Sentinel: A Safety Architecture for SAE J3016 Level 5 Autonomous Vehicles

Spencer R Deevy, Alan Wassyng, Mark Lawford, Vera Pantelic, Richard Paige

Sentinel: A Safety Architecture for SAE J3016 Level 5 Autonomous Vehicles

Spencer R. Deevy*, Alan Wassyng*, Mark Lawford*, Vera Pantelic*, Richard Paige*
*McMaster Centre for Software Certification, McMaster University, Canada

***Abstract*— The rapid adoption of artificial intelligence (AI) techniques is being used to develop increasingly capable autonomous vehicles. While the major focus has been on improving the performance and accuracy of AI techniques applied to autonomous vehicles, development towards functional safety has been lagging behind. This paper proposes Sentinel, a fault-tolerant safety architecture, designed to mitigate safety concerns surrounding AI techniques employed by upcoming SAE J3016 Level 5 autonomous vehicles. The architecture draws inspiration from existing autonomous vehicle architectures as well as architectures in the related domains of AI and organic computing. An assurance case was constructed to demonstrate that Sentinel provides high level features that support compliance with SAE J3016 Level 5 autonomy and that Sentinel meets or exceeds the safety of other autonomous vehicle architectures.**

I. Introduction

Artificial intelligence (AI) has been the major enabler of increased autonomy in autonomous vehicles [1], [2], [3], [4]. In particular, the highest level of autonomy according to the Society of Automotive Engineers (SAE) J3016 Levels of Driving Automation [5], Level 5, assumes full autonomy under all possible environmental conditions with no driver intervention. Vehicles employ AI techniques such as machine learning (ML) to achieve dynamic driving tasks (DDTs) such as lane detection, lane keeping, vehicle and pedestrian detection and predictive path planning under all environmental conditions. Coping with all types of roadways, all weather conditions, and any number of potentially unknown obstacles, increases the need for robust AI on-board such vehicles. Further, for Level 5 autonomous vehicles, degradation below full autonomous operation is not possible in the absence of human intervention. Therefore, a significant concern to be addressed with Level 5 autonomous vehicles is ensuring their safety under all potential hazardous scenarios. In particular, to the best of the authors' knowledge, no publicly available architecture exists for achieving Level 5 autonomy, namely, a safety architecture that recognizes the need for AI techniques to achieve Level 5 features while maintaining vehicle safety.

This paper proposes a novel safety architecture for Level 5 autonomous vehicles, *Sentinel*. Sentinel leverages selected features of several modern architectures proposed in domains such as artificial intelligence, autonomous vehicles, and organic computing. It also uses traditional safety patterns. We reason about and demonstrate the effectiveness of the Sentinel architecture using an assurance case. The assurance case argues that Sentinel provides design features that support compliance with SAE J3016 Level 5 autonomy. Further, it argues that Sentinel meets or exceeds the safety of other autonomous vehicle architectures. The architecture provides a high-level framework to enable autonomous vehicle functionality, whereas lower-level design decisions and implementation methods are left as future work.

II. New Architecture: Sentinel

The proposed architecture, Sentinel, is shown in Figure 1. At the highest level of abstraction, the architecture consists of twelve software modules. The twelve modules can be grouped into three categories: Operational, Support, and Safeguard. Operational modules (indicated by blue boxes in Figure 1) are responsible for basic driving behaviour of the vehicle and consist of the sensors, perception, planning, plan execution, actuator interface and actuators modules. Support modules (indicated by green boxes in Figure 1) provide several services to the operational modules to aid in vehicle safety and performance but are not strictly required for regular vehicle operation. Modules in this category are the agent interface, data management, simulation, and proactive safety modules. Safeguard modules (indicated by orange boxes in Figure 1) provide a safe control bypass from sensors to actuators, thereby allowing the vehicle to safely react to unanticipated hazardous conditions. The reactive safety module and the control mode selector constitute the safeguard category.

A. Operational Modules

Operational modules enable the base functionality of autonomous driving, where each of the modules is essential to achieving this goal and no additional functionality beyond these capabilities is present in these modules. Readers will note that these modules essentially make up the common sense-plan-act chain commonly used across several industries, discussed in [6].

The sensory module encapsulates the on-board sensor suite and corresponding software components interfacing those sensors to other vehicle software. The data collection and any preprocessing performed on raw sensor data belong to this module. Additionally, sensor error detection is captured in this module, to be used by safety modules for initiating a DDT. A similar pattern is used in the cognitive ADAS architecture described in [6], which contains a path from the sensor layer, through an Internet of Things (IoT) connection, to the failure detection module. Upon detection of a sensor failure a signal is sent to the cognition layer to initiate a fallback DDT action. Sentinel's sensory module is responsible for producing

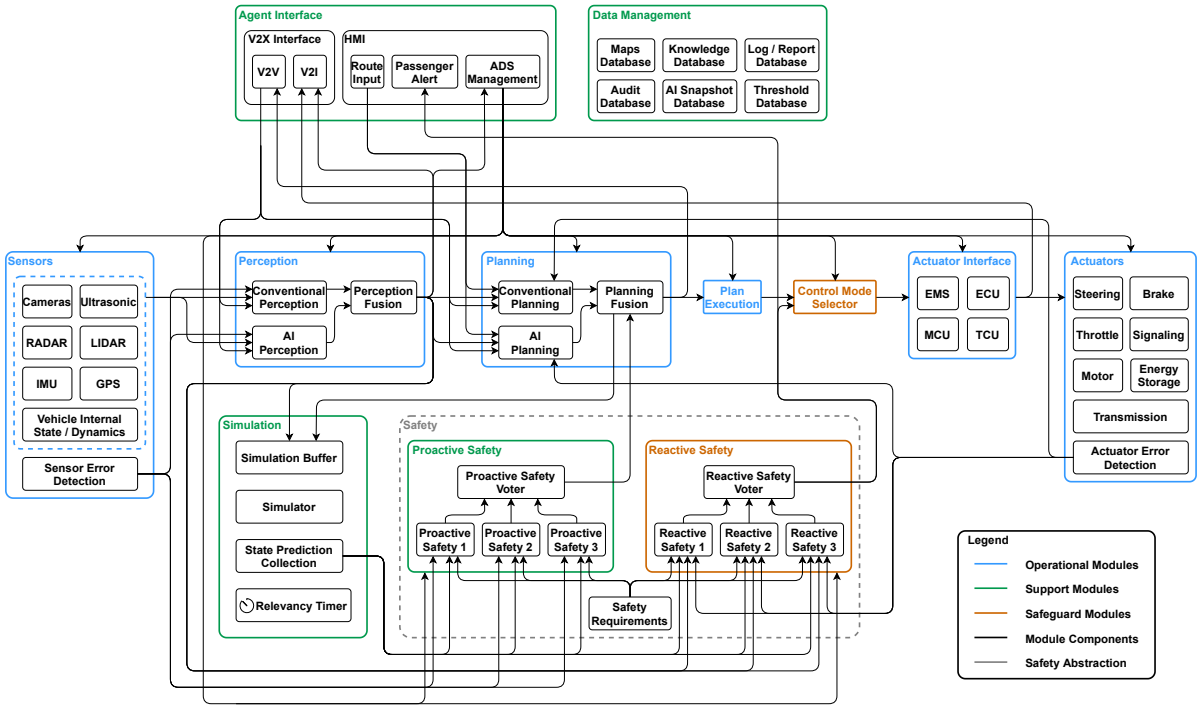

Fig. 1. Sentinel Architecture

a set of raw and/or preprocessed sensor data to be sent to the perception module for sensor fusion and reality model generation tasks.

Broadly speaking, the perception module is responsible for constructing a model of reality to be utilized by the planning module. This task normally involves sensor fusion and sensor data consolidation and involves multiple features in the environment being identified. The perception module makes use of both artificial intelligence algorithms and traditional algorithms for perception, with the intent of partial function overlap between any artificial intelligence algorithm and at least one traditional algorithm, taking inspiration from the arguing machines framework described in [7]. While the arguing machines framework utilizes an AI algorithm for redundancy, Sentinel makes use of several traditional algorithms to provide partial function overlap with any in-use AI algorithm, utilizing Dempster-Shafer theory to combine (probabilistic) outputs of all algorithms into a final belief in the combined outputs of each algorithm, similar to the method used in [8]. This is done to reduce the risk of any singular AI component failing by performing cross validation against traditional alternatives.

Another distinction between the proposed solution and existing ones is that reality model(s) produced by each perception algorithm should express each object and/or characteristic of the environment with a probabilistic belief in the validity of the object/characteristic being true. Probabilistic decision making in autonomous vehicles utilizing AI algorithms can be seen in [9], [8], [3]. This is done so that the perceptual fusion component can properly combine probabilities of each feature to construct a final, more accurate probabilistic reality model. This reality model describes perceived objects within a perceived environment, while ascribing a probability of that belief to each object in the model. These objects need not only be ascribed one definition and probability. For instance an object may have a 97% probability of being an object in the environment while also having a 70% probability of being a bicycle. This overlay of probabilities allows for more nuanced and appropriate decision-making in later stages of the sense-plan-act chain.

The planning module is responsible for generating a set of routes and associated high-level actions, to be utilized by the plan execution module to generate high-level plan execution commands. These tasks require a minimum of the aforementioned reality model and a specified goal the plan aims to achieve. Several other modules are utilized to improve the safety of generated plans. The majority of existing planning techniques solely rely on the current reality model and goal for plan generation. Humans, however, do not plan vehicle routes this way. Humans take note of other vehicles and entities in the environment, track their trajectories, and plan according to a predicted future state. For example, if a lane-change manoeuvre is desired and another vehicle is approaching in the other lane at a high speed, a human would often wait for the vehicle to pass before initiating a lane change. Operating on only the current reality model may lead to an unsafe condition where the lane change manoeuvre is undertaken while the vehicle in the neighbouring lane is approaching. To mitigate this, some recent research has been done on predictive planning methods [1], [2], rather than the traditional reactive approach. To enable predictive planning, we first send the collection of generated routes to a simulation module, where the environmental state and routes are used to predict near-term future states of entities in the environment. Fusion of several planning algorithms can then be performed to consolidate the plans with the highest

probability of success over the current and near-term future states into a singular plan to then be executed by the vehicle. This information is then sent to the proactive safety module for a safety analysis to be conducted on those future states. The idea of running near-term predicted environmental states through a safety check for vehicle planning is novel.

The plan execution module is responsible for generating a final set of control execution commands that enacts one of the provided action plans. These commands are to be sent through the control mode selector module to the actuator interface module, to be utilized for actuator control. To reduce the set of action plans to a singular output set of high-level control signals, our novel proposal is that the action plans be processed in order of the highest probability of belief and lowest predicted risk combination. This permits plans to be generated with the most accurate data, so that plans with the lowest predicted risk to take precedence.

The actuator interface module encapsulates the actuator-specific electronic control unit (ECU) modules responsible for executing specific portions of the path execution signals provided by the control mode selector module. Path execution commands are sent to corresponding ECUs and converted into timed low-level control signals to be sent to the actuators.

The actuation module encapsulates the on-board actuator suite and corresponding software components interfacing those actuators to other vehicle software. Additionally, actuator error detection is captured in this module, to be used by safety modules for initiating a DDT fallback.

B. Support Modules

Support modules enhance and/or aid the operational modules with additional information and features that allow for more nuanced and appropriate decision-making, especially when faced with uncertainty and predictive planning.

The agent interface module is responsible for allowing direct communication with external agents, namely other autonomous vehicles, infrastructure, and passengers. This allows the vehicle to make more informed decisions and aids in passenger safety and satisfaction. This is a design feature that has been employed by modern architectures such as the multi-layer observer/controller architecture and the cognitive ADAS architecture from [6]. It has been shown to improve the validity and reliability of on-board perception and path planning in autonomous vehicles [10], [11].

The data management module is responsible for storing and allowing access to data pertaining to vehicle operation. Several components provide a support role to the sense-plan-act chain, such as knowledge databases and maps databases for localization and route planning tasks, as seen in [12]. An additional AI snapshot database has been added based on overfitting issues pertaining to reinforcement learning (RL), if such algorithms are deemed necessary for Level 5 functionality. To avoid extreme overfitting cases, this component may take snapshots of the current parameters of each RL algorithm being used. Should the performance of any RL algorithm falter, the algorithm could be rolled back to a previous iteration. This sacrifices some of the knowledge learned by the algorithm since the last snapshot but may provide a mechanism to avoid the overfitting issue. The rollback feature is inspired by similar functionality applied to genetic algorithms in the organic computing architecture in [13].

The simulation module is responsible for producing a set of predicted near-term future states for each action plan it simulates. As inputs, it receives perceptual and planning information from the respective modules in the sense-plan act chain. Utilizing this information, object tracking and predictive planning mechanisms enable near-term future locations of objects in the environment. Near-term simulation has been successfully applied to several autonomous control problems in [3], [13], and is a core feature of the organic computing architecture in [13].

The proactive safety module is the second stage in the predictive planning process. While predictive planning is not a new concept, the notion of running predicted states through a safety check in order to calculate predicted risk levels is novel. This module receives predicted near-term future environmental states from the simulation module and a set of safety requirements common to both the proactive and reactive safety modules. This module is responsible for performing a safety analysis on the set of predicted future states linked to each candidate action plan initiated by the planning module. The safety analysis associated with each action plan candidate involves checking predicted states against a set of safety requirements to determine the number and severity of safety violations in each set of predicted states. This is done for each set of predicted states and the final collection of safety reports is sent to the planning module. The planning module can then use this information to decide on a subset of candidate action plans that pose the least amount of risk to future scenarios. The proactive safety module utilizes a classical redundancy pattern, triple modular redundancy (TMR).

C. Safeguard Modules

Safeguard modules are responsible for bypassing the sense-plan-act chain in the event that the sense-plan-act chain fails to predict and react to a hazardous event. This time-criticality and failure of the sense-plan-act chain requires instantaneous intervention in order to bring the vehicle to a safe state. Since the Level 5 autonomy does not include a human driver, an automated controller is responsible for this task.

The reactive safety module is responsible for sensing unsafe conditions, providing continual fail-safe control actions, and assuming control over actuators in the event of unsafe conditions. This module receives environmental information from the perception module and safety requirements from the common safety requirements data store. The limitation of this approach is that it relies on perceptual data rather than raw sensor data, where the former may have inaccuracies introduced via the AI algorithms contained within the perception module. This was chosen since the reactive safety module may not have enough information to make an appropriate decision with raw sensor data. The distinction between a pedestrian

and a plastic bag, for example, is an important one in a reactive scenario and must be dealt with accordingly. Unless predictable alternatives to the currently used machine learning algorithms can be developed to handle these situations, we will likely have to make reactionary decisions based on perceptual data rather than raw sensor data. The reactive safety module uses the perceptual data and the safety requirements shared with the proactive safety module to determine whether a safety violation has occurred, and constructs a suitable response. In all situations the reactive safety module must always have a set of control actions ready should there be too little time for re-computation. In the event of a safety violation, the reactive safety module sends its set of control actions to the control mode selector, enabling the safety bypass of the sense-plan-act chain. Once a safe state is reached, the control signals are no longer sent to the control mode selector. This design coincides with the *simplex pattern*, used to achieve graceful degradation of a system in the event of a failure [14], [15].

The control mode selector is responsible for switching control between the main sense-plan-act chain and the reactive safety module. This selector enables control for the reactive safety module so long as it is receiving control actions from reactive safety. Once the control signals from the reactive safety module cease, control is switched back to the main sense-plan-act chain.

D. Architecture Evaluation

An implementation of the Sentinel architecture has not yet been realized. To provide some basis for confidence in the architecture we constructed an *assurance case* as a qualitative analysis; see [16]. We reason about the capability of the Sentinel architecture in enabling SAE J3016 Level 5 compliance, and that Sentinel improves upon safety features offered by existing autonomous vehicle architectures. The assurance case breaks down the recommendations provided in the SAE J3016 definition of Level 5 autonomy, decomposing each into claims any autonomous vehicle architecture would need to meet in order to achieve compliance. We terminate the argument at the point where an architecture, such as Sentinel, provides the architectural components necessary to achieve the stated claim, as well notes on some of the features and acceptance criteria required of the architectural component(s). Upon implementation of Sentinel, or any other Level 5 autonomous vehicle architecture, the assurance case can be continued from the bottom-most claims, arguing how the implementation provides sufficient aforementioned Level 5 functionality.

III. Conclusions

Level 5 autonomy is still in the early phases of research and development. Extensive research still needs to be conducted for many of the techniques discussed in the Sentinel architecture before they can be used in highly autonomous vehicles. The Sentinel architecture, or at least the integration of several of the aforementioned modules, would need to be implemented before a quantitative evaluation of the architecture is possible. Thus, that evaluation is left as future work.

Sentinel is clearly resource intensive and challenging to implement with today's technology. However, we should not be planning the safety of fully autonomous vehicles on the basis of the status quo.

It may appear that our view is too software centric. We realize that safety is a system property. Our goal is to implement a safety architecture that ensures safety of the vehicle and its environment. Autonomous vehicles will be controlled by software. Sentinel is primarily a software intensive system design that uses hardware and software to plan and monitor a vehicle's behaviour in its known environment, and to safeguard the vehicle, its occupants, and its environment in cases even when it encounters unforeseen events and conditions.

References

[1] C. Katrakazas, M. Quddus, W.-H. Chen, and L. Deka, "Real-time motion planning methods for autonomous on-road driving: State-of-the-art and future research directions," *Transportation Research Part C: Emerging Technologies*, vol. 60, pp. 416–442, 11 2015.

[2] M. Shimosaka, K. Nishi, J. Sato, and H. Kataoka, "Predicting driving behavior using inverse reinforcement learning with multiple reward functions towards environmental diversity," in *2015 IEEE Intelligent Vehicles Symposium (IV)*, Seoul, KR, 6 2015, pp. 567–572.

[3] R. McAllister, Y. Gal, A. Kendall, M. van der Wilk, A. Shah, R. Cipolla, and A. Weller, "Concrete problems for autonomous vehicle safety: Advantages of bayesian deep learning," in *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence (IJCAI-17)*, Melbourne, AU, 2017, pp. 4745–4753.

[4] R. Salay, R. Queiroz, and K. Czarnecki, "An analysis of iso 26262: Machine learning and safety in automotive software," in *WCX World Congress Experience*. SAE International, 4 2018.

[5] S. International, *Taxonomy and Definitions for Terms Related to Driving Automation Systems for On-Road Motor Vehicles*, 6 2018.

[6] K. P. Divakarla, A. Emadi, and S. Razavi, "A cognitive advanced driver assistance systems architecture for autonomous-capable electrified vehicles," *IEEE Transactions on Transportation Electrification*, vol. 5, no. 1, pp. 48–58, 3 2019.

[7] L. Fridman, L. Ding, B. Jenik, and B. Reimer, "Arguing machines: Human supervision of black box ai systems that make life-critical decisions," in *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*. Long Beach, CA, USA: IEEE, 6 2019, pp. 1335–1343.

[8] G. Gündüz, Ç. Yaman, A. U. Peker, and T. Acarman, "Driving pattern fusion using dempster-shafer theory for fuzzy driving risk level assessment," in *2017 IEEE Intelligent Vehicles Symposium (IV)*, Los Angeles, CA, USA, 6 2017, pp. 595–599.

[9] F. Garcia, D. Martin, A. De La Escalera, and J. M. Armingol, "Sensor fusion methodology for vehicle detection," *IEEE Intelligent Transportation Systems Magazine*, vol. 9, no. 1, pp. 123–133, 2017.

[10] B. Peters and K. Rick, "I spy: V2x technology can provide information about objects and conditions although they may be obscured," *Vision Zero International*, 1 2017.

[11] J. Wang, Y. Shao, Y. Ge, and R. Yu, "A survey of vehicle to everything (v2x) testing," *Sensors*, vol. 19, no. 2, 2019.

[12] A. C. Serban, E. Poll, and J. Visser, "A standard driven software architecture for fully autonomous vehicles," in *2018 IEEE International Conference on Software Architecture Companion (ICSA-C)*, Seattle, WA, USA, 4 2018, pp. 120–127.

[13] C. Müller-Schloer and S. Tomforde, *Organic Computing – Technical Systems for Survival in the Real World*, ser. Autonomic Systems. Birkhäuser, 12 2017.

[14] A. Desai, S. Ghosh, S. A. Seshia, N. Shankar, and A. Tiwari, "Soter: A runtime assurance framework for programming safe robotics systems," in *2019 49th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*, 2019, pp. 138–150.

[15] S. A. Shah, "Safe-av: A fault tolerant safety architecture for autonomous vehicles," Master's thesis, McMaster University, Hamilton, CA, 2019.

[16] S. Deevy, "Sentinel: A software architecture for safe artificial intelligence in autonomous vehicles," Master's thesis, McMaster University, 12 2019.