A Rigorous Iterative Analysis Approach for Capturing the Safety Requirements of Self-Driving Vehicle Systems

Fahad Alotaibi
ECS, University of Southampton
Southampton, U.K.
Email: F.A.Alotaibi@soton.ac.uk

Thai Son Hoang
ECS, University of Southampton
Southampton, U.K.
Email: t.s.hoang@soton.ac.uk

Michael Butler
ECS, University of Southampton
Southampton, U.K.
Email: m.j.butler@soton.ac.uk

***Abstract*—This paper presents a methodology called Rigorous Analysis Template Process (RATP) for analysing the behaviours and interactions of multiple components in a Self-Driving Vehicle (SDV) to ensure its system safety, especially when a human driver is involved as a fallback option for handling hazardous events. RATP uses Systems-Theoretic Processes Analysis (STPA) and Event-B formal method to gradually identify safety requirements and build their rigours models. The output of RATP is a set of safety requirements that guide the development of a rigorous model to maintain the system safety against identified hazardous states at different levels of abstraction. The main advantage of RATP is to allow the behaviours of a system to be analysed from a high-abstraction layer to a more detailed concrete layer.**

***Index Terms*—SDV, STPA, Event-B, Automated lane centering, Driver monitoring system**

I. Introduction and Motivation

Self-Driving Vehicles (SDVs) [1] are a type of autonomous system that rely on multiple subsystems communicating and working together to achieve a system goal, using artificial intelligence (AI) to perceive the driving environment, make driving decisions, and replace the actions of human drivers.

However, SDVs are designed with the human driver in the loop. The role of human drivers in SDVs can vary based on three design principles, including semi-automation and fallback architecture, which are used to improve system design and demonstrate autonomy levels. According to the International Society of Automotive Engineers (SAE) [2], automation levels 1 to 3 (semi-automation) involve a human driver in the dynamic driving tasks (DDTs), while automation levels 4 and 5 (high automation) do not engage a human driver in the DDTs.

One of the design challenges in intelligent systems such as SDVs is ensuring safe interactions between the SDV and human drivers, particularly when it comes to safety [1]. In some situations, SDVs may require drivers to play a fallback role in ensuring the autonomous software functions properly. Therefore, it is important to establish secure interactions between the SDV's internal components and drivers to increase trust in the safety and reliability of SDVs.

To evaluate inadequate interactions in safety-critical systems such as SDV systems, numerous hazard identification methods have been developed in the domain of safety-critical systems [3]. Among these, Systems Theoretic Process Analysis (STPA) has been shown to be effective in examining complex system component interactions [4]. However, STPA lacks a formal aspect, and formal methods such as Event-B can complement STPA by proving that hazardous events have been mitigated at the design level. Event-B [5] is a modelling language that supports both theorem proving and model checking and has several comprehensive tools to enhance system analysis.

This paper presents a new approach based on existing work [6] to identify Safety Requirements (SRs) and build a formal model gradually through refinement steps. The approach, called Rigorous Analysis Template Process (RATP), combines STPA and Event-B for system behaviour analysis. The approach is validated using an Automated Lane Centring (ALC) case study. Section II provides background on the STPA, the Event-B, and the introduction to our case study. Section III gives an overview of our approach and the processes involved. Section IV gives a conclusion and related works of our approach.

II. Background

A. Systems Theoretic Process Analysis (STPA)

The goal of Systems Theoretic Process Analysis (STPA) is to prevent the occurrence of losses by mitigating the hazards based on the enforcement of the safety requirements/constraints on the dynamic behaviours of a system [7]. Hazards are potential causes of losses during the development of a target system, and losses may occur when safety constraints/requirements on system component interactions or behaviours are violated. STPA is represented in a horizontal structure that can be summarized into four steps: 1) identification of the purpose of the analysis, 2) inadequate safety control structure, 3) unsafe control actions, and 4) missing behaviors. We have chosen STPA for several reasons. Firstly, it focuses on the dynamic control of a system. Secondly, it can be adopted at an early stage of design. Thirdly, it is an iterative analysis methodology that allows for the gradual building of architectural changes to cover new or missing aspects of a target system.

B. Event-B

Event-B [8] is a formal modelling language used to specify and verify the behaviour of systems. An Event-B machine corresponds to a transition system where the states are represented by the variables v (constrained by invariants I(v) and the transition are represented by events). An event is *'an atomic transition'* that changes the states of the system. The transition state of an event is constrained through the guards and actions. For instance, for an event e with parameters t, the guard of the event can be written as G(t,v), and the action of the event can be represented as $v := E(t, v)$. An event e can only be enabled when its guard G (t, v) holds for some parameter t and its effects on variable v are specified by the action E(t,v).

$$e == \textbf{any}\ t\ \textbf{where}\ G(t,v)\ \textbf{then}\ v := E(t,v)\ \textbf{end}$$

We have chosen Event-B for several reasons. Firstly, Event-B allows for modelling the design aspects of the system at the system level, rather than just at a software level. Secondly, Event-B gradually introduces the system specifications into the formal model through refinement techniques, which enables the traceability of critical requirements with associated formal representations.

C. Case Study

The Automated Lane Centring (ALC) system is a well-known Advanced Driver Assistance System (ADAS), classified as automation levels 2 and 3. Both academic researchers and industry practitioners have highlighted the diversity of the ALC system's features. Some researchers emphasize that the ALC aims to identify the vehicle's desired path and modify the steering to keep the vehicle in the center of a target lane [9, 10]. Others identify further features of the ALC system, such as sending warning messages to the driver to pay attention and take corrective driving action [11]. Additionally, the ALC works with the Driver Monitoring System (DMS) to ensure the driver's responsiveness when the system may require human intervention in critical driving scenarios [12, 13].

D. Automation aspects of the ALC system

In a previous work [6], the ALC system component diagram (see Fig. 1) was constructed in relation to how it was implemented in the SDV system. It focused on the automation aspect of the system and explained the functions of the different modules that involve in the SDV, which are the perception, decision, and control modules. In more detail, the perception module detects a path that the SDV can follow with a certain level of confidence. Then, the decision module modifies the steering angle of the vehicle, based on the detected path, to move the SDV to a new target position. Finally, the control module takes the modified steering angle and applies it to the vehicle's components to move the SDV to the desired target position.

Furthermore, the SDV system is responsible for monitoring the awareness level of the human driver, especially when the ALC system requests intervention. To address this, the Driver Monitoring System (DMS) was introduced to verify the driver's awareness level by detecting the hands-on steering wheel or sensitive monitoring feature. The intervention request is then divided into warning messages and auditory notifications, which alert the driver to either immediately intervene or inform the driver that the ALC will release control of the SDV.

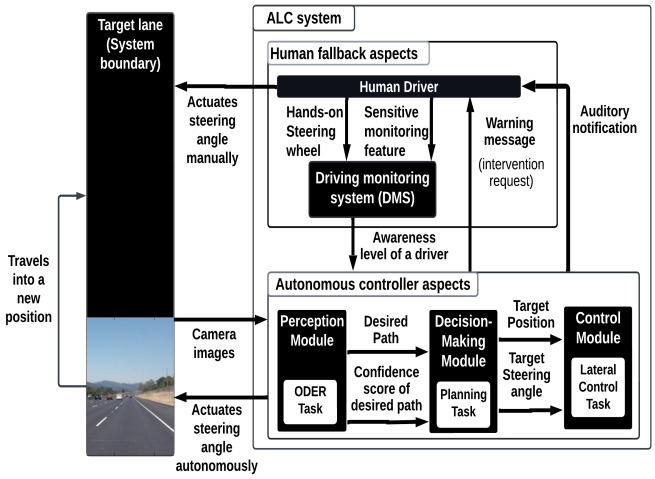

Fig. 1: The automation aspects involved in the ALC system, adapted from the existing work [6].

III. Rigorous Analysis Template Process

This paper presents the Rigorous Analysis Template Process (RATP) for analysing the behaviours of SDVs, such as the ALC system. The RATP uses a systematic iterative analysis process to identify safety requirements and build a formal model based on the automation aspects from existing work [6], as mentioned earlier in Fig. 1. The iterative systematic steps of RATP are summarised in Section III-A, while the application of RATP in the ALC case study is highlighted in Section III-B.

A. The systematic steps of the RATP

The RATP is a systematic iterative process consisting of five steps for each iteration, as shown in Fig. 2 on the following page[1]. The steps are as follows:

Step 1. Instantiating Component Diagrams: A series of hierarchical system boundary diagrams is provided to support the development process at different levels of abstraction. The system components are gradually introduced via several iterations of the process in order to show how these component interacts with its boundary.

Step 2. Identifying the Purpose of Analysis: The purpose of analysis is identified in each iteration based on the identification of safety constraints to mitigate the occurrence of potential hazardous states. The identification of system hazards is associated with the system losses. The system losses and hazards can be illustrated based on an instantiated component

[1]The notation used in Fig. 2 is a standard notation for describing work processes called 'solution-patterns', (see https://vvpatterns.ait.ac.at/about-vv-patterns/).

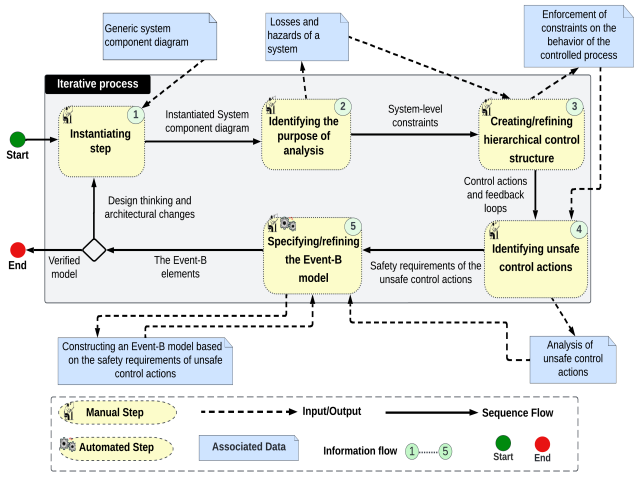

Fig. 2: Process overview of the RATP approach.

diagram from the previous step, focusing on how a system interacts with its boundary.

Step 3. Creating/Refining Hierarchical Control Structure: After identifying the system hazards and their safety constraints, the control structure is constructed. An effective control structure will enforce safety constraints on the behaviour of the overall system as a set of Control Actions (CAs) and Feedback Loops (FLs) between autonomous and human controllers.

Step 4. Identifying Unsafe Control Actions: The identification of Unsafe Control Actions (UCAs) and their safety requirements will be gradually driven based on how the control structure is constructed in the previous step. The Safety Requirements (SRs) can be defined as the functional requirements that mitigate the occurrence of system hazards, where the causal factors or system actions that contribute to the these hazards are demonstrated.

Step 5. Specifying/Refining the Event-B Model: The behaviours of a system are modelled to ensure the enforcement of safety requirements based on a set of unsafe CAs and FLs identified in the previous step.

The RATP approach uses an iterative analysis process from Steps 1 to 5 to re-perform the analysis and refine the formal model. This methodology offers several benefits, such as allowing the abstraction behaviours of a system to be analysed and modelled, supporting traceability of system losses and hazards, and providing a precise syntax for safety invariants to be formally verified for consistency.

B. Automated Lane Centring Case Study

The RATP approach is utilised in five layers for the ALC case study, with each layer representing a single iterative process that adds more complexity to the system. The aim of each layer is specific, ranging from providing a high-level overview of the ALC system to modelling the driving scenarios where human intervention is required. The results of each layer, including system losses, hazards, safety requirements, control structures, and formal models, are available through a provided link [2]. Due to space constraints, we only provide some aspects of the RATP steps for the ALC case study as follows. In particular, we only show the results obtained at different steps. Notice that the results are obtained through 5 layers (iterations) of the process described in Fig. 2.

Step 1. Instantiation: An abstract system component diagram is constructed in Fig. 3. It shows how the ALC system interacts with its system boundary, i.e., the SDV may change its position into a new position inside its target lane. The boundary of a system is specified as a target lane, where the ALC system is supposed to operate.

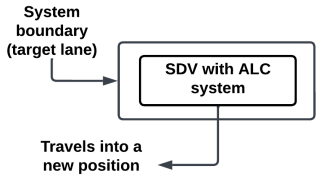

Fig. 3: System component diagram involved in a high-abstraction layer.

A concrete system component diagram is demonstrated in Fig. 4. It breaks down the SDV system into two main controllers: autonomous and human fallback, where the automation aspects involved in the ALC system (mentioned earlier in Fig. 1) are instantiated. For instance, the autonomous controller receives incoming sensing images to preforms the autonomous operations such as perception, while the human fallback integrates with the DMS to ensure the awareness level of the human driver.

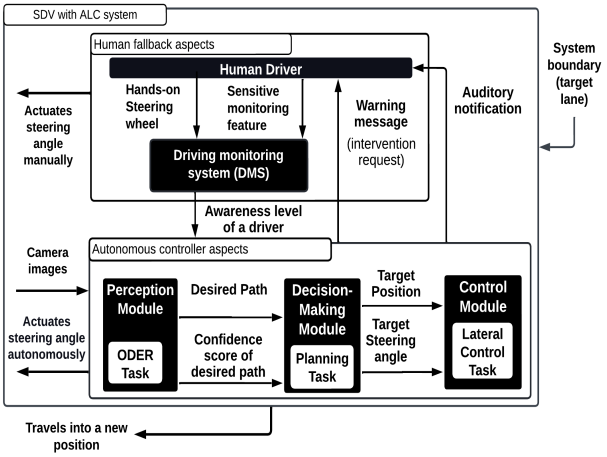

Fig. 4: System component diagram involved in a more detailed concrete layer.

Step 2. The purpose of analysis: At the high-level of abstraction, the primary concern of the system is to keep the vehicle within its lane, and the main loss of the system (**L0**) is defined as 'the SDV collides with an object outside its target lane'. The primary hazard associated with L0 is **H0**: 'the SDV travels into a position outside its target lane'. To prevent this hazard,

[2]The results of the RATP approach for an ALC case study are available as a PDF document and a Rodin archive at https://t.ly/TfVw.

the safety constraint (**SC0**) is established that 'the SDV must always be located inside the target lane'.

At the concrete level, different system losses and hazards are identified to explore how the automation aspects of the ALC system interact with autonomous and human controllers. For example, a refined system loss (**L1**) is defined as 'the human driver does not take over control of the vehicle when the ALC requests intervention'. Based on L1, two system hazards are identified: **H1**, which refers to 'the human driver not paying attention to the autonomous operations of the ALC system', and **H2**, which refers to 'the perception module identifying the desired path that places the SDV outside its target lane'. To prevent these hazards, a new safety constraint (**SC1**) is introduced that 'the ALC must ensure that the SDV stays within its target lane when the ALC issues a request to intervene'.

Step 3. Control structure: The abstract control structure presented in Fig. 5 has two main components: the SDV with an ALC system and the target lane. To represent the interaction between the SDV and the target lane, the high-level safety constraint (**SC0**) is translated into Control Action (**CA1**) and Feedback loop (**F1**). Firstly, **CA1** allows the SDV to change its position. Secondly, **F1** monitors the SDV's position to ensure it remains in the target lane.

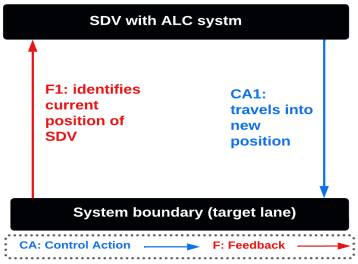

Fig. 5: Control structure involved in an abstraction level.

At the concrete level, the control structure is further refined in Fig. 6 by identifying more than twenty-driven STPA requirements and constraints from the abstract to the concrete level. For example, the autonomous controller must compute the desired path (**CA4**) based on incoming images (**F3**). Meanwhile, the driver's awareness level (**CA8**) is determined by the DMS, which detects the hands-on steering wheel (**CA9**) and uses a sensitive monitoring feature (**CA10**).

In accordance with **SC1**, the ALC system sends an auditory notification (**F8**) if the driver does not respond to a request to intervene (**F6**, **F7**), i.e, **F6** indicates that the ALC system is uncertain about whether the identified path will keep the SDV in the target lane or cause it to move outside the lane.

Step 4. Identifying unsafe control actions: The Unsafe Control Actions (UCAs) of **CA1** is demonstrated in Fig. 7. The *UCA1.1* indicates that a system might travel to a new position outside the target lane; therefore, the identification of SRs for mitigating the occurrence of **H0** are **SR0**: the SDV must travel into a new position inside the target lane.

Based on **H1**, the UCAs of DMS are presented in Fig. 8. For instance, the *UCA 2.2* indicates that the ALC system may start working before the DMS verifies the awareness level of

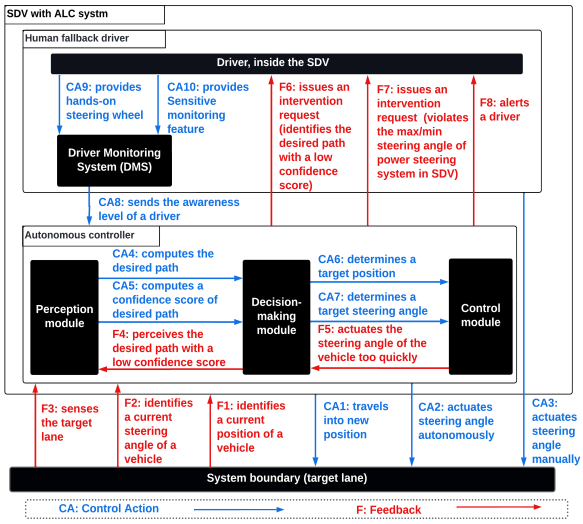

Fig. 6: Control structure involved in a concrete level.

Control Action (CA)	Not providing CA leads to hazards	Providing CA leads to hazards	CA applied early, late, or out of order	CA stopped too soon or applied too long
CA1	N/A	**UCA1.1:** The SDV may travel from a position inside the lane to a new position outside the lane.	N/A	N/A
Causal Factor				
Unsafe movement of a vehicle.				

Fig. 7: Unsafe control actions involved in CA1.

a driver. Therefore, the the potential SRs associated with the DMS (**H1**) can be identified as follows: **SR1**: to activate the ALC system, the DMS must ensure the awareness level of the driver, **SR2**: the DMS must compute the awareness level of a driver based on the hands-on steering wheel and a sensitive monitoring feature and **SR3**: If the driver does not provide the hands on steering wheel and a sensitive monitoring feature, the ALC system will be immediately deactivated.

Control Action (CA)	Not providing CA leads to hazards	Providing CA leads to hazards	CA applied early, late, or out of order	CA stopped too soon or applied too long
CA9 and CA10	N/A	**UCA2.1:** Autonomous controller performs its autonomous operations while the awareness level of a driver is unverified/unknown.	**UCA2.2:** ALC performs its autonomous operations before the DMS verifies a driver's awareness level.	N/A
Causal Factor				
A driver is unaware of the autonomous operations of a system.				

Fig. 8: Unsafe control actions involved in CA9 and CA10.

Furthermore, the UCAs of a perception module (**H2**) is shown in Fig. 9. Several UCAs are identified, i.e., the autonomous controller may identify the desired path without sensing the target lane (*UCA3.1*). Therefore, the potential SRs associated with the perception component can be identified as follows: **SR4:** the perception component must update the sensing information (images) according to the current position of an SDV, **SR5:** the perception component must use an incoming image to identify the desired path for keeping an SDV inside the detected left and right lane lines in the target

lane and **SR6:** if the perception module identifies the desired path with a low confidence score, the ALC system will issue a request to intervene.

Control Action (CA)	Not providing CA leads to hazards	Providing CA leads to hazards	CA applied early, late, or out of order	CA stopped too soon or applied too long
CA4 and CA5	**UCA3.1:** Autonomous controller may identify the desired path without sensing the target lane (F3).	**UCA3.2:** Desired path (CA4) does not update. **UCA3.3:** Desired path (CA4) identified according to a low confidence score (CA5).	**UCA3.4:** Desired path(CA4) identified according to wrong/late sensing data (F3).	N/A
Causal Factor				
Failure in the identification of a desired path				

Fig. 9: Unsafe control actions involved in CA4 and CA5.

Step 5. Event-B model: At the high-level of abstraction, we use a constant Lane to specify the all positions of SDVs in the lane, i.e., $Lane \subseteq POSITION$. The physical position of the SDV (**CA1**) is modelled as a variable SDV_POSITION_env, with a safety invariant @safety: $SDV\_POSITION\_env \in Lane$, i.e., the SDV must always be within the lane (**SC0**). In addition, the initial position of an SDV (**F1**) is modelled as a constant init_position, where the initialisation position of SDV must be inside the lane, i.e., $init\_position \in Lane$. Finally, the move event abstractly captures a movement of an SDV into a new position as follows:

```
event move
any new_position where
// SR0: the SDV must travel into a new position inside the target lane
 @grd1 : new_position ∈ Lane
then
 @act1 : SDV_POSITION_env := new_position
end
```

At the concrete level, the formal model undergoes multiple refinements to introduce the automation aspects of the ALC system. For instance, the first refinement is designed to showcase the autonomous operations of the ALC system. A constant function camera is defined to capture the sensing information in multiple positions, i.e., $camera \in POSITION \rightarrow IMAGE$. An important specification of an SDV is demonstrated in a consistency invariant @consistency:$IMAGE\_env = camera(SDV\_POSITION\_env)$, i.e., the SDV system must receive an incoming image (**F3**) based on the physical position of the SDV (**SR4**). The operations of an ALC system are organised into five stages: Perception, Decision, Control, Intervention and AutonomousDriving. These stages are defined in variable stage_ac, with gluing invariant @glu_inv1: $ALC\_Status = ON \Rightarrow stage\_ac \in STAGE$, i.e., the ALC system might be in any stage when a system is activated. For instance, the ALC system receives an incoming image (**F3**) from the camera system in order to identify the desired path (**CA4**) and its confidence score (**CA5**); therefore, the constant functions 'Seen_image' and 'OEDR_task' represent the computation made by the autonomous controller to identify the desired path based on an incoming image.

```
/* show the expected results of a seen image as the detected left lane line, detected right
     lane line and the confidence scores (probabilities)*/
Seen_image ∈ (IMAGE × LeftLane × RightLane) → Confidence_score
/* show how ALC accomplishes the OEDR task based on the seen image to identify a
     desired path (set of positions) */
OEDR_task ∈ (IMAGE × LeftLane × RightLane × Confidence_score) → ℙ(
     POSITION)
```

In addition, the ALC abstractly identifies the desired path (**CA4**) and its confidence score (**CA5**) as follows:

```
event perception
any leftLane rightLane where
@grd1 : leftLane ⊆ LeftLane
@grd2 : rightLane ⊆ RightLane
@grd3 : ALC_Status = ON
@grd4 : stage_ac = Perception
then
/* obtain an image from camera based on SDV's position */
@act1 : IMAGE_env := camera (SDV_POSITION_env)
@act2 : leftLanePoints_ac :∈ LeftLane
@act3 : rightLanePoints_ac :∈ RightLane
/* identify confidence score based on the detection results of a received image as the left
     /right lane lines*/
@act4 : confidentScore_ac := Seen_image (IMAGE_env ↦
     leftLanePoints_ac ↦ rightLanePoints_ac)
/* (SR5): recognise the desired path (set of position ) based on the left/right lane lines,
     image and confidence score */
@act5 : desirePath_ac := OEDR_task (IMAGE_env ↦ leftLanePoints_ac ↦
     rightLanePoints_ac ↦ confidentScore_ac)
/* change a stage of ALC to be in Decision */
@act6 : stage_ac := Decision
end
```

While the stage of the ALC system changed to Decision, the ALC system might issue a request to intervene if the low confidence score (**F4**) is used to compute the desired path (**SR6**).

```
event lowConfidenceScore_interven
where
@grd1 : stage_ac = Decision
/*(SR6): if confidence score is detected, system issued a request to intervene*/
grd2 : confidenceScore_ac < 80 // less 80 is an example of low confidence score
then
/*issued a request to intervene*/
@act1 : stage_ac := Intervention
end
```

Furthermore, the second refinement involves more details about how the DMS ensures the awareness level of a driver during the performance of automotive operations. A Boolean variable awarenessLevel_ah (**CA8**) is introduced to compute the awareness level of a driver (**SR2**) based on how the DMS detects the handsOnSteeringWheel_hf (**CA9**) and a sensitiveMonitoringFeature_hf (**CA10**); therefore, a new invariant is written, as follows:

```
awarenessLevel_achf = TRUE ⇒ (handsOnSteeringWheel_hf = TRUE ∧
     sensitiveMonitoringFeature_hf =TRUE)
```

In order to ensure a driver is receptive to any intervention scenarios, a new invariant is added to verify the awareness level of a driver when the ALC performs its operations **(SR1)**. Therefore, the DMS must verify the awareness level of a driver when the ALC system is at any stage, such as Intervention etc.

```
awarenessLevel_achf = TRUE ⇒ stage_ac ∈ {Perception , Decision, Control
     , Intervention, AutonomousDriving }
```

IV. Conclusion and Related Work

This paper presents an iterative analysis methodology called the Rigorous Analysis Template Process (RATP). The RATP adopts STPA to identify the safety requirements and the Event-B formal method to provide assurance about the consistency of the safety requirements obtained from the STPA. Based on the template prototype designed in existing work [6], the RATP allows the behaviours of a system to be analysed from

a high-abstraction layer to a more detailed concrete layer. In addition, the RATP helped a robust traceability of the system losses, hazards and their corresponding safety requirements into associated formal representations. The ALC case study was presented to validate the RATP approach. Future work will focus on applying the RATP approach to further case studies in the domain of the automotive field. The ALC case study can be extended to investigate how the autonomous controller accomplishes the lateral (steering) and longitudinal (acceleraion/deceleration) tasks in conjunction with ensuring the responsiveness of a human fallback driver.

a) Related Work on Safety Properties of SDVs in STPA: Traditionally, automotive designers have built their safety strategy on the concept that the human fallback driver is essentially responsible for safety [14]. STPA might integrate within these standards to identify other failures, such as inadequate component interactions, which may occur within or without the occurrence of component failures [15]. Abdulkhaleq et al. [16] proposed a dependable architecture for SDVs based on STPA, where the SDV system is decomposed into three levels: vehicle level, system level, and component level. Abdulkhaleq and Wagner [17] also applied STPA to the Adaptive Cruise Control (ACC) in order to show that the result of STPA is applicable for identifying the potential accident scenarios, e.g., human decision-making errors and component interaction accidents. Hanneet et al. [18] applied STPA to a Lane Keeping Assist (LKA) system to derive safety constraints and requirements. However, they consider only the LKA system, and the driven requirements did not cover the interactions of LKA with the DMS or other autonomous systems such as ACC.

b) Related Work on Combining Formal Methods with STPA: Colley and Butler [19] developed a method to demonstrate and formally analyse the critical requirements (artefacts) generated by the STPA analysis method, through the use of the Event-B formal method and its Rodin toolset. Similarly, Howard et al. [20] adopted STPA to generate the critical requirements, while the formal models were constructed in Event-B to verify that those critical requirements completely mitigate against vulnerable system states. Dghaym et al. [21] extended the work of [19, 20] to develop a compositional approach to elicit the critical requirements for autonomous functions and then formalised those critical requirements into Event-B models. Moreover, STPA has also been used with other formal methods. Hata et al. [22] formally modelled the constraints/requirements obtained from STPA as pre and post conditions in VDM++. Thomas and Leveson [23] described a formal syntax for unsafe control actions, which are recognised as a result of applying STPA. In our previous work [6], we presented a rigorous approach to capture high-level system component interactions in the ALC system, mainly ensuring the safe component interactions between a human fallback driver and an SDV. This paper adopts an iterative process RATP based on STPA and Event-B, which allows the behaviours of a system to be analysed from an abstraction layer to a concrete layer.

References

[1] Y. Xing, C. Lv, D. Cao, and P. Hang, "Toward human-vehicle collaboration: Review and perspectives on human-centered collaborative automated driving," *Transportation research part C: emerging technologies*, vol. 128, p. 103199, 2021.

[2] SAE, "Taxonomy and definitions for terms related to on-road motor vehicle automated driving systems," *SAE Standard J*, vol. 3016, pp. 1–16, 2014.

[3] C. H. Fleming, M. Spencer, J. Thomas, N. Leveson, and C. Wilkinson, "Safety assurance in nextgen and complex transportation systems," *Safety science*, vol. 55, pp. 173–187, 2013.

[4] R. Bongirwar, "Leveraging systems theoretic process analysis (STPA) for efficient iso 26262 compliance," SAE Technical Paper, Tech. Rep., 2021.

[5] J.-R. Abrial, *Modeling in Event-B: system and software engineering*. Cambridge University Press, 2010.

[6] F. Alotaibi, T. S. Hoang, and M. Butler, "High-level rigorous template for analysing safety properties of self-driving vehicle systems," in *2022 IEEE 46th Annual Computers, Software, and Applications Conference (COMPSAC)*. IEEE, 2022, pp. 1643–1648.

[7] N. G. Leveson and J. P. Thomas, "STPA handbook," *Cambridge, MA, USA*, 2018.

[8] T. S. Hoang, "An introduction to Event-B modelling method," *Industrial Deployment of System Engineering Methods*, pp. 211–236, 2013.

[9] R. Michelmore, M. Kwiatkowska, and Y. Gal, "Evaluating uncertainty quantification in end-to-end autonomous driving control," *arXiv preprint arXiv:1811.06817*, 2018.

[10] Comma.ai, "Openpilot: an open source driver assistance system," June 2022. [Online]. Available: https://github.com/commaai/openpilot/

[11] C. Becker, L. Yount, S. Rozen-Levy, J. Brewer *et al.*, "Functional safety assessment of an automated lane centering system," United States. Department of Transportation. National Highway Traffic Safety ..., Tech. Rep., 2018.

[12] Volvo, "Tips for using pilot assist, car functions, volvo support." May 2022. [Online]. Available: https://www.volvocars.com/uk/support/topics/use-your-car/car-functions/tips-for-using-pilot-assist

[13] O'Kane, "Tesla starts using in-car camera for autopilot driver monitoring." August 2021. [Online]. Available: https://www.theverge.com/2021/5/27/22457430/tesla-in-car-camera-driver-monitoring-system

[14] P. Koopman, U. Ferrell, F. Fratrik, and M. Wagner, "A safety standard approach for fully autonomous vehicles," in *International Conference on Computer Safety, Reliability, and Security*, 2019, pp. 326–332.

[15] Q. Van Eikema Hommes *et al.*, "Assessment of safety standards for automotive electronic control systems," United States. National Highway Traffic Safety Administration, Tech. Rep., 2016.

[16] A. Abdulkhaleq, D. Lammering, S. Wagner, J. Röder, N. Balbierer, L. Ramsauer, T. Raste, and H. Boehmert, "A systematic approach based on STPA for developing a dependable architecture for fully automated driving vehicles," *Procedia Engineering*, vol. 179, pp. 41–51, 2017.

[17] A. Abdulkhaleq and S. Wagner, "Experiences with applying STPA to software-intensive systems in the automotive domain," 2013.

[18] H. S. Mahajan, T. Bradley, and S. Pasricha, "Application of systems theoretic process analysis to a lane keeping assist system," *Reliability Engineering & System Safety*, vol. 167, pp. 177–183, 2017.

[19] J. Colley and M. Butler, "A formal, systematic approach to STPA using event-b refinement and proof," 2013.

[20] G. Howard, M. Butler, J. Colley, and V. Sassone, "Formal analysis of safety and security requirements of critical systems supported by an extended STPA methodology," in *2017 IEEE European Symposium on Security and Privacy Workshops*. IEEE, 2017, pp. 174–180.

[21] D. Dghaym, T. S. Hoang, S. R. Turnock, M. Butler, J. Downes, and B. Pritchard, "An STPA-based formal composition framework for trustworthy autonomous maritime systems," *Safety science*, vol. 136, p. 105139, 2021.

[22] A. Hata, K. Araki, S. Kusakabe, Y. Omori, and H.-H. Lin, "Using hazard analysis STAMP/STPA in developing model-oriented formal specification toward reliable cloud service," in *2015 International Conference on Platform Technology and Service*. IEEE, 2015, pp. 23–24.

[23] J. Thomas and N. Leveson, "Generating formal model-based safety requirements for complex, software-and human-intensive systems," in *Proceedings of the Twenty-First Safety-Critical Systems Symposium, Bristol, UK*. Safety-Critical Systems Club, 2013.