safety specifications reachable and adjacent to a given safety specification. This enables the localization of the side-effects of a modification and the identification of relationships that must be reconfirmed, thereby increasing the assurance that when changes are necessary they will be complete and consistent.

- Internal consistency of the safety specifications of an SSG is verified by conducting horizontal and vertical checks. The horizontal checks establish that at each layer of an SSG the safety specifications are not in conflict with each other. The vertical checks are applied between different layers of an SSG, and confirm that safety (absence of hazards) is maintained down an SSG; the relationships that must be established, to ensure compliance between the layers, follow from the edges of an SSG.

- An SSG establishes a bridge between the safety analysis of the system and that of the software. This enables the risk of the software to be assessed in the system context.

- An SSG supports the construction and organization of a safety case. The basic safety arguments which constitute the safety case follow from the evidence recorded at the edges, and the structure of an SSG provides a regime in which to organize the safety arguments into a coherent safety case.

4. Systematic Approach to Vulnerability Analysis

The safety analysis of safety-critical software, typically, has been conducted by applying a combination of “best practice” techniques, such as:
• **Structured Walkthroughs** – a structured examination of the software that checks the correspondence between software design and software requirements.

• **Static Code Analysis** – tool supported analysis of software that checks for compliance against programming standards and procedures (e.g. MALPAS/TACS 92/), and detects a range of coding anomalies.

• **Testing** – execution of the software in a test environment involving checks against a specified set of test cases and acceptance criteria.

The above techniques are best employed at the later stages of software development and provide only limited assurance that the risk associated with the software is acceptable. These shortcomings and the growing maturity of formal methods have encouraged investigations into more formal approaches:

• **Semantic Analysis Criteria** – examination of a formal specification against predefined mathematical criteria for process control systems /Jaffe 91/; the criteria enable formal checks to be conducted for robustness, essential timing assumptions and reachability of safe states.

• **Formal Proofs** – mathematical analysis which confirms that software specifications maintain specified safety properties /Rushby 93/; specific approaches that support formal proofs include: formal development with VDM /Hill 88/, the ProCoS work on Duration Calculus /Ravn 93/, and formal documentation of system requirements /Schouwen 93/.

However, there are a number of arguments against relying solely on formal methods /Rushby 93/. More recently, there has been considerable interest in the application of traditional safety analysis techniques because of their successful use in application domains. Variants of the following techniques are used across a range of application domains:

• **HAZOPS (Hazard and Operability Studies)** – a qualitative procedure that enables a systematic search for hazards to be performed by generating questions based on potential deviations of system parameters /Lawley 73/.

• **FMEA (Failure Mode and Effect Analysis)** – a qualitative procedure based on identifying the possible failure modes of each component and predicting the consequences of these failures /BSI 91/.

• **FTA (Fault Tree Analysis)** – a procedure that is used to analyse a specific hazard, in order to determine the logical structure of event sequences that could lead to its occurrence /Vesely 81/.

• **ETA (Event Tree Analysis)** – a procedure that is constructed by defining a primary event and then defining the consequence events and paths that follow from this /Lees 80/.
The migration of traditional safety analysis techniques into software development has the potential to provide a good basis for an integrated approach to the overall safety analysis for both the application domain and software domain /Dhanjal 93/. However, the application of traditional techniques tends to be less effective, and more complex, when applied to software. Examples are the need for templates in software fault trees /Leveson 91/, and the “modified” keywords of extended HAZOPS techniques /Chudleigh 93, Burns 93/. An alternative method would be first to understand the basic (semantic) notions of the safety analysis techniques, and then investigate how such notions can be combined with the formal methods currently employed in the software domain /Bruns 93, Gorski 94/.

In this section we firstly discuss the principles that underpin the safety analysis techniques and their applicability to the vulnerability analysis, and then introduce the basic activities of the vulnerability analysis.

4.1. Analytical Styles

Typically, qualitative safety analysis is conducted by examining the causal relations between events and states in sequences connecting failures of components to hazard states of the system. There are two basic approaches for analysing causal relations: inductive analysis and deductive analysis /Henley 92/.

- **Inductive analysis.** Inductive (forward) analysis starts with a set of particular facts and reasons to the more general. When employed for safety analysis, inductive analysis starts with a set of failure events and proceeds forward, seeking possible consequences (i.e. hazards) resulting from the events. Typical examples are FMEA and ETA.

- **Deductive analysis.** Deductive (backward) analysis starts with a general fact and reasons towards the more particular. When employed for safety analysis, deductive analysis starts with a hazard and proceeds backwards, seeking possible failures that can lead to the specific hazard. The deductive analysis approach is typified by FTA.

Some techniques employ both inductive and deductive analysis, examples include, HAZOPS, and cause–consequence analysis /Nielsen 74/.

Both analytical styles can be employed during vulnerability analysis in order to analyse the failure behaviours of the safety specifications. The role of deductive analysis is to determine the causes of failure behaviour, and the role of inductive analysis is to determine the consequences of failure behaviour. The degree of application of one style of analysis versus the other varies according to the level of abstraction being considered and the representation technique. At higher levels of abstraction there is a natural tendency to use deductive analysis, since the description of the safety specifications is more general. Inductive analysis is more appropriate at lower levels of abstraction, since the information
on the elements of a safety specification and their inter-relationships will be more concrete.

4.2. Activities of the Vulnerability Analysis

As previously mentioned, the aim of the vulnerability analysis is to examine the failure behaviours of the safety specifications and determine their impact on the safe behaviour of the system. The overall approach closely follows the principles behind the HAZOPS technique. Consequently, the vulnerability analysis is partitioned in the following activities, to identify the failure modes of the safety specification, and to consider the causes and consequences of each of the identified failure modes on the safe behaviour of the system. The relationship between the activities of the vulnerability analysis is depicted by the SADT diagram in figure 3.

Figure 3. Relationships between the activities of the vulnerability analysis

After the failure modes of the safety specifications (output specifications) have been identified, the next two activities are to determine all possible causes for the failure behaviours and their consequences, by applying, respectively, deductive and inductive
analysis techniques. After performing the vulnerability analysis, defective safety specifications might be localized which require further corrective action, and new hazard states introduced by the safety strategies may be identified. If at this stage of the safety analysis, a safety specification is not considered to be defective, and does not introduce new hazard states, this provides evidence that can be used to claim that the demanded criteria has been fulfilled.

5. Relationship with International Standards

For the proposed approach to the safety analysis to be practicable it must be compatible with the standards against which the system will be certified. Recently many sector-specific standards and guidelines have been proposed, including: RTCA D0178B /RTCA 92/ for airborne systems, and MoD 00-55/00-56 /MoD 91/ for defence systems. It has been recognised that harmonisation would be helpful to confirm that the standards are based on the same generic principles. The preparation of a generic standard is being undertaken by the IEC (International Electrotechnical Commission) in the form of IEC/SC65A /IEC 92/. It is the intention that this standard be used as the basis for sector-specific standards, for example the draft standard BRB/LULTD/RIA /RIA 91/ for railway signalling systems.

The proposed approach, has the aim to address many of the concerns highlighted by IEC/SC65A. Specifically, the following aspects are considered.

- **Risk based.** The derivation of the safety requirements is directed by a continual risk assessment, confirming the acceptability of the associated risk or providing guidance for risk reduction.

- **Systems approach.** The hierarchical approach to the requirements analysis enables the safety analysis to optimise the overall risk of the system, by assessing the risk posed by a safety specification in the context of the overall risk rather than focusing on the risk of a safety specification in isolation.

- **Safety integrity/Safety criteria.** The approach to safety analysis accommodates the principles of establishing safety integrity levels and employing different techniques in accordance with defined safety criteria.

6. Concluding Remarks

Safety analysis has played a pivotal role in the development of traditional safety-critical systems, in assessing risk and guiding system development. Recently, it has been recognised that software development should also be subjected to safety analysis, however, unlike traditional systems, problems still exist in selecting the appropriate methods and techniques to effectively conduct the safety analysis.

In this paper, we have provided a basis for systemising safety analysis during requirements analysis. Three specific concerns were addressed.
• **Integrated safety analysis.** The relationships between the phases of requirements analysis and safety analysis are explicitly characterised. This enables the integration of safety analysis within the development process, ensuring that the safety analysis is cost effective by conducting it as early as possible and feeding the results immediately back into the development process.

• **Documentation of results.** The SSG provides a concise and organized means to document the results of the safety analysis.

• **Methods and techniques.** Progress has been made towards the selection of appropriate techniques, from two directions: an examination of the basic principles of traditional safety analysis techniques, and characterisation of the desirable features for methods and techniques that would be applicable to the safety analysis of specifications.

In future work, we will focus on identifying candidate methods and techniques for the safety analysis. Firstly, following current trends, we will consider formalisation and modification of a set of traditional safety analysis techniques that are considered appropriate for software analysis. A more radical theme is to devise safety analysis techniques based on novel abstraction notions, that are appropriate for both application domain and software domain.

**Acknowledgements**

The authors would like to acknowledge the financial support of British Aerospace (DCSC), and the SERC (UK) SCHEMA project.

**References**


