Combining GSN and STPA for Safety Arguments

Celso Hirata[1][0000-0002-9746-7605] and Simin Nadjm-Tehrani[2][0000-0002-1485-0802]

[1] Instituto Tecnológico de Aeronáutica, São José dos Campos, SP, Brazil
[2] Linköping University, Linköping, SE-581 83 Sweden
hirata@ita.br, simin.nadjm-tehrani@liu.se

Abstract. Dependability case, assurance case, or safety case is employed to explain why all critical hazards have been eliminated or adequately mitigated in mission-critical and safety-critical systems. Goal Structuring Notation (GSN) is the most employed graphical notation for documenting dependability cases. System Theoretic Process Analysis (STPA) is a technique, based on System theoretic Accidents Model and Process (STAMP), to identify hazardous control actions, scenarios, and causal factors. STPA is considered a rather complex technique, but there is a growing interest in using STPA in certifications of safety-critical systems development. We investigate how STAMP and STPA can be related to use of assurance cases. This is done in a generic way by representing the STPA steps as part of the evidence and claim documentations within GSN.

Keywords: GSN, Assurance case, STAMP, STPA.

1 Introduction

Assurance case or safety case has been employed in many safety-critical systems such as avionics, nuclear, and railway control systems. It is used to explain why all critical hazards that create unacceptable risks have been eliminated or adequately mitigated in mission-critical and safety-critical systems.

Goal Structuring Notation (GSN) [1-2] is a graphical notation that is used in assurance cases, well-represented in both academia and industry. In general, it is used with other hazard analysis techniques.

Systems-Theoretic Accident Model and Processes (STAMP) is an accident causality model, based on system theory [3]. System-Theoretic Process Analysis (STPA) [4] is a technique, based on STAMP, to identify hazardous control actions, scenarios, and causal factors. STPA is considered a rather complex technique to be used since it requires a different analysis perspective compared to other hazard analysis techniques, such as fault tree analyses, failure modes and effects analyses. STPA derives the analysis in terms of control actions, feedbacks, and other interactions.

There is growing interest in using STAMP and STPA in certifications and definitions of standards of safety-critical systems development because it is claimed that STPA is able to identify more loss scenarios due to hazards in the concept stage of development

life cycle [5]. There is also growing interest in adopting GSN as part of OMG standards and other practical guidelines [6].

In the best practice, an engineering organization starts a dependability case early in the development life cycle, using the case's structure to influence assurance-centred actions throughout the life cycle. In this paper, we pose the question: how does a systems engineer leverage the benefits of STPA when analysing safety at the concept stage, and weave the argumentation structure into an assurance case with GSN?

Building an assurance case based on STPA can aid determining what claims can be made, what assumptions, contexts and justifications are employed by STPA, and how evidence, potentially created by alternative techniques, can be used to support such claims.

We investigate how STAMP/STPA can be combined with GSN so that one contributes with the safety analysis and the other with safety case construction. We use a simple example to illustrate the joint approach, and then go ahead with making a generic pattern that will aid applying the technique to other examples.

Using this preliminary investigation, we find it feasible to use GSN for supporting certification decisions, improving communication among safety engineers, and importing the argumentation structure from a (favourite) safety analysis approach, in this case STPA. This is useful for those engineers who are familiar with GSN but may have resorted to other hazard analysis techniques earlier. Conversely, we create a pattern for employing STPA analyses when creating evidence in assurance cases, in particular using GSN.

The rest of paper is organized as follows. The next section provides the background and related work. Section 3 presents the assurance case building on STPA using GSN. In Section 4, we discuss the case and conclude our work.

2 Background and Related Work

We begin by providing a brief overview of the used approaches and then compare with the related works.

2.1 GSN, STAMP and STPA

Assurance case is a reasoned and compelling argument, supported by a body of evidence, that a system, service or organization operates as intended for a defined application in a defined environment. Assurance cases have particular foci or contexts. The contexts can vary depending on concerns, for instance, safety and security, or within phases or activities of development process, such as design and implementation.

Goal Structuring Notation (GSN) [1, 2, 7] is a graphical notation for creating assurance cases that can be used to explicitly document the elements and structure of an argument and the argument's relationship to evidence.

An argument is defined as a connected series of claims intended to establish an overall claim. Claims can be structured as a hierarchy of claims and sub-claims that are supported by evidences. Claims and sub-claims are *goals* and are represented by rectangles. Evidence is asserted to support the truth of the claim and it is also known as

solution. Evidence is represented as a circle. *Strategy*, which is represented by a parallelogram, is the reasoning or the nature of the argument that links the claim to its subclaims. *Context*, which is represented by rectangles with round corners, helps documenting the operational usage environment for the objective to be relevant or the strategy. It helps describing how a claim or strategy should be interpreted.

Most claims and argumentation strategies are expressed in the context of assumptions. The assumptions must be valid for the claim or the strategy to be valid. *Assumptions* are represented by ellipses. Claims and argumentation strategies need justifications to be acceptable. Justifications are also represented by ellipses. A diamond attached to an element, indicates that a line of argument has not been developed yet.

Two types of linkages between GSN elements are *SupportedBy* and *InContextOf*. SupportedBy relationships – represented by lines with solid arrowheads – indicate inferential or evidential relationships between elements. InContextOf relationships – represented as lines with hollow arrowheads – declare contextual relationships.

Systems-Theoretic Accident Model and Processes (STAMP) [3] is based on three concepts: (i) a Safety Control Structure (SCS) – which is a hierarchical representation of the system under analysis on which upper level components impose constraints on lower level components; (ii) a Process Model - a model of the process being controlled; and (iii) Safety Constraints – restrictions that the system components must satisfy to assure safety.

System-Theoretic Process Analysis (STPA) [4] is a technique based on STAMP for accident analysis. STPA has four main steps. *Define the Purpose of the Analysis* aims to identify losses, hazards, and the system boundary. *Model the Control Structure* captures functional relationships and interactions using STAMP. The third step - *Identify Unsafe Control Actions* - identifies the potentially Unsafe Control Actions (UCA) and associated safety constraints. For each Control Action (CA) – a command usually issued towards the controlled process – the analyst must identify cases where a CA can be hazardous. The fourth step – *Identify Loss Scenarios* - reveals potential causes of issuing UCAs. For each UCA identified earlier, the goal is to discover scenarios and associated causal factors that can lead the system to a hazardous state, and to generate safety requirements.

In general, each unsafe control action can be inverted to define a constraint. In this work, we opt to employ unsafe control actions instead of constraints in the argumentations because unsafe control actions are also required in the fourth step of STPA *Identify Loss Scenarios*. Safety constraints and requirements assist designers in eliminating or mitigating the potential causes of unsafe control and the occurrence of hazards. The fourth step demands safety analysts' expertise, time, and effort for elaboration and verification. It is common to miss cases (e.g. scenarios, causal factors, requirements) when performing this step. So all other existing methods that can strengthen elaboration and verification can be useful, for instance formal model-based analysis – but this is outside the scope of this paper.

2.2 Related Work

Rinehart et al. [8] provide an extensive report on assurance case practices and their effectiveness. They link the success of assurance cases to evidences, and show that goal-orientation and explicit argumentation are core strengths of assurance case methods. They posit that the assurance methods are more comprehensive than conventional methods such as fault tree analysis, failure modes and effects analysis, and are typically employed in conjunction with them. However, the report lacks references to STAMP/STPA. Rinehart et al. also observed that many academic research papers involve small toy example cases. On the other hand, practitioner reports tend to focus on lessons learned and experience, without presenting to the reader how the complexity of large systems is handled. This is a considerable concern in their view. We will come back to this later in our discussion.

Dependability cases are being recommended as a good way to explain why all critical software hazards have been eliminated or adequately mitigated in mission-critical and safety-critical systems. Goodenough and Barry [9] present an example of a software-related hazard to show the value that a dependability case adds to a traditional hazard analysis. The example shows the power of the claims-arguments-evidence structure to clarify the hazards and show why the selected mitigations are effective.

Complementarily, STPA includes software and human operators in the analysis, ensuring that the hazard analysis captures potential causal factors of losses. In one study, STPA not only found all the causal scenarios found by the more traditional analyses but also identified more scenarios compared to those [4].

Model-based analysis of architectural decisions that inevitably impact safety analyses spans over another category of work. These lead to provision of concrete evidence and support for safety arguments. An example is the methods and tools proposed by Hugues and Delange [10]. For a broader view of how modelling, verification, hazard analysis, and safety assurance argument documentation can interact, we refer the reader to Denney and Pai [11] but we will not focus on model-driven verification for generation of evidence in this paper. Rather, we focus on the combination of the high-level arguments captured in GSN and hazard analysis through STPA respectively.

3 Using GSN to Document Application of STPA

In this section, we first apply the basic idea of the paper using a running example that clarifies the different roles for each approach. Then we go on to create a general pattern that is based on an abstraction of this example and hopefully a good basis for further work.

3.1 The Train Door Controller as Running Example

We use a simple system - Train Door Controller (TDC) - as an example which was earlier described in Thomas' work [12], and as a well-known example helps to ease the understanding of argumentations. We assume that the STPA analysis is already made. The results of the analysis include identification of accidents, hazards, safety control

structure (controller, door actuator, door system, sensors of person and door position, control actions, feedbacks, input and output of the controlled process, external communications, process model, and algorithm), unsafe control actions, scenarios and causal factors, and safety requirements. Fig. 1 shows the safety control structure of the TDC.

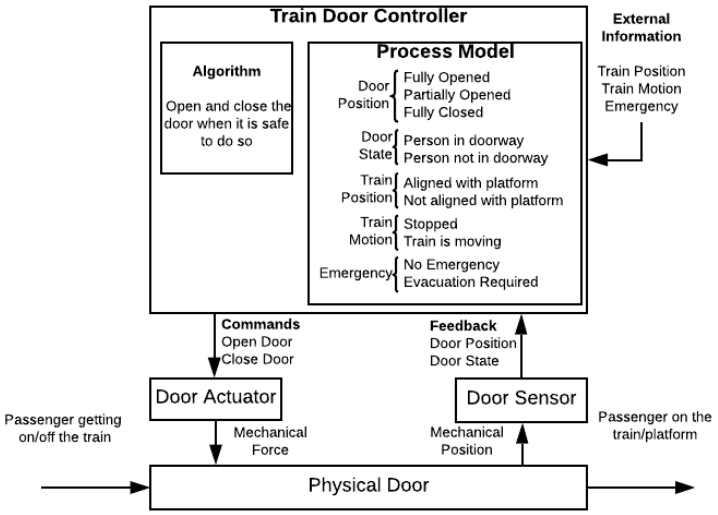

Fig. 1. Safety control structure of Train Door Controller (adapted from [12]).

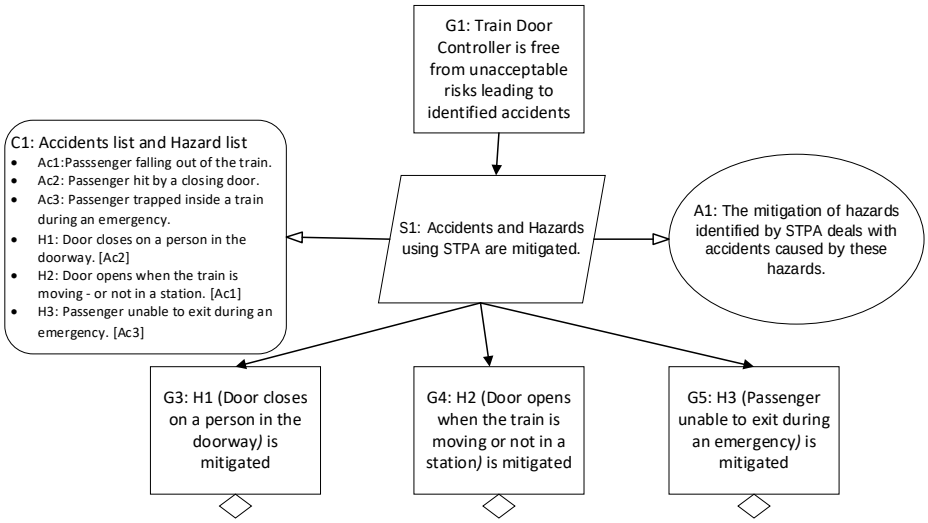

Fig. 2. Goal: *Train Door Controller is free from the identified accidents*.

Fig. 2 shows the GSN notation for goal *G1: Train Door Controller is free from unacceptable risks leading to identified accidents*.
The aim is to show that this goal is achieved by identifying hazards through STPA and showing that they are mitigated. The goal G1 is addressed by arguing about accidents and hazards using the strategy S1. This strategy can only be executed in the context of knowledge of identified accidents and hazards (C1). S1 assumes that *the mitigation of hazards identified by STPA deals with accidents caused by these hazards* (A1). As indicated in the context for S1, TDC has three identified accidents, which are *Ac1: Passenger falling out of the train, Ac2: Passenger hit by a closing door*, and *Ac3: passenger trapped inside a train during an emergency*. The identified hazards are *H1: door closes on a person in the doorway*, *H2: door opens when the train is moving or not in a station*, and *H3: passenger unable to exit during an emergency*. The accidents and hazards are identified in the STPA step *Define the Purpose of the Analysis.*

Fig. 3 shows how the hazards H1, H2, and H3 are mitigated (with H1 illustrated in detail). These are represented by the goals G3: *H1 (Door closes on a person in the doorway is mitigated) is mitigated*, G4: *H2 (Door opens when the train is moving or not in a station) is mitigated*, and G5: *H3 (Passenger unable to exit during an emergency) is mitigated*. The goals are supported by reasoning over the safety control structure (Figure 1), which is a product of the STPA analysis. The strategy S2 considers the context of knowledge of the safety control structure (C2). We assume that SCS provides the knowledge to identify unsafe control actions, represented by A2. The safety control structure is built in the STPA step *Model the Control Structure*.

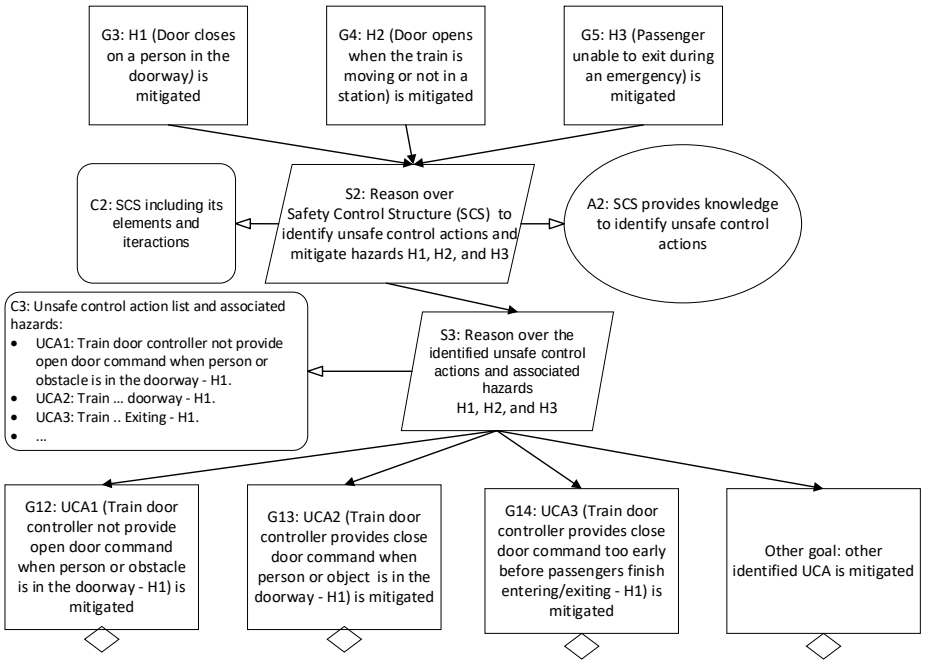

Fig. 3. Mitigation of hazards: goals 3, 4, and 5.

G3, G4, and G5 are supported by the goal G12, G13, and G14 using the strategy S3 in the context of knowledge of the list of unsafe control actions (C3), which is result of the STPA step *Identify unsafe control actions*. The strategy S3 considers the list of unsafe control actions and their associated hazards. For instance, the UCA1 (Train door controller not provide open door command when person or obstacle is in the doorway) is associated to hazard H1.

Fig. 4 shows how the goal G12: *UCA1 (Train door controller not provide open door command when person or obstacle is in the doorway) is mitigated*. G12 must be supported by the identified scenarios and causal factors of the unsafe control action UCA1 being addressed.

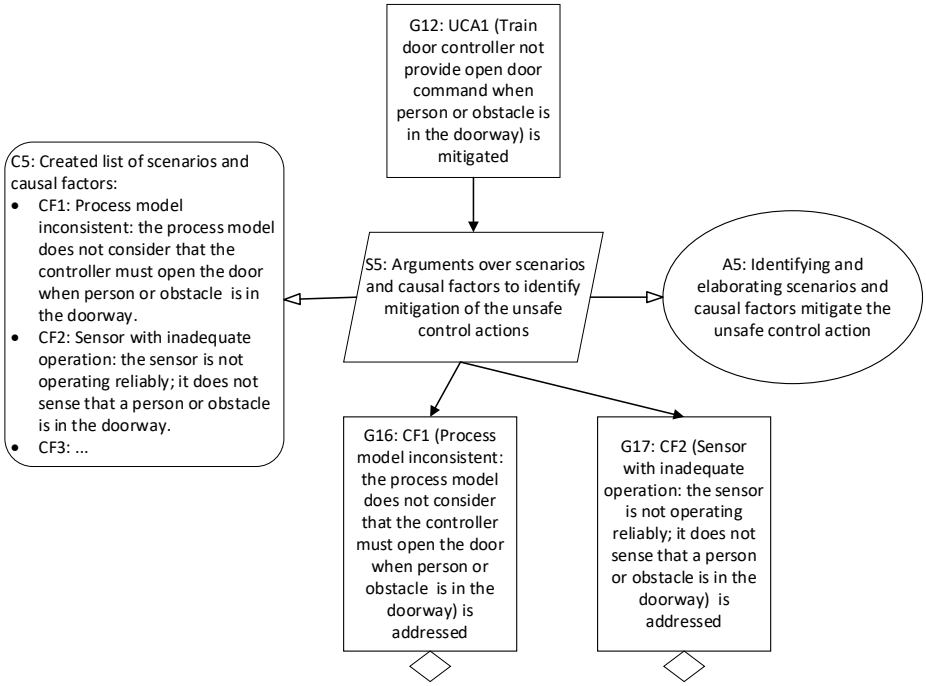

Fig. 4. Goal: UCA1 (Train door controller not provide open door command when person or obstacle is in the doorway) is mitigated.

As an example, Fig. 4 shows two scenarios and causal factors of UCA1: CF1 and CF2. The two scenarios and causal factors are addressed by the goals G16 and G17. G16 refers to CF1 (*Process model inconsistent: the process model does not consider that the controller must open the door when person or obstacle is in the doorway*) being addressed while G17 refers to CF2 (*Sensor with inadequate operation: the sensor is not operating reliably; it does not sense that a person or obstacle is in the doorway*) being addressed. Both scenarios and causal factors are addressed through the strategy S5 in the context of knowledge of the list of scenarios and causal factors (C5). This list is the result of the STPA Step *Identify Loss Scenarios*.

Fig. 5 shows how the goal G16: *CF1 (Process model inconsistent: the process model does not consider that the controller must open the door when person or obstacle is in the doorway) is addressed*. G16 is supported by one undeveloped goal G20 (*Req1 is correctly and completely implemented and verified*). Req1 is the requirement *When Door state is "person in doorway" and Door position is "Partially open" then "Open door" control action shall be issued*. As the goal is not part of STPA, it will not be analysed further. The goal is justified through the strategy S6 in the context of knowledge of the list of requirements (C7). This list is also the result of the STPA Step Identify Loss Scenarios.

Other requirements can be produced. For the causal factor CF2 of Fig. 4, the following requirements can be generated: *Probability of sensor failure per year shall be less than 0.01* (Req2) and *Sensor continuous correct operation shall be monitored* (Req3). The implementation and verification of these requirements shall be developed.

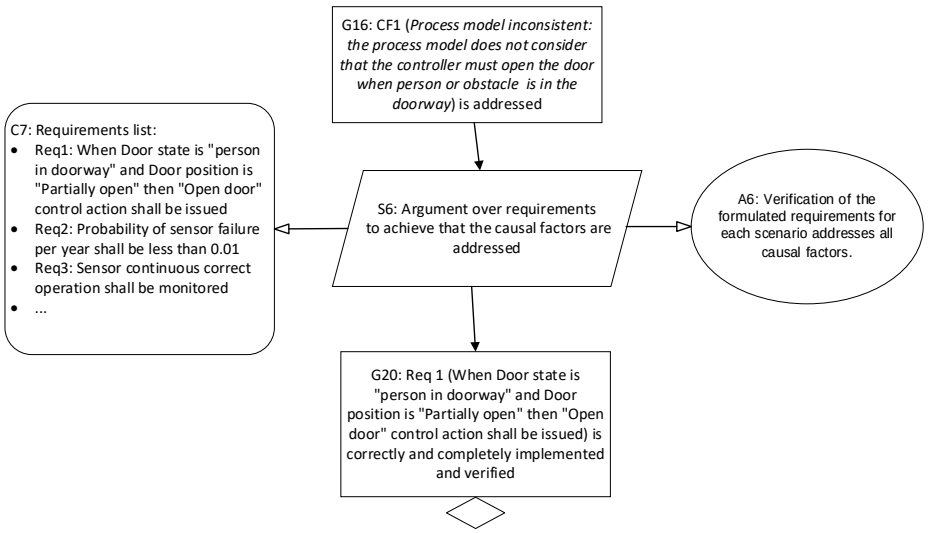

Fig. 5. Goal: CF1 (Process model inconsistent: the process model does not consider that the controller must open the door when person or obstacle is in the doorway) is addressed.

3.2 A Generic Pattern for the Joint Approach

We now move on to the generalisation of the approach, which uses a pattern [13] and elaborates a GSN created for generic safety assurance using STPA for hazard analysis.

Fig. **6** shows the use of patterns, creating a generic GSN for this purpose. The presented GSN pattern refers to an arbitrary system, i.e. the goal G1 must be instantiated for a specific system (system X). G1 is addressed by arguing on accidents and hazards through STPA (S1). The strategy S1 can only be executed in the context of knowledge

of identified accidents and hazards (C1). G3 (mitigation of a hazard) must be instantiated for all identified hazards of system X using STPA. G3 has multiplicity *m* (number of hazards). The goals G5, G7, and G9 in **Fig. 6** must then be instantiated and further developed.

4 Concluding Remarks

In this paper, we presented the use of GSN for safety assurance in combination with STPA for hazard analysis, and illustrated it using a Train Door Controller System. The presented generic GSN pattern will aid documenting that a system is free from unacceptable risks leading to accidents identified by STPA through mitigating the identified hazards.

The GSN safety case documentation complements STPA, which is a complex hazard analysis technique employed in the concept stage of development. STPA involves many steps, which are not easy to follow and understand. Embedding the outcomes of the STPA analysis in GSN hopefully helps understanding how the steps of STPA can support the ultimate safety claim in later certification stages.

Conversely, through the GSN documentation we are able to identify the contexts that make the STPA claims justifiable. They include A*ccident list and hazard list (C1), Safety control structure (C2), Unsafe control action list (C3), List of Scenarios and causal factors (C5),* and *List of requirements (C7).* These contexts are the product of the STPA analysis and are critical to make the claim that "the system is free from unacceptable risks leading to accidents". These contexts may be used as certification goals to be verified in later stages of the life cycle.

Patterns provide a suitable means to foster systematic artefact reuse and aid in the development of new safety cases. We believe that the elaborated generic pattern can help the documentation of assurances cases of any system using GSN and STPA.

In the TDC STPA analysis, there are 13 unsafe control actions and 3 safe control actions that if not followed are unsafe. These unsafe situations result in dozens of scenarios and causal factors, which result in dozens of GSNs. This is obviously a consequence of the system complexity which would lead to a multiplicity of documents to be reviewed or analysed using supporting tools, no matter which pattern would be deployed.

Future works include applying the presented pattern to more examples, including more realistic cases. A different and clearly interesting direction of work that we are currently pursuing is to combine with meta-modelling from the model-based development approaches, and also use the relevant verification results as evidence that would enrich the overall safety case when documented with GSN.

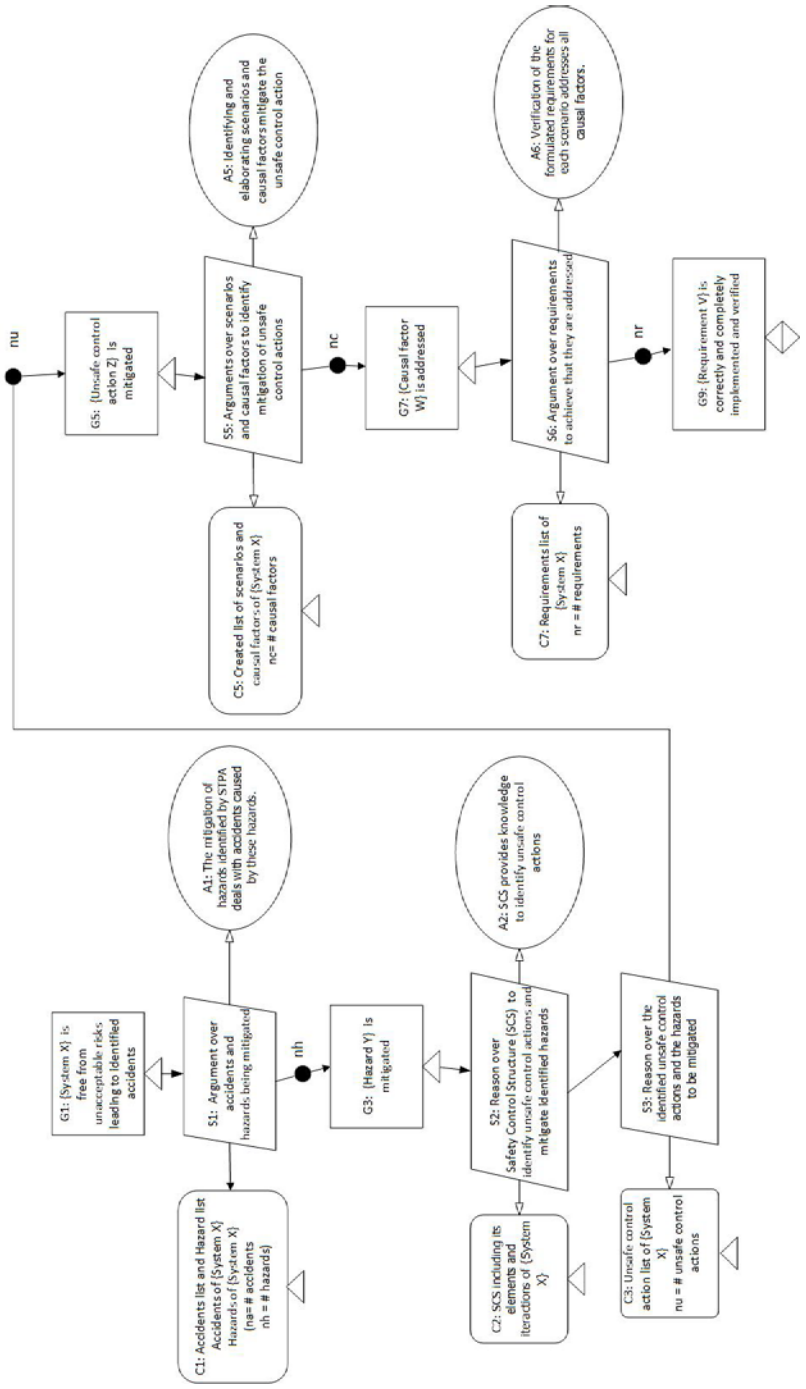

Fig. 6. GSN pattern for System X is free from unacceptable risks leading to accidents.

Acknowledgements. The work of the first author was supported by CNPq (grant numbers 403921/2016-3 and 306186/2018-7). The work of the second author was supported by the national projects on aeronautics (NFFP7-04890) and the research centre on Resilient Information and Control Systems (www.rics.se).

References

1. Kelly, T.: Arguing Safety – A Systematic Approach to Managing Safety Cases, PhD Thesis, Department of Computer Science, University of York (1998).
2. Kelly, T., Weaver, R.: The goal structuring notation - a safety argument notation. In: Proceedings of the Dependable Systems and Networks Workshop on Assurance Cases. (2004).
3. Leveson, N.: Engineering a safer world: Systems thinking applied to safety. MIT Press (2011).
4. Leveson, N., Thomas, J.: STPA Handbook, https://psas.scripts.mit.edu/home/, last accessed 2019/5/6 (2018).
5. Thomas, J.: STPA in Industry Standards, https://psas.scripts.mit.edu/home/2019-stamp-workshop-presentations/, last accessed 2019/5/6 (2019).
6. Rinehart, D., Knight, J., Rowanhill, J.: Current Practices in Constructing and Evaluating Assurance Cases with Applications to Aviation. NASA/CR– 2015-218678 (2015).
7. The Assurance Case Working Group: Goal Structuring Notation Community Standard (Version 2). https://scsc.uk/scsc-141B, last accessed 2019/5/6 (2018).
8. Rinehart, D., Knight, J., Rowanhill, J.: Understanding what it means for assurance cases to 'work'. NASA/CR–2017-219582, (2017).
9. Goodenough, J., Barry, M.: Evaluating Hazard Mitigations with Dependability Cases, AIAA 2009-1943, AIAA Infotech@Aerospace Conference. Seattle, Washington, (2009).
10. Hugues J., Delange J.: Model-Based Design and Automated Validation of ARINC653 Architectures Using the AADL. In: Cyber-Physical System Design from an Architecture Analysis Viewpoint. Springer, Singapore (2017).
11. Denney, E., Pai, G.: Tool support for assurance case development. Automated Software Engineering 25(3), 435-499 (2018).
12. Thomas, J.: Extending and automating a systems-theoretic hazard analysis for requirements generation and analysis (Doctoral dissertation, MIT) (2013).
13. Kelly, J., McDermid, J.: Safety Case Construction and Reuse using Patterns. In: Proceedings of 16th International Conference on Computer Safety, Reliability and Security (SAFECOMP) (1997).