Specifying safety monitors for autonomous systems

Mathilde Machin, Jean-Paul Blanquart, Jérémie Guiochet, David Powell, Hélène Waeselynck

Specifying safety monitors for autonomous systems

Mathilde Machin*†, Jean-Paul Blanquart§, Jérémie Guiochet*†, David Powell*‡ and Hélène Waeselynck*‡

* *CNRS, LAAS, 7 avenue du colonel Roche, F-31400 Toulouse, France*
† *Univ de Toulouse, UPS, LAAS, F-31400 Toulouse, France*
‡ *Univ de Toulouse, LAAS, F-31400 Toulouse, France*
{mmachin | guiochet | dpowell | waeselynck}@laas.fr

§ *EADS Astrium, 31 rue des cosmonautes, 31402 Toulouse, France*
jean-paul.blanquart@astrium.eads.net

I. Introduction

Autonomous systems aim to be versatile and able to perform tasks in various ill-defined environments. These systems are often critical since their failure can lead to large financial losses or human injury. In this context, classical safety measures are inflexible and are not sufficient to guarantee that the system behaves safely. For instance, emergency stop buttons, if used alone, transfer responsibility for safety to the user, which is clearly inadequate for a system that is supposed to be autonomous. Electromechanical solutions such as bumper motor switches reduce versatility since, for example, they cannot be used for systems that need to push objects or operate in the presence of fragile obstacles.

As a result, autonomous systems have to be equipped with means for context-dependent safety enforcement. We consider here a device called a safety monitor, which is equipped with the necessary means for context observation (i.e., sensors) and able to trigger, when necessary, the appropriate safety action. We require the monitor to be *maximally permissive*, in that it should only restrict its *versatility* (i.e., what the system is able to do) to the extent necessary to ensure safety.

Flexible safety measures are used in the robotic domain in [1], where the safety context is simply the distance from the robot to a human, because only the robot-to-human collision hazard is taken into account. A richer context is needed in order to cover a wider range of hazards, obtained through hazard analysis, as in [2].

Once a hazard is identified by hazard analysis, it is necessary to specify what the monitor has to do to avoid it. This is not straightforward. To determine when to act, precursory conditions have to be extracted from the hazard analysis. Obviously, how to intervene is also of interest and according to the chosen strategy, the precursory conditions may be different. We call the couple when/how or observation/intervention a *safety rule*, which is a requirement for the safety monitor.

We distinguish *initiative* and *restriction* rules. An initiative rule launches an *action* in order to change the state. On the contrary a restriction rule *inhibits* certain state changes, e.g., by means of an interlock device or by request filtering.

Synthesis of restriction rules has been widely studied in the context of *supervisory control* [3], which is a formal method that generates a 'controller' from a system model and a constraint model (to avoid the catastrophic state in our case), both expressed as automata. In this context, 'control' means forbidding the occurrence of certain controllable events.

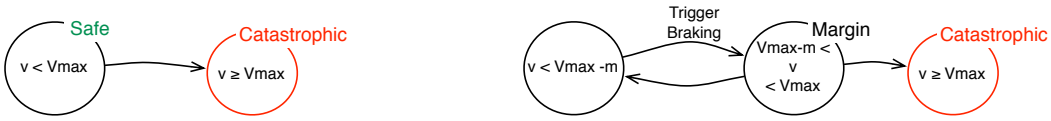
However, the need for initiative rules arises when inhibitions alone cannot ensure safety or are inefficient. For instance, to avoid a mobile obstacle, the triggering of 'brake' or 'swerve' actions would be more efficient then forbidding 'acceleration'. Since any such action takes a non-zero time to produce an effect (for example, braking in order to stop), it has to be anticipated by some *safety margin* (with respect to time or some other physical variable)

Our approach for specifying safety monitors is based on hazard analysis and considers both initiative and restriction rules. After a brief summary of previous work, we give the directions of our current research aimed at strengthening requirement elicitation by the use of formal methods.

II. Previous work

Previous work [4] has addressed the process for eliciting safety rules, based on a HAZOP-UML hazard analysis [5]. The system is described abstractly with UML use cases and sequence diagrams. Each row of a HAZOP table considers a deviation of the UML model and its consequences, assigning a severity level to it. For each deviation with a severity level of "serious" or higher, a constraint is formulated in natural language, by negation of the deviation or of an effect of the deviation.

The constraint is then expressed formally as an invariant, with predicates on variables that are observable by the monitor. A region graph is built from the invariant; the region violating the invariant is called the catastrophic region. Each transition leading to the catastrophic region has to be neutralized either by an inhibition or by insertion of an action and an associated safety margin (e.g., see Figure 1). Safety rules elicited from each deviation should be applied in the context of the considered use case. The currently applicable use case is assumed to be identified on-line by the safety monitor. The problem of simultaneous and potentially conflicting interventions is addressed by composition of graphs and manual analysis.



(a) Region graph from invariant

(b) Region graph with a margin state

Fig. 1: Example of margin insertion in region graph applied to a simple speed limitation

III. Current work

We are currently extending the three steps of the work described in Section II: hazard analysis, safety modeling and safety rule elicitation.

1) HAZOP-UML improvements: Since a use case may in fact encompass several different situations, different safety rules can be required. We need contexts at a finer granularity than use cases. We extend the HAZOP table by the addition of a contextual information column. It aims to identify in the whole context (speed, temperature, human distance...) the relevant conditions under which the considered deviation leads to a given consequence. This extra piece of information enables the safety rules to be applied only in relevant cases and incites experts to discover other contexts where the consequences are different.

2) Towards a formal model: HAZOP-UML is an informal analysis using natural language whereas we aim to have a computable model. To ease formal modeling, we propose to disambiguate the constraint formulated from a HAZOP row by using a CNL (Controlled Natural Language), which is a natural language with restriction on syntax and vocabulary [6].

In addition, we aim to propose a template for analyzing each constraint. Safety-relevant variables are identified as well as thresholds. Unobservable conditions have to be replaced by predicates on observable variables. We also need to reconsider the differences between the ideal physical variables mentioned in HAZOP and their logical representation within the monitor, taking account of sensor accuracy, precision, range, sampling rate, and so on. Once the region graph is built, potential actions are examined. If the observed variables are controllable, this is quite straightforward. In case of uncontrollable variables, indirect actions have to be considered, which could imply additional observable variables. Actions require the insertion of margin regions. Variables, thresholds and actions are iteratively added to the graph until it models the state space behavior of the complete set of safety rules.

3) Formal support: We aim to improve the scalability of the method by the use of formal methods applied at each of three steps:

a) *Design and checking of safety rules for one constraint.* Model-checking can be used to guarantee that the local catastrophic state is inaccessible. Supervisor synthesis can find possible restrictions. It may not be possible to define an automatic method to find initiative rules. However, suggesting initiatives may be possible from a catalogue of actions, with models of their effects on observable variables.

b) *Evaluation of redundancy between constraints.* We have found from several case studies that it is quite common for a single safety rule to cover constraints from several HAZOP rows. To discover this automatically, we need a formal model, i.e., the safety invariant, and a formal method that can reason about predicates on real variables, such as an SMT (Satisfiability Modulo Theory) solver.

c) *Validation of the overall set of rules.* Once all safety rules are defined, all relevant thresholds are identified and real domains with their predicates are reduced to enumerated sets of values. Therefore, we could use classical boolean model-checkers, to check overall properties such as the presence of simultaneous actions. To assess monitor permissiveness, versatility has to be modeled either by CTL (Computational Tree Logic) properties or by graph models.

IV. Conclusion

Our current research aims to ensure correctness, completeness and consistency of safety monitor requirements by formal methods. The results of formal methods are exact. However, if the underlying models are inaccurate, exact results have no interest. Therefore, the modeling step has to be done very carefully (in our case, the constraint analysis). Using different formal methods, the problem of translation and equivalence between formalisms will arise. The underlying goal is to have a formal proof of correctness from hazard analysis to implementation.

Our method will be applied on an industrial robot that helps aeronautics workers to install brackets in aircraft. The task is a collaborative one: the robot projects an image and prepares the surface with a solvent, while the worker glues the bracket.

References

[1] S. Haddadin, M. Suppa, S. Fuchs, T. Bodenmüller, A. Albu-Schäffer, and G. Hirzinger, "Towards the robotic co-worker," in *Robotics Research*. Springer, 2011, pp. 261–282.

[2] R. Woodman, A. F. Winfield, C. Harper, and M. Fraser, "Building safer robots: Safety driven control," *The International Journal of Robotics Research*, vol. 31, no. 13, pp. 1603–1626, 2012.

[3] P. J. Ramadge and W. M. Wonham, "Supervisory control of a class of discrete event processes," *SIAM journal on control and optimization*, vol. 25, no. 1, pp. 206–230, 1987.

[4] A. Mekki-Mokhtar, J.-P. Blanquart, J. Guiochet, D. Powell, and M. Roy, "Safety trigger conditions for critical autonomous systems," in *18th. Pacific Rim Int'l Symp. on Dependable Computing (PRDC)*. IEEE, 2012, pp. 61–69.

[5] J. Guiochet, D. Martin-Guillerez, and D. Powell, "Experience with model-based user-centered risk assessment for service robots," in *12th Int'l Symp. on High-Assurance Systems Engineering (HASE)*. IEEE, 2010, pp. 104–113.

[6] R. Schwitter, "Controlled natural languages for knowledge representation," in *23rd Int'l Conf. on Computational Linguistics (COLING '10): Posters*. Association for Computational Linguistics, 2010, pp. 1113–1121.