# Analysing the Safety of Decision-Making in Autonomous Systems

Matt Osborne(✉), Richard Hawkins, and John McDermid

Assuring Autonomy International Programme, Department of Computer Science, University of York, Deramore Lane, York YO10 5GH, UK
{matthew.osborne,richard.hawkins,john.mcdermid}@york.ac.uk

**Abstract.** We characterise an autonomous system as one that has the capability to take decisions independently from human control. This independent and autonomous decision making could give rise to new hazards or hazard causes not present in an equivalent human-controlled system, e.g. through lack of human real-world understanding. Despite the increased adoption of autonomous systems there has been a dearth of research in the area of safety analysis and assurance of decision-making for autonomous systems. This paper is intended to be a first step to fill this gap. We compare and contrast the differing causal models of autonomous and non-autonomous systems, and build on existing safety engineering techniques in order to define a process (Decision Safety Analysis) for the analysis of autonomous decision-making. We show, using a real-world example, how this process supports the development of safety requirements to mitigate hazardous scenarios.

**Keywords:** Decision-making · Autonomous systems · Safety analysis

## 1 Introduction

As the use of autonomous systems (AS) for safety-related tasks continues to increase, safety-related decision-making has consequently started to transfer from the human to the AS. There is a clear and pressing need to assure the safety of (the AS making) those decisions. There are well established safety analysis approaches that have been shown to be effective in assuring the safety of traditional systems. In this paper we investigate how autonomous decision-making challenges the use of these existing approaches [1]. We identify a lack of understanding of how these approaches may be applied effectively to autonomous decision-making. For example, there are existing techniques for the analysis of human error and erroneous human decision-making but it is not clear that these could be applied to decision-making *by AS*. We therefore propose a process for analysing autonomous decision-making (Decision Safety Analysis (DSA)) that addresses these limitations.

This work is funded by the Assuring Autonomy International Programme https://www.york.ac.uk/assuring-autonomy.


M. Trapp et al. (Eds.): SAFECOMP 2022, LNCS 13414, pp. 3–16, 2022.
https://doi.org/10.1007/978-3-031-14835-4_1

This paper makes the following contributions:

1. We provide a process for analysing the safety of decisions made by an AS
2. We show how the process can be used to specifically focus further, efficient safety analyses of the AS
3. We demonstrate how the outcomes of the process can help to elicit safety requirements in mitigation of unsafe decisions that could be made by an AS.

We present the background and discuss the problem space in Sect. 2 before presenting our proposed DSA approach in Sect. 3. We present an evaluation of the process in Sect. 4, before describing the results, wider applications, and future work in Sect. 5.

## 2 Background

AS can be characterised as systems that have the capability to take decisions free from human control. From a safety perspective it is therefore the ability of an AS to make *safe* decisions that is of primary concern. It is crucial where the actions of an AS may lead to hazardous events, that such decisions are analysed for their safety impact and sufficient mitigations, or barriers, put in place.

The decision-making of an AS could give rise to new causal/failure paths that would not be present in an equivalent, human-controlled system (such as an autonomous robot operating in a typical office environment being 'unaware' of the dangers presented by blind corners, or water on a floor). Alternatively, autonomous decision-making could bring new causes to existing hazards (for example an office robot failing to detect a door that is comprised of transparent material). Figure 1 shows a representation of an accident model for a system. The system is represented as an agent that must:

- Sense the environment in which it operates using exteroceptive sensors
- Understand the information provided from the sensors (and other information) in order to create a model of the environment
- Decide how the system should respond based on its environment model
- Act in order to implement the decision made.

This is a continuous process for the system as it responds to changes in the environment by updating its understanding in order to make new decisions. For the case represented by the grey boxes at the top of Fig. 1, both the understanding and deciding aspects are dealt with by a human who forms a mental model of the environment from the information presented by the sensors and then, based on that mental model, decides on the best option to ensure the system meets its operational objectives in a sufficiently safe manner. Figure 1 illustrates that some of the actions the human may choose could result in an accident. The system would be designed with a combination of human and system checks that are intended to prevent failures resulting in accident outcomes. These are represented in Fig. 1 as barriers at multiple points in the model.

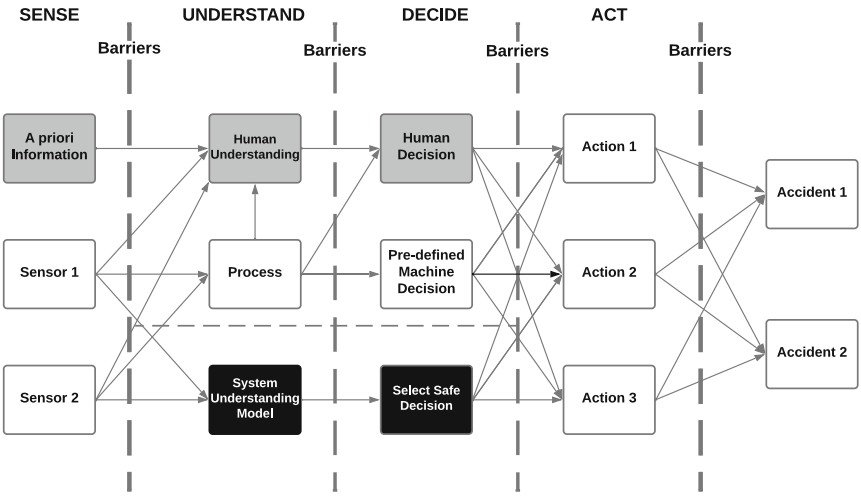


**Fig. 1.** A causal model of accidents for different types of system

For a traditional software-controlled system, as shown in Fig. 1, information is processed by the system (the unshaded boxes) and decisions either made by the human, or suggested by the system, with human oversight. The decisions made by the software are in fact pre-determined rules which have already been hard-coded into the design by a human. Analysis and assurance of safe decisions is primarily a human factors issue for such traditional systems - through analysis of human behaviour and mitigation through human factors measures such as procedures, training and supervision (although it does depend on the veracity of information provided by the system).

Figure 1 shows a similar causal model for an AS, represented with dark boxes. *In this case, only the bottom line of the understand and decide elements are relevant.* Here the system has primary responsibility for understanding the environment, creating an accurate model, and making safe decisions. A key difference here, as well as the removal of human over-sight, is the fact that the decisions are not hard-coded at design time as they were for traditional software systems [2], rather the system is given autonomy to determine the best action, given its understanding of the environment. Although the level of autonomy given to an AS can vary, we focus here on the case where there is no human oversight. From a safety perspective this represents a significant challenge for two important reasons.

Firstly, for traditional systems, humans have a particularly important role in dealing with unanticipated or unusual situations that the system may encounter. A human operator is able to use their contextual knowledge and general intelligence to react safely to unexpected occurrences. This innate ability to generalise often plays a crucial role in ensuring safe decisions are taken. Although machines may present a high level of artificial intelligence, this intelligence is typically very

narrow in nature [15], meaning that they can perform well for very specific tasks, but their ability to generalise to unanticipated situations is limited. From a safety perspective this potentially opens up a large new set of hazard causes.

Secondly, safe decision-making can no longer be treated as a human factors issue. Once the human is removed, it becomes a purely technical issue. This requires that the safety of decision-making be brought more explicitly into the system safety engineering process in a manner that it never was previously. In recent decades there has been extensive research on evaluating the process of human decision-making (such as [10–12], and [20]) and work assessing the human-robot collaborative space (such as [14] and [6]), however these do not help to address the challenge of autonomous decision-making, *per se*.

The focus of this paper is on addressing this challenge by developing an approach for the safety analysis of the decision-making in AS. This analysis seeks to understand the way in which hazards and accidents may arise from decisions made by an AS, and to derive safety requirements in mitigation. Whilst autonomous systems that are designed to operate within a controlled and controllable environment (such as automated passenger railways) can reasonably rely on 'classical' safety engineering techniques, a different approach is required for assuring the safety of decision-making by AS when its operating environment is more complex.

Although a lot of work has considered the implementation and verification of AS decision-making, there is an assumption that what constitutes safe behaviour is known (examples include [19] and [16] but these are by no means exceptions), we have found very little work on safety analysis and identification of mitigations for such systems. This paper begins to address that gap.

## 3 A Decision Safety Analysis Process

Key to the safe operation of an AS is the establishment of a suitably-defined Operational Domain Model (ODM) (often referred to in the automotive industry as an Operational Design Domain (ODD) [13]). The ODM defines the scope of operation within which the AS is to be shown to be acceptably safe. This will include any assumptions made, the features of the operating environment (e.g. people, road type and layout, weather conditions) which the AS is expected to sense, understand, and potentially interact with prior to making decisions as it carries out its tasks. If the ODM is insufficiently defined then the AS may encounter scenarios during its operation that were not considered during the development of the system, and which could therefore be unsafe and for which no assurance is provided. It is crucial therefore that all relevant aspects, features and interactions within and of the operational domain are defined - including those non-mission interactions [8]. Despite the importance of a sufficiently defined ODM, the assurance of an ODM is out of the scope of this paper.

Use cases can be created for each of the tasks which identify the elements of the ODM which the AS must understand and with which it may interact. Examination of these use cases reveals the occasions when key decisions must be

made by an AS. These decisions must be modelled and analysed to determine the nature of any hazardous scenario that may arise as a result of the AS decision.

We consider hazardous scenarios to be special cases of the Operating Scenarios for the AS, which are identified as those which could result in an unsafe outcome. 'Scenarios' describe the combination of the AS Operating Scenario and the relevant environment variables. Potentially hazardous scenarios for an AS arise due to decisions taken which are unsafe in a given environmental state when performing a particular operating scenario (the same decisions might be safe in other circumstances). Hazardous scenarios for the AS can therefore be described using the general form:

***<AS operating scenario><relevant environment variables>AND <decision>***, where:

- An AS Operating Scenario describes what task(s) the AS is undertaking
- A Relevant Environment Variable is **one or more** features of the environment relevant to the decision point
- The Decision is the selected course of action as a result of the scenario and relevant environment variables.

As an example, for an autonomous passenger shuttle undertaking the task of navigating a (UK) roundabout, we can identify a decision point for whether the shuttle should enter the roundabout. For this case, an example of a 'relevant' environment variable would be a cyclist on the roundabout to the right of the AS. A pedestrian on the footpath 20 m behind the AS would not be considered relevant as they will not influence the decision taken by the AS. Other variables could concern the road state, or weather conditions at the time a decision is required to be made.

Decisions taken by the AS can only be in relation to variables that the system can control, i.e. speed and/or direction - as the environmental variables are outside of the control of the AS. As such, the options for the passenger shuttle at this 'decision point' are:

1. Enter the roundabout (at variations in speed)
2. Stop and wait.

The approach we present in this paper can be used to identify hazardous scenarios by considering the real world state in combination with the belief state of the AS, and each of the options at the decision point, e.g. the 2 options for the shuttle identified above. This analysis would thus identify hazardous scenarios such as:

- <the passenger shuttle is approaching a roundabout><with a cyclist on the roundabout to the vehicle's right> **AND** <the passenger shuttle enters the roundabout> or,
- <the passenger shuttle is approaching a roundabout><with no cyclist present> **AND** <the passenger shuttle stops and waits>.

Whilst the first case presents an obvious risk, the second case may not always be safe, as should the AS decide to brake rapidly and unexpectedly, a hazardous outcome may be realised in the form of a rear-end collision.

### 3.1 The Decision Safety Analysis Process

Before we discuss the DSA process in detail, we must first consider what we mean by 'decision-making' in AS. The nature of the decisions made by AS can vary enormously depending on the type of system and the application domain. For example:

- An autonomous cancer-screening system decides on the appropriate patient referral based upon information from scans and other medical data
- An autonomous vehicle decides on a safe course of action if it detects an object in its path. The decision must take account of multiple other environmental variables, such as the presence of other road users, and weather conditions.

For the DSA Process it is important to distinguish between what is actually the decisions of interest for the safety analysis and what is part of the understanding task. This distinction can be highlighted in the second example above, which can be split into two parts:

- **Understand** - is there an object in the path of the vehicle?
- **Decide** - what am I going to do about it?

For AS the complexity of the operating environment can have a much greater impact on safe behaviour than for traditional systems, as this increases the chance of unanticipated and unusual events (sometimes referred to as 'edge cases'). This can be a particular challenge since AS typically operate in highly complex environments that often cannot be fully specified at design time [3]. Whilst dealing with complex environments is not limited to AS, traditional systems operating in complex environments place a lot of reliance on the human ability to deal with any unanticipated events. With an AS we cannot rely on a human to ensure a safe state is maintained, and must rely on the AS to respond safely under all situations within the entire operating domain. This requires that the analysis incorporates consideration of the operating environment in a more systematic and explicit manner than is currently the case.

Guiochet advocates the use of Use Cases and Sequence Diagrams (and then State Charts as required) as the models for undertaking HAZOP-UML analysis [7]. However, we have found that these do not make good models for analysing AS decisions as they do not make the decisions explicit, nor do they lend themselves to methodical analysis with defined start/finish points. Instead our approach uses Activity Diagrams for each use case (as shown in Fig. 3 for the example of an autonomous robot) with the following explicit information included:

- *Decision points* (annotated 'DP' within the diamonds) identified from the Use Cases and ODM. These represent the instances where a decision must be made by an AS due to a required interaction with the environment.
- *Options* associated with each decision point (represented as circles). These represent the options that an AS **could** select for each decision point.
- *Understanding points* in the use case represent the points at which the AS requires information about a particular relevant environment variable (non 'DP' diamonds).

In addition, for each Activity Diagram it is important to explicitly model all relevant assumptions and preconditions as these must be considered as part of the analysis.

It is important that the model is a complete representation of the operating scenario, particularly that the decision and understanding points have been adequately elicited. We can gain confidence in this through utilising an explicit domain model, but further work is required, and research such as that presented in [17] has started to address the completeness of ODMs via modelling and simulations, but further work is required.

Having established the Activity Diagram(s) for the system, we then analyse that model to determine hazardous scenarios, i.e. the way in which the decision could lead to selecting an option that is unsafe given the relevant environment variables. As for many similar safety analysis tasks, we propose the use of deviation-based analysis of those decision points in order to identify plausible unsafe behaviours. We considered a number of existing deviation-based techniques that have been applied to software-based systems and could be adapted to decision analysis such as FFA/FHA [5], STPA [9], and HAZOP [4]. In particular we considered HAZOP-UML [7] which was developed as a method for analysing UML models of robot systems. In general, the use of a HAZOP-based approach does seem reasonable, yet it does not support the analysis of the decision models we propose, nor explicitly consider the impact of the operating domain as part of the analysis. In addition the HAZOP-UML approach is exhaustive but unfocused, and will therefore quickly lead to a state explosion requiring substantial analysis effort without necessarily revealing the safety issues of most concern.

We have therefore developed our DSA approach to ensure the analysis is driven by consideration of the identified decision points to establish potentially hazardous deviations. As well as identifying hazardous scenarios associated with decision points, our process also provides the focus for further, more detailed analysis using, for example, HAZOP-UML. Our process is summarised in Fig. 2 and is described below. A more detailed description is provided at [18].

**STEP 1.** The DSA requires the identification of the relevant environment variables pertinent to the scenario under analysis. These variables are identified through recourse to the ODM, as discussed earlier. One potential approach would be to use [8], but the decision model we present does not presume any particular method.

**STEP 2.** Decision Points are identified by considering the decisions required to be made as a result of the interactions between the AS and the environment. Once the decision point is identified, the options available to the AS are enumerated through considerations of the system variables, as discussed previously.

In the example at Table 1 in Sect. 3.2, as the AS can only amend system variables it has control over (speed and/or direction), we defined 4 options in this case:

1. Continue on the current path at current speed
2. Continue on the current path at reduced speed

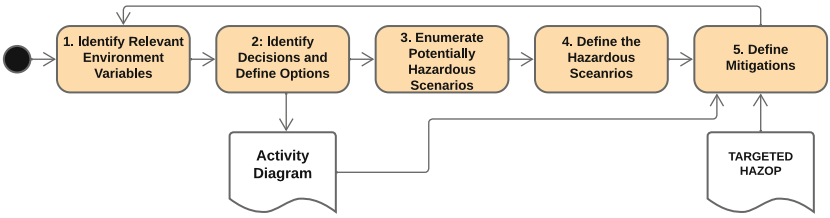


**Fig. 2.** The decision point analysis process

3. Take and alternative route at current speed
4. Stop and wait.

It is now possible to create an Activity Diagram that includes Understanding Points identified in Step 1 and Decision Points identified in Step 2. An extract of an Activity Diagram for an example system is provided at Fig. 3.

**STEP 3.** The potentially hazardous scenarios must be determined (represented in the 1st column of Table 1). This is done by firstly considering the possible options defined at Step 2 in combination with both the real world state, and the system belief state at the point at which that decision is made. Real world and system belief states are often represented as Booleans. In the example we give in Sect. 3.2, the state of 'True' for the real world state means a blind corner is present, and a state of 'True' for the system belief means that the AS "knows" this. The potentially hazardous scenarios will also consider false negatives (i.e. a real world state of 'False' and a system belief state of 'True').

For the extract in Table 1 in Sect. 3.2 we can see 14 of the scenarios which are enumerated by considering the 4 options along with the real world, and system belief states regarding the presence of a blind corner and a static object.

**STEP 4.** The outcome of each potentially hazardous scenarios enumerated in Step 3 must be determined by considering the real world impact should that scenario manifest. For any scenarios with hazardous outcomes, the hazardous scenarios can be specified using the general form described in Sect. 3.

**STEP 5.** The process then focuses on mitigating hazardous outcomes. Such mitigations could be in the form of design changes (e.g. adding a diverse sensor), or through derived safety requirements. These mitigations can be levied against the sense capability, detection capability, against the decision-making algorithm itself or on supporting infrastructure, where appropriate.

Identifying effective mitigations requires further more detailed analysis. The hazardous scenarios defined at Step 4 are used to identify the logic nodes of interest in the Activity Diagram (such as Understanding points) against which a targeted analysis such as HAZOP-UML can then be applied. The extract at Fig. 3 shows, in red font, the logical nodes of interest to which the Targeted HAZOP will be applied.

The findings of the targeted HAZOP are used to elicit further safety requirements to mitigate potential causes of hazardous scenarios. This targeted approach to undertaking the HAZOP is explained in full and illustrated on a mobile robot at [18].

It is only because we have already assessed the possible outcomes using DSA that the HAZOP can be targeted in this manner. It allows us to focus the analysis on the logic points of interest that could contribute to an erroneous decision being taken. This approach prevents the state explosion that manifests from applying HAZOP guidewords against every logical node in an Activity Diagram by allowing scenarios resulting in safe (if not always efficient) outcomes to be removed from further analysis.

In the next section we present an example of applying our approach to robots that are designed to be used for delivering small packages within one of our University buildings [18].

### 3.2 Robot Delivery System Example

We are developing a number of small robots that are capable of delivering packages around a university building. Building occupants may request a robot to come to them anywhere in the building and deliver a package to a desired destination. The building comprises 3 floors containing offices, laboratories of varying size, meeting/conference facilities, and various comfort/rest areas. A large goods lift in the centre of the building provides access to all floors, and a large shared atrium houses the reception area for visitors. The building benefits from a building management system (BMS) that provides automation and control of climate, lighting, doors, and the lift.

Through interacting with the BMS the robot is able to open/close doors and use the lift to move around the building. A central server (Robot HQ) is used to coordinate the allocation of tasks amongst the multiple robots that operate in the building at any time, but all movement around the building is controlled locally by each individual robot. In addition to the basic delivery and messenger tasks, the robots must interact with human occupants and other robots within the fabric of the building. The primary overall use case (01) for the robots is 'Package Delivery'. We broke this down to the following, more detailed use cases:

02. Request robot
03. Load package
04. Travel to destination
05. Unload package.

Within use case 04 a number of exception cases were identified, including:

A. Dynamic object in path of robot
B. Static object in path of robot
C. Forbidden zone on planned path
D. Use Lift

E. Pass through doorway
F. Blind corner on route.

Within use case 04 a number of preconditions were also identified, including:

- **Precondition1:** Robot is available
- **Precondition2:** Sender and receiver are in accessible locations
- **Precondition3:** Robot battery charge is sufficient for task.

The identified use cases relates to 4 different actors:

- Sender
- Receiver
- Building Management System
- Robot HQ.

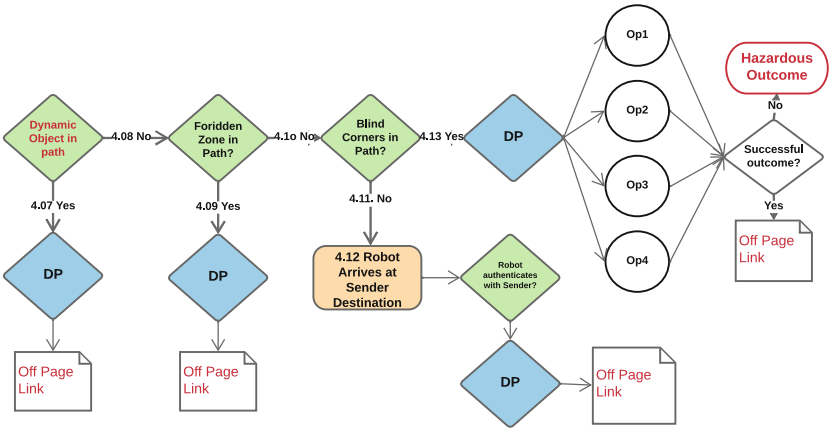


**Fig. 3.** An extract of the decision activity diagram for use case 04 - "travel to destination"

These, along with the elements of the operating environment defined in the ODM, assumptions, normal flows, alternative flows, and safety requirements/invariants represent the primary interactions that the robots make.

We have used the DSA Process to analyse the system described above. As described in Sect. 3.1 the first thing we require is a model of the decisions taken by the delivery robots. Figure 3 shows an example decision activity diagram for Use Case 04. Through consideration of the interactions of the robot with the elements of the defined ODM, a number of scenarios were identified including the robot approaching a blind corner. By blind corner we mean one the robot cannot "see" around, and these might be permanent, e.g. due to walls, or temporary, e.g. due to a bag being placed against the side of a desk blocking the normal line

**Table 1.** Extract of a decision safety analysis table

| Operational scenario: travel to destination | | | | | |
|---|---|---|---|---|---|
| Environment variables: <Robot approaching blind corner><Static object in path> | | | | | |
| Potentially hazardous scenarios | Real world state | System model belief | Option | Outcome | Safety Reqmt |
| 1 | T | T | Continue on current path at current speed | Hazardous - collision | #1 #2 #3 |
| 2 | | | Continue on current path at reduced speed | Hazardous - though severity may reduce | #2 |
| 3 | | | Take an alternative route at current speed | Correct decision | #2 #3 |
| 4 | | | Stop and wait | Safe but inefficient | |
| 5 | T | F | Continue on current path at current speed | Hazardous | #4 |
| 6 | | | Continue on current path at reduced speed | Hazardous - though severity may reduce | #5 |
| 7 | | | Take an alternative route at current speed | Safe - but predicated on erroneous understanding | #3 |
| 8 | | | Stop and wait | Safe but inefficient | |
| 9 | F | T | Continue on current path at current speed | Safe - but predicated on bug in path planning | #6 |
| 10 | | | Continue on current path at reduced speed | Safe - but predicated on bug in path planning | #6 |
| 11 | | | Take alternative route at current speed | Safe - but predicated on bug in path planning | #6 |
| 12 | | | Stop and wait | Potentially unsafe (sudden stop) | #6 |
| 13 | F | F | Continue on current path at current speed | Correct decision | |
| 14 | | | Continue on current path at reduced speed | Safe - but predicated on bug in path planning | |

| Safety requirements | |
|---|---|
| 1 | The robot shall take into account blind corners when route planning |
| 2 | The robot shall reduce its speed to 0.25 m per second when approaching a blind corner |
| 3 | The robot shall provide audio and visual alerts when approaching a blind corner |
| 4 | The Building Management System shall enforce robot speed reductions in areas of blind corners |
| 5 | The robot shall be aware of blind corners on its planned path |
| 6 | The robot shall not falsely detect the presence of blind corners |

of sight for the robot's optical sensors. Table 1 shows the results of the analysis of this scenario, showing 14 identified scenarios and their potential outcomes. Note that requirement #4 deals with permanent blind corners but will be unable to deal with temporary problems caused by placing of bags, etc. Requirement #5 requires the robot to sense blockages that create temporary blind corners – for example determining, using a depth camera, that it cannot "see" as far as normal in the relevant direction at this location. This is therefore a requirement on the

understanding component of the system, which also requires information about the layout of the building (from maps or the BMS) to detect temporary blind corners and to inform the decision-making algorithm accordingly. Details such as the above would be added as the safety requirements are allocated to the system components and refined.

This analysis elicited a number of hazardous scenarios, for example:

<the robot is travelling to its destination><approaching a blind corner with a static object in its intended path> AND <the robot continues at current speed>.

In conjunction with a targeted HAZOP we were able to identify a number of mitigations levied against both the robots and other actors (e.g. the BMS being required to enforce robot speed reductions in areas including permanent blind corners). The mitigations identified from the DSA can be seen in Table 1, and mitigations from the targeted HAZOP can be found in full at [18].

## 4 Process Evaluation

We have so far evaluated our DSA process in two ways. Firstly, we assessed the usability of the approach by applying it to a real-life case study. This showed the process was able to successfully generate a set of safety requirements in mitigation of identified hazardous scenarios. Secondly, we have evaluated the efficiency of the process by also carrying out a full HAZOP-UML analysis, and comparing the effort and outputs for both the DSA and HAZOP-UML.

When we applied our DSA process to the robots, we elicited 32 safety requirements, of which 3 were allocated to the BMS and the rest to the robots themselves. The process was simple to apply using the Activity Diagrams that had been created for the robot tasks. In carrying out this evaluation it was noted, however that well-defined use cases and a clearly structured and complete ODM were essential to the efficacy of the process. For example, if the ODM is missing any of the key elements of the operating environment, this would mean that potentially critical interactions could be missed. The challenge of specifying use cases and ODMs is widely reported ([13] for example) and requires further work which is outside the scope of this paper.

It was also noted that for the requirements definition phase, it is often necessary to specify constraints as part of the derived safety requirements. As is always the case for complex systems and environments, defining these safety constraints can be challenging. For example eliciting the required range, bearing, and detectable distance of static and dynamic objects will be influenced by the size, speed, and braking capability of the AS. Defining such constraints as part of the safety requirements is something we intend to explore further, but is out of scope for this paper.

We have described how our process enables further focused analysis of causes of unsafe decisions through the use of a targeted HAZOP that enabled safe scenarios to be excluded from the analysis. This allows a complete analysis without

the need for exhaustive state coverage. In order to test this, a full HAZOP-UML was undertaken on the entire use case for our system (considering all logic nodes in the activity diagrams). We found that this full HAZOP analysis was a very time consuming activity due to the expected state explosion (generating 834 lines of analysis). Despite this, it did not identify any safety requirements to be placed on the robots in addition to those elicited much more efficiently from applying our process and subsequent targeted HAZOP. The full use cases, DSA results, HAZOP, and safety requirements elicited in mitigation can all be found at [18].

## 5 Discussion and Conclusions

This paper has made the following contributions. Firstly, we have demonstrated a process for analysing the safety of AS decision-making. Secondly, we have demonstrated how the process can be used to facilitate a targeted approach to HAZOP, that avoids analysing safe (but perhaps inefficient) outcomes and prevents the state explosion associated with applying guidewords to every logical node of a use case. Thirdly, our approach enables the elicitation of safety requirements in mitigation of hazardous decisions made by an AS.

We have, so far, only applied the DSA approach to a single robot system in a controllable environment, and do not yet make an argument regarding the generalisability of our approach. In order to check the wider applicability of the approach we are applying the process to more complex, less controllable operating domains, including outdoor operation. This will also be further extended to consider a multi-robot system, and concurrent and consecutive decision-making. Our current case study considers robots that operate in an environment that includes humans, but it has not yet been applied to systems involving robot-human collaboration (where humans work together with the robots to fulfil tasks). We therefore also plan to apply the process to a COBOT [6] system.

We anticipate carrying out additional applications of our approach in order to further validate its efficacy and to demonstrate its wider applicability.

## References

1. Safety and ethics of autonomous systems project overview. Technical report, Royal Academy of Engineering (2020)
2. Adler, R., Feth, P., Schneider, D.: Safety engineering for autonomous vehicles. In: 2016 46th Annual IEEE/IFIP International Conference on Dependable Systems and Networks Workshop (DSN-W), pp. 200–205. IEEE (2016)
3. Burton, S., Habli, I., Lawton, T., McDermid, J., Morgan, P., Porter, Z.: Mind the gaps: assuring the safety of autonomous systems from an engineering, ethical, and legal perspective. Artif. Intell. **279**, 103201 (2020)
4. International Electrotechnical Commission: IEC 61882 (2016)
5. Ericson, C.A., et al.: Hazard Analysis Techniques for System Safety. Wiley, Hoboken (2015)

6. Gleirscher, M., Johnson, N., Karachristou, P., Calinescu, R., Law, J., Clark, J.: Challenges in the safety-security co-assurance of collaborative industrial robots. arXiv preprint arXiv:2007.11099 (2020)
7. Guiochet, J.: Hazard analysis of human-robot interactions with HAZOP-UML. Saf. Sci. **84**, 225–237 (2016)
8. Harper, C., Caleb-Solly, P.: Towards an ontological framework for environmental survey hazard analysis of autonomous systems. In: SafeAI@ AAAI (2021)
9. Ishimatsu, T., Leveson, N.G., Thomas, J., Katahira, M., Miyamoto, Y., Nakao, H.: Modeling and hazard analysis using STPA (2010)
10. Kahneman, D.: Thinking, Fast and Slow. Macmillan (2011)
11. Klein, G.A., Orasanu, J., Calderwood, R., Zsambok, C.E., et al.: Decision Making in Action: Models and Methods. Ablex Norwood, New Jersey (1993)
12. Koehler, J.J.: The influence of prior beliefs on scientific judgments of evidence quality. Organ. Behav. Hum. Decis. Process. **56**(1), 28–55 (1993)
13. Koopman, P., Fratrik, F.: How many operational design domains, objects, and events? In: SafeAI@AAAI (2019)
14. Lesage, B.M.J.R., Alexander, R.: SASSI: safety analysis using simulation-based situation coverage for cobot systems. In: Proceedings of SafeComp 2021, York (2021)
15. Marcus, G., Davis, E.: Rebooting AI: Building Artificial Intelligence We Can Trust. Vintage (2019)
16. Medrano-Berumen, C., İlhan Akbaş, M.: Validation of decision-making in artificial intelligence-based autonomous vehicles. J. Inf. Telecommun **5**(1), 83–103 (2021)
17. Oberheid, H., Hasselberg, A., Söffker, D.: Know your options-analysing human decision making in dynamic task environments with state space methods. Hum. Centred Autom. 285–300 (2011)
18. Osborne, M.: ISA Robot Safety of Decision Making. https://www-users.cs.york.ac.uk/mo705/isarobot.html
19. Stansbury, R.S., Agah, A.: A robot decision making framework using constraint programming. Artif. Intell. Rev. **38**(1), 67–83 (2012)
20. Walker, G., et al.: Modelling driver decision-making at railway level crossings using the abstraction decomposition space. Cogn. Technol. Work **23**(2), 225–237 (2021)