# Safety-Counter-Player: Utilizing potentially unsafe capabilities in safety-critical systems

Mario Trapp, Benjamin Herd, Benedikt Frank

# Safety-Counter-Player: Utilizing potentially unsafe capabilities in safety-critical systems

1st Mario Trapp
*Technical University of Munich*
Munich, Germany
mario.trapp@tum.de

2nd Benjamin Herd
*Fraunhofer Institute for Cognitive Systems IKS*
Munich, Germany
benjamin.herd@iks.fraunhofer.de

2nd Benedikt Rank
*Technical University of Munich*
Munich, Germany
benedikt.rank@tum.de

***Abstract*—In safety-critical systems, integrating machine learning components (MLCs) presents significant challenges in balancing safety with functional performance. Engineers strive to harness machine learning to enhance both system functionality and safety. However, they face obstacles in ensuring sound safety assurance for these machine learning components. This paper proposes a novel architecture that distinguishes between two roles: the safety-player, which is responsible for making critical safety interventions, and the counter-player, which focuses on optimizing functional performance. By permitting the safety-player to intervene only when absolutely necessary, the counter-player is allowed greater freedom in its operations. This separation not only improves performance but also maintains safety, fostering a more effective interaction between safety, comfort, and overall system utility.**

***Index Terms*—safety, utility, safety architecture, safety monitor, automated driving systems**

## I. Introduction

The safety of machine learning components (MLCs) is of paramount importance, and safety monitors are essential for ensuring their safety. However, a significant drawback of these monitors is their often overly conservative nature, which can lead to a substantial reduction in overall system performance. This conservativeness arises from the necessity for safety monitors to maintain simplicity but also from compromises that often mix safety with comfort instead of allowing for a clear separation of concerns.

To illustrate the latter point, consider the example of braking distances. When these distances are determined, considerations for passenger comfort can lead to a deceleration that falls well below the vehicle's physical capabilities. If we were to disregard comfort and even accept the possibility of minor accidents –as long as no one is harmed– we could unlock a greater degree of operational freedom. While the aim is still to avoid permanent harsh braking maneuvers or accidents, such occurrences could be framed as a reliability issue, assuming we ensure that the risk of harm remains at an acceptable level.

Furthermore, technologies like machine learning can greatly enhance our understanding of the driving environment, contributing to tactical safety. For instance, the vehicle could predict whether a car in front is about to turn onto another road or analyze brake lights and traffic signals to forecast traffic flow. While this would allow for a more anticipatory and tactical style of driving, similar to that of a human driver, integrating advanced machine learning and the necessary perceptual capabilities into the safety-critical path remains a challenging task.

Therefore, the main question we are exploring in this paper is whether a stricter separation between comfort and reliability versus actual safety can create the necessary freedom to minimize safety measures. This would allow us to utilize sophisticated capabilities within the reliability path to optimize the system's utility without compromising safety.

This paper presents initial ideas for a novel counter-player architecture pattern, which introduces concepts that diverge from established paradigms. Although these ideas are still in their early stages of development, they offer a fresh perspective on the topic. The counter-player pattern features two distinct participants: the safety-player and the counter-player. The safety-player is designed to intervene only at critical moments, employing the most severe countermeasures -—such as maximum deceleration—- when reaching a true point of no return. On the other hand, the counter-player adapts the system's functionality to optimize utility, which may include factors like traveling speed, driving comfort, or minimizing jerk. Most importantly, when the safety-player intervenes, the utility is set to zero. Hence, the counter-player counters the safety-player to maximize utility by continuously adapting the system to improve performance while minimizing the probability of the safety-player's intervention.

By clearly separating the safety-player from its counter-player, we establish a distinct safety-critical path. The safety-player acts as the last line of defense, while the counter-player functions outside this critical path as a high-reliability component. This separation allows the counter-player to leverage sophisticated, AI-based perception systems and utilize valuable information from cloud services –pertaining to traffic, weather, and more– without the same stringent safety requirements. While there remains a strong emphasis on reliability, this pattern enables us to avoid the complexities and uncertainties associated with components within the safety-critical path, paving the way for innovation and enhanced performance using MLCs while carefully considering the interplay between safety and comfort.

This work was funded by the Bavarian Ministry for Economic Affairs, Regional Development and Energy as part of a project to support the thematic development of the Institute for Cognitive Systems.

## II. The People Mover Use Case

To further illustrate the idea of the counter-player pattern, let's consider a simple use case involving a people mover system designed to serve as a VIP shuttle service. This shuttle can transport up to four passengers at a maximum speed of 15 kph and operates exclusively within a private campus, facilitating travel between a parking lot and a visitor center. To enhance safety, the system ensures all passengers are securely fastened in their seats for maximum deceleration.

One of the primary challenges faced by this system arises from conventional safety mechanisms that require the shuttle to stop frequently or significantly reduce its speed when pedestrians may cross its path. This conservative behavior is mainly due to the absence of designated walkways. When pedestrians approach the shuttle, its safety system often triggers an automatic stop or a substantial slowdown. Given the shuttle's role in transporting VIPs, the engineers decided that there was a pressing need to limit maximum deceleration, which resulted in earlier braking interventions.

To address these challenges, we could analyze known walking patterns in the environment, enabling the shuttle to anticipate where pedestrians are likely to cross. The shuttle could incorporate the ability to actively interact with pedestrians by assessing their gestures, facial expressions, and eye contact. This would allow the shuttle to react in a manner similar to human drivers navigating shared spaces, engaging in non-verbal communication and predicting pedestrian movements. However, implementing these measures would not be allowed in the safety-critical path, as they heavily rely on potentially unsafe machine learning and potentially unsafe data. This is where the counter-player architecture comes into play.

## III. The counter-player architecture

### A. Separation of safety and comfort

In the counter-player architecture, a clear design decision is made to separate safety from comfort, ensuring that the safety-player is exclusively focused on pure safety. This separation provides the necessary freedom for the counter-player to optimize the system's functional performance. The safety-player is programmed to intervene only at the last possible moment, focusing solely on bringing the vehicle to a safe state without considering comfort factors. While this might lead to more abrupt and harsh braking maneuvers, it streamlines the safety strategy by implementing a basic perimeter check with a reduced intervention radius that dynamically adjusts based on the shuttle's speed. To reduce the likelihood of these last-moment interventions, the safety-player provides continuous *alertness* values. These values offer real-time feedback on the proximity to an intervention threshold, allowing the counter-player to make more gradual adjustments.

The counter-player, on the other hand, can utilize all the sophisticated approaches mentioned above since it operates outside the safety-critical path. It continuously gathers data to adapt the shuttle's speed and trajectory to optimize operational efficiency. This means that travel time between the parking lot and the visitor center is minimized, while the likelihood of sudden braking by the safety-player is concurrently reduced. By decoupling advanced technologies in the counter-player from the safety-critical path, the system can leverage innovative technologies without compromising safety.

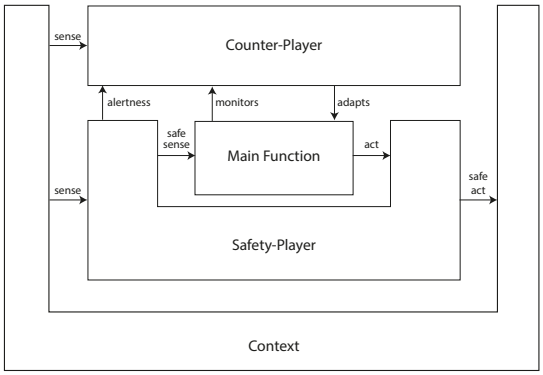


Fig. 1: Basic Architecture of the Counter-Player Pattern.

The starting point, as shown in Figure 1, does not significantly differ from other safety-related architectural patterns, such as safety cages [1] or a simplex architecture [2]. In a typical automotive system, the *main function* would be the main driving function, while the *safety-player* functions as a safety monitor or safety envelope, similar to the monitors used in Responsibility Sensitive Safety (RSS) [3]. Additionally, the safety-player implements a minimum-risk maneuver to bring the vehicle to a safe state. We assume that the safety-player dynamically adapts to the current context, such as the vehicle's operational context, as realized in approaches like adaptive RSS [4]. This adaptability enhances the system's permissiveness, but it is not the primary feature of the counter-player pattern.

Instead, the key novelty lies in adding a counter-player that continuously adapts the main function in order to maximize the function's functional performance while minimizing the probability of an intervention by the safety-player at the same time. While the pattern, therefore, may suggest game theory, where one player competes against the other, we opted for a more conservative approach, maintaining the idea of the counter-player competing against the safety-player.

### B. Self-adaptation

There are various approaches designed to develop or train the main function in a way that counteracts the safety component [5]. However, this can introduce implicit complexities to the main function that are difficult to verify and may result in unexpected interferences. In contrast, the counter-player pattern requires a distinct counter-player that is clearly separated from the main functionality. This decreases the chances of interference with the safety concept and makes it easier to assure the counter-player's reliability.

As illustrated in Figure 2, we realize the interaction between the counter-player and the main function as a self-adaptive system [6]. In this context, the main function serves as the

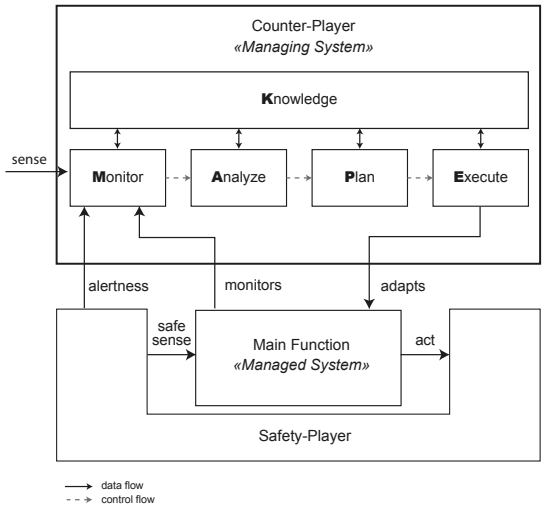


Fig. 2: Adaptive Counter-Player Architecture

managed system, responsible for implementing the actual functionality, while the counter-player acts as the managing system that facilitates the adaptation behavior. This means that the counter-player competes with the safety-player by continuously adapting the main function. To this end, the counter-player utilizes the established MAPE-K cycle [7], which starts with continuously *m*onitoring the context and the main function. Since the counter-player is a highly reliable component, but not part of the safety-critical path, we can utilize complex and sophisticated perception systems along with infrastructure data in this monitoring step. Then, the counter-player *a*nalyzes whether further optimization is necessary or if the probability of intervention from the safety-player becomes too high. If an adaptation is necessary, it further evaluates potential adaptation variants, selects the most promising one – aiming for maximal performance at an acceptable probability of a safety-player intervention – and then *p*lans and *e*xecutes the adaptation. All of these steps rely on *k*nowledge representation, typically using models that are available at runtime, enabling the system to reason about valid and purposeful adaptations.

The counter-player can only adjust the main function within a specific adaptation space. This space can be quite limiting if we define explicit operation modes, such as various perception or planning algorithms, as only one example. Alternatively, it can be more flexible if we additionally allow for adaptations to the algorithm's parameters, such as maximum speed, minimum distance to obstacles, etc. By this means, the adaptation becomes explicit, which is essential for mastering the resulting complexity.

Different approaches exist for creating a self-adaptive system that utilizes the MAPE-K cycle. However, a crucial aspect of the counter-player pattern is the analysis step, which identifies the best adaptation variant for a specific situation. This step optimizes the main function by adapting it to the optimal adaptation variant, maximizing its functional performance while simultaneously reducing the risk of intervention from the safety-player. Hence, the key to understanding the counter-player primarily lies in the analysis step.

### C. Counter-Player: Analysis

The analysis step relies on sophisticated monitoring of the system's context and serves as an optimization task that must be performed at run time. Given the current state and capabilities of the main function, as well as the vehicle's operational situation, the analyzer must identify an optimal configuration within the adaptation space. This configuration should maximize the system's performance while minimizing the likelihood of a safety-player intervention.

One approach would be the explicit modeling of different variants in a variational safety concept [8] and selecting the most appropriate variant at run time. While this method offers complete control and simplifies assurance, it remains somewhat restrictive and does not fully utilize the freedom that the pattern aims to provide for the counter-player.

In our use case, understanding and predicting the behavior of other traffic participants is essential for optimizing the shuttle's performance. To minimize unnecessary stops or slowdowns, we can analyze statistical data regarding typical pedestrian paths on campus. By enhancing this data with insights from individual pedestrians' previous trajectories, gestures, facial expressions, and other relevant factors, we can significantly improve our system's effectiveness. This approach allows us to derive accurate probabilities of pedestrians crossing the shuttle's path. However, integrating all this technology into the system's safety-critical framework would present currently unsolved challenges. However, as long as the manufacturer is confident in the reliability of this data and the required machine-learning components, we can incorporate them into our counter-player since it remains outside the safety-critical path.

Considering the complexity and high dimensionality of tasks involving human interactions, a typical conservative solution that analyzes the adaptation space based on explicit, manually defined Boolean rules would not be effective. On the other hand, learning a machine learning model that analyzes the adaptation space might reveal an unbeatably high peak performance. However, it would be very difficult to build confidence in its reliability. As an intermediate solution, analysis models of self-adaptive systems can be realized using Fuzzy Inference Systems [4] or Bayesian Networks or comparable stochastic models [9]. Defining such models manually, would however be very labor intensive and still too rigid. In order to combine the best of both worlds, we use reinforcement learning (RL) [10] to *learn* a fuzzy-based analysis model.

Using RL, we aim to develop an analysis model that offers the best adaptation policy $\pi$ in a given situation, optimizing a value function. In our case, the value is defined by a utility function $u \rightarrow [0, 1]$, which considers two factors: First, it includes the system's functional performance $p \rightarrow [0, 1]$. This $p$ could incorporate various metrics, such as the time needed between the parking lot and the visitor center, the jerk during driving, a lateral acceleration profile, or any other criteria we

choose to define good functional performance for our system. Second, it includes an alertness value $\alpha \in [0, 1]$, which reflects the proximity to a safety-player intervention. The safety-player would intervene when $\alpha \geq \epsilon_\alpha$. Our utility function can then be defined as $u = p \cdot (1 - \alpha)$. This means that, regardless of how high the functional performance $p$ may be, the utility $u$ will approach zero if the alertness $\alpha$ is high. Therefore, the safety-player does not merely implement a binary guard to indicate whether or not intervention is necessary; it also provides an additional output in the form of an alertness value $\alpha$.

We further assume that our system and its context are too complex to have an explicit model that specifies how an adaptation in a given state impacts utility. This is why we apply a model-free learning approach and hence need to apply Q-learning [10], where we learn a Q-function $Q(s, a)$. This Q-function takes the current state $s = s_S \cup s_C$ as its first input, which is a combination of the system's internal state $s_S$ and the state of its context $s_C$, i.e., the system's operational situation. The second input is an action $a$ that is valid in the current state $s$. In our case, the action $a$ refers to an adaptation action through which the counter-player can adapt the managed main functionality. As mentioned before, we realize this Q-function as a Fuzzy Inference System (FIS) [11], which we refer to as $Q_{Fuzzy}$. To train $Q_{Fuzzy}$, we can leverage simulations. We expect that the simulation-to-reality gap will be manageable, as we do not train at the perception level and can account for uncertainties in the perception chain, which are inherently modeled using Fuzzy Logic. We train $Q_{Fuzzy}$ by applying various actions $a$ in a given state $s$ across multiple simulation runs. Using our utility function $u$, we can calculate a reward for each step and determine the total value for the entire simulation run. To this end, we can employ established methods for learning fuzzy systems, as applied in similar settings in [12]. This process gradually enhances $Q_{Fuzzy}(s, a)$, ultimately leading to accurate predictions regarding the overall utility of selecting action $a$ in state $s$.

Once we have learned $Q_{Fuzzy}$, the analysis step of our adaptation manager at runtime can be streamlined. We take the current state $s$, as provided by the monitoring step, to determine an optimal adaptation strategy $\pi^*$ for that state, with $\pi^*(s) = \text{argmax}_a \, Q_{Fuzzy}(s, a)$.

## IV. Conclusion

There are still many unanswered questions, such as whether we can ensure safety while removing all safety margins from the safety-player, and whether the additional freedom granted to the counter-player justifies the added complexity. However, the potential of advanced machine learning (ML) approaches, like comprehensive scene analysis, to enhance safety and performance is enormous. Moreover, the rapid pace of innovation has led to the development of even more powerful techniques, such as foundation models, which we are currently unable to utilize due to safety concerns. The counter-player pattern could provide us with entirely new possibilities, allowing us to leverage new technologies outside of safety-critical paths. This would enable us to gather sufficient data and experience to incorporate these technologies into safety-critical areas in the future. For these reasons, we believe it is worthwhile to discuss this progressive approach that balances functional performance and safety.

## References


[1] G. Weiss, P. Schleiss, D. Schneider, and M. Trapp, "Towards integrating undependable self-adaptive systems in safety-critical environments," in *Proceedings of the 13th International Conference on Software Engineering for Adaptive and Self-Managing Systems*, 2018, pp. 26–32.

[2] L. Sha, R. Rajkumar, and M. Gagliardi, "The simplex architecture: An approach to build evolving industrial computing systems," in *Proceedings of the ISSAT Conference on Reliability*, 1994.

[3] S. Shalev-Shwartz, S. Shammah, and A. Shashua, "On a formal model of safe and scalable self-driving cars," *arXiv preprint arXiv:1708.06374*, 2017.

[4] A. Salvi, G. Weiss, M. Trapp, F. Oboril, and C. Buerkle, "Safety implications of runtime adaptation to changing operating conditions," in *2022 IEEE 25th International Conference on Intelligent Transportation Systems (ITSC)*. IEEE, 2022, pp. 2444–2449.

[5] D. T. Phan, R. Grosu, N. Jansen, N. Paoletti, S. A. Smolka, and S. D. Stoller, "Neural simplex architecture," in *NASA Formal Methods*, R. Lee, S. Jha, A. Mavridou, and D. Giannakopoulou, Eds. Cham: Springer International Publishing, 2020, pp. 97–114.

[6] D. Weyns, *An Introduction to Self-Adaptive Systems: A Contemporary Software Engineering Perspective*, ser. IEEE Press. Wiley, 2020.

[7] J. O. Kephart and D. M. Chess, "The vision of autonomic computing," *Computer*, vol. 36, no. 1, pp. 41–50, 2003.

[8] A. Kreutz, G. Weiss, and M. Trapp, "Automatic deduction of the impact of context variability on system safety goals," in *2024 19th European Dependable Computing Conference (EDCC)*, 2024, pp. 1–8.

[9] J. Reich and M. Trapp, "SINADRA: Towards a Framework for Assurable Situation-Aware Dynamic Risk Assessment of Autonomous Vehicles," in *16th European Dependable Computing Conference EDCC*, 2020.

[10] R. S. Sutton, "Reinforcement learning: An introduction," *A Bradford Book*, 2018.

[11] L. T. H. Lan, T. M. Tuan, T. T. Ngan, N. L. Giang, V. T. N. Ngoc, P. Van Hai *et al.*, "A new complex fuzzy inference system with fuzzy knowledge graph and extensions in decision making," *IEEE access : practical innovations, open solutions*, vol. 8, pp. 164 899–164 921, 2020.

[12] A. Salvi, G. Weiss, and M. Trapp, "Online identification of operational design domains of automated driving system features*," in *2024 IEEE Intelligent Vehicles Symposium (IV)*, 2024, pp. 1743–1749.