



Universidade
Federal de
Pernambuco



Specifying Safety Requirements with GORE languages



Jéssyka Vilela Jaelson Castro Luiz Martins Tony Gorschek Carla Silva

Contact: jffv@cin.ufpe.br



Outline



- Introduction
- Research Questions
- Research methodology
- Proposal
- Conclusions
- Further research

Introduction

- Safety-critical systems (SCS) are those composed of a set of hardware, software, processes, data and people
 - whose failure can result in accidents that cause environmental damage, financial loss, injury to people and even loss of lives.
- Problems in the specification of safety-critical systems
 - have been identified as a major cause of many accidents and safety-related catastrophes.
- In safety requirements specification, there are many relationships among safety concepts that must be identified and specified.
 - Achieving an adequate representation of safety-critical systems requirements is quite fundamental for a successful safety analysis.
- Safety concerns should be considered early in the development process, especially in the RE phase.
 - An elaborated requirements engineering (RE) approach is crucial in the development of SCS in order to meet time, cost, and quality goals in SCS development.



Introduction

- Despite the need of addressing safety concerns early in the development process
 - there is no consensus on the features an RE language must provide to support the description of such systems.



- In order to improve the safety requirements specification
 - it is necessary to define a conceptual foundation as well as the features that requirements languages should have to support this task.
- In this context, Goal-Oriented Requirements Engineering (GORE) emerged as new paradigm to improve the RE process.
 - A variety of GORE languages have become popular to represent and analyze requirements.
 - In recent systematic mapping, Horkoff et al. [21] observed that i*, GRL, NFR framework, GBRAM, and KAOS are the most adopted languages.

Research Questions

- *RQ1: What is the conceptual foundation for safety requirements specification in RE process?*
- *RQ2: What are the main features that requirements languages should support in terms of safety requirements specification?*
- *RQ3: What are the similarities and differences among GORE languages support for the features of RQ2?*



Research methodology

Definition of research questions

Establishment of a safety conceptual foundation

Development of a conceptual model for safety requirements specification

Features selection

Comparison of GORE languages

Conceptual foundation for specification of safety requirements in RE process (RQ1)

Table 1: List of sources of information.

#	Source	Type
1	ISO 61508	Generic standard
2	ISO 26262-6	Automotive standard
3	ISO/IEC 25010	Generic standard
4	ISO/IEC 9126	
5	ISO 15998-1	Machinery standard
	ISO 15998-2	
6	ISO 20474-1	Machinery standard
7	ECSS-E-HB-40A	Space standard
	ECSS-E-ST-40C	
8	ISO-13849-1	Machinery standard
	ISO-13849-2	
9	MIL-STD-882C	Defense standard
	MIL-STD-882D	
	MIL-STD-882E	
10	ISO/TR-14639-1	eHealth standard
	ISO/TR-14639-2	
11	Vilela et al.	SLR
12	Martins and Gorschek	SLR
13	Zoughbi et al.	Journal Paper
14	Markovski et al.	Conference Paper



Features of Safety-Critical Systems (RQ2)

#	Feature	Source/Inspiration
1	Modeling of accident	[8][13][47][48]
2	Modeling of hazard	[8][13] [47][48]
3	Modeling of cause of hazard	[8][13] [47][48]
4	Modeling of environmental condition	[8][13] [47][48]
5	Modeling of functional safety requirement	[8][13] [47][48]
6	Representation of constraint	[13][14][15] [47][48]
7	Representation of obstacle	[13][14][15] [47][48]
8	Representation of pre and post condition	[13][14][15] [47][48]
9	Allow to represent the relationships among hazards, their causes, the environmental conditions and the functional safety requirements in a graphical form	[8][16]
10	Ability to specify how a particular event affects system safety	[10][12] [47][48]
11	Ability to specify the criticality level of safety-critical elements or the element's contributions to failure conditions	[17][18] [47][48]
12	Model and reasoning of safety strategies	[8][10][12]
13	Ability to model resources	[10][12]
14	Modeling of accident impact level	[8][10][12] [47][48]
15	Support of textual description of safety requirements	[8] [47][48]

Comparison of GORE Languages (RQ3)

- We evaluated the support of the four most used [21] GORE languages (i*, KAOS, GRL, NFR-Framework) for the safety-related concepts.
- To conclude whether a language supports a feature:
 - we relied on existing case studies;
 - the description of the languages in the investigated papers and tools;
 - we did not considered extensions for these languages.

Table 2: Papers adopted to evaluate the languages.

Language	Paper adopted	Tool
i*	[23] ERIC, S. K. Social modeling for requirements engineering. Mit Press, 2011.	OpenOme
KAOS	[24] DARDENNE, Anne; VAN LAMSWEERDE, Axel; FICKAS, Stephen. Goal-directed requirements acquisition. Science of computer programming, v. 20, n. 1-2, pp. 3-50, 1993.	RE-Tool
NFR	[25] MYLOPOULOS, John; CHUNG, Lawrence; NIXON, Brian. Representing and using nonfunctional requirements: A process-oriented approach. IEEE Transactions on software engineering, v. 18, n. 6, pp. 483-497, 1992.	OME
GRL	[26] AMYOT, Daniel; MUSSBACHER, Gunter. Development of Telecommunications Standards and Services with the User Requirements Notation. In: Workshop on ITU System Design Languages, 2008.	OME

Comparison of GORE Languages (RQ3)

	Feature	i*	KAOS	GRL	NFR Framework
1	Modeling of accidents	N	P (Obstacle)	N	N
2	Modeling of hazards	N	P (Obstacle)	N	N
3	Modeling of causes of hazards	N	P (Sub-obstacles)	N	N
4	Modeling of environmental conditions	N	Y (Trigger conditions)	N	N
5	Modeling of functional safety requirements	Y (Tasks)			Y (Operationalizations)
6	Representation of constraints	Y (Contribution Links)			
7	Representation of obstacles	N	Y (Obstacle)	N	N
8	Representation of pre and post conditions	N	Y (pair Precondition, PostCondition)	N	N
9	Allow to represent the relationships among hazards, their causes, the environmental conditions and the functional safety requirements in a graphical form	N	N	N	N
10	Ability to specify how a particular event affects system safety	Y (Sofgoals and Contribution Links)			
11	Ability to specify the criticality level of safety-critical elements or the element's contributions to failure conditions	N	N	N	Y (Priority “!” symbol in softgoals)
12	Model and reasoning of safety strategies	Y (Sofgoals and Contribution Links)			Y (Operationalizations and Contribution Links)
13	Ability to model resources	Y (Resource Element)			Y (operationalizations)
14	Accident impact level	N	N	N	N
15	Support of textual description of safety requirements	N	N	N	N

Comparison of GORE Languages (RQ3)

- In summary, all surveyed approaches lack explicit modeling constructs to express how hazards can occur in the system, the accidents, their impact and how they can mitigated.
- KAOS better supports some features in relation to the other languages
 - The language is composed by different models and have more concepts than other languages.
 - KAOS was the only GORE language present in the SLR we conducted about the integration between RE and safety analysis [8].
- The features not supported by KAOS are either not supported by i*.
- i* and GRL have similar coverage.
- NFR is the least appropriate language to specify the requirements of safety-critical systems.
 - It does not have constructs to model accidents, hazards, their causes, environmental conditions, obstacles, and pre and post conditions.

Conclusions



- The safety concepts and features outlined in this paper may be used by requirements engineers to represent the results of a preliminary safety analysis (PSA).
- In a complete safety analysis, a richer set of attributes and relationships are specified.
 - In this paper, we are concerned with the core concepts that are available in the RE process.
 - The full list addressed of concepts used in safety analysis, such as design options and components, is out of the scope of this paper.
- The high level specification of such safety concepts may be used by safety engineers as an input of a rigorous and detailed safety analysis in the preparation of reports for system certification.

Conclusions



- An early safety requirements specification will contribute to:
 - reduce the errors in requirements specifications (increasing quality);
 - better information presentation and increased information consistency;
 - it allows exhaustive and detailed user feedback making possible to discover and specify the complete system behavior.
- We proposed a set of 15 features that requirements languages should provide to an early safety requirements specification.
 - This set is a novel contribution that can be used as criteria to guide the selection of requirements languages for specifying SCS.
- We evaluated four GORE languages (i*, KAOS, GRL, NFR-Framework) against this set of features.
 - We observed that none of these languages fully support the specification of SCS.
 - KAOS and i* appears to be promising languages in which extensions could be proposed to support the features.

Further research

- How is the support of non-GORE languages such as UML to describe SCS requirements?
- In what extent do extensions to security cover the concepts and needs of SCS?
- How can we evaluate the completeness of the proposed set of features?
- How can we adapt the evaluated languages to support the features?





Universidade
Federal de
Pernambuco



Specifying Safety Requirements with GORE languages



Jéssyka Vilela

Jaelson Castro

Luiz Martins

Tony Gorschek

Carla Silva

Contact: jffv@cin.ufpe.br

Join our survey:

<https://sites.google.com/view/unirepm>