# Designing Human-in-the-Loop Autonomous Cyber-Physical Systems

Miriam Gil, Manoli Albert, Joan Fons, Vicente Pelechano

*Centro de Investigación en Métodos de Producción de Software,*
*Universitat Politècnica de València,*
*Camino de Vera s/n, 46022 Valencia, Spain*

**Abstract**

Even though full autonomy in Cyber-Physical Systems (*CPS*s) is a challenge that has been confronted in different application domains and industrial sectors, the current scenario still requires human intervention in these autonomous systems in order to accomplish tasks that are better performed with human-in-the-loop. Humans, machines, and software systems are required to interact and understand each other in order to work together in an effective and robust way. This human integration introduces an important number of challenges and problems to be solved in order to achieve seamless and solid participation. To manage this complexity, appropriate techniques and methods must be used to help *CPS* developers analyze and design this kind of human-in-the-loop integration. The goal of this paper is to identify the technological challenges and limitations of integrating humans into the *CPS*s autonomy loop and to break new ground for design solutions in order to develop what we call *HiL-$\mathcal{A}$CPS* systems. This work defines a conceptual framework to characterize the cooperation between humans and autonomous *CPS*s and provides techniques for applying the framework in order to design proper human integration. The emergent autonomous car domain is considered as a running example. It covers some of the current limitations of involving drivers into the autonomous functionalities. Finally, to validate the proposal, an autonomous car prototype was built applying the conceptual framework. This prototype was evaluated to check whether the human integration implemented behaves as defined in its specification.

*Keywords:* Cyber-Physical Systems, Autonomous Systems, Human-in-the-Loop, Human-Computer Interaction, Autonomous Cars

## 1. Introduction

The future "smart world" is being designed as a complex eco-system that is made up of a wide variety of physical devices and distributed services that interact each other and that are controlled by a great number of computing elements. The physical devices have been empowered with computing and communications capabilities to provide them with some sort of intelligence that allows them to be connected to the digital world [1]. This has opened up a wide variety of possibilities in different application domains such as Smart Cities, Factories of the Future and Autonomous Vehicles. Systems of this kind are broadly referred to as Cyber-Physical Systems (*CPS*) in which both the physical and the digital worlds are intertwined to provide smart physical services.

These systems are required to be capable of autonomously adapting themselves at run time to new environment conditions, unpredictable situations, changing user needs and intentions, new types of devices, new technologies to interact with and new services to consume. A lot of work has been done in the Autonomic Computing field of research to design systems with self-managing capabilities. These systems apply control theory principles to make use of control loops [2, 3]. In this context, *CPS*s can be considered as autonomous systems that are engineered and designed to be capable of self-managing the relation between the physical and the digital worlds in order to provide smart services that are adapted to the current context conditions. Systems of this kind must continuously monitor

*Email addresses:* `mgil@pros.upv.es` (Miriam Gil), `malbert@pros.upv.es` (Manoli Albert), `jjfons@pros.upv.es` (Joan Fons), `pele@pros.upv.es` (Vicente Pelechano)

themselves and their surrounding environment in order to identify events or patterns that require taking action, decide how to react, design an action plan, and proactively execute that plan. In this work, we use the term *𝒜CPS* to emphasize the autonomous nature of the *CPS*.

Future software systems are going to gradually and widely introduce autonomous capabilities in everything [4]. Different application domains and industrial sectors (such as autonomous cars, robots, or drones) face the challenge of achieving full autonomy. Nevertheless, the diversity of systems, domains, environments, context situations, and social and legal constraints, all point to a world where this full autonomy is a utopia within the short or medium term [5, 6]. This scenario leads software engineers to face the challenge of developing *𝒜CPS*s that support the concept of autonomous levels [4], with the realization that an autonomous system cannot be fully automated and that human involvement is required to execute some tasks.

Therefore, humans are required for intimate collaboration with *𝒜CPS*s in the execution of certain tasks that cannot be performed autonomously ("human-in-the-loop"). Humans, machines and software must interact and understand each other in order to work together in an effective and robust way. Even in full autonomous scenarios (fully autonomous cars, robots at factories of the future, etc.), *𝒜CPS*s are required to explain their behavior to humans, provide them with feedback, and to allow them to override system actions. A trust-based relationship must be built between humans and machines so they can work together and achieve effective human-machine integration. A new vision of this relationship is required, because complete integration is necessary [7]. This *Human-in-the-Loop & Autonomous CPS* (*HiL-𝒜CPS*) interaction differs from current HCI models, which fail in providing proper constructs and models to design and validate such interactions. Unfortunately, while many *CPS*s are human-centric applications where humans are an essential part of the system, most of these systems still consider humans to be an external and unpredictable element in the control loop [1]. Human integration into the *𝒜CPS* must be natural, robust, and non-intrusive. Therefore, in addition to the functional and autonomous facets of the *𝒜CPS*, human factors, such as training level, stress, or fatigue must also be taken into account. These factors influences the likelihood of succeeding at carrying out specific tasks [8].

To manage this complexity, system designers should engineer their systems to support human integration and maximize their chances of performing their tasks successfully [9]. From an engineering point of view, the involvement of humans in *𝒜CPS*s introduces an important number of challenges and limitations to be solved in order to achieve seamless and robust participation, even in those situations where the humans have their attentional, cognitive, and physical resources limited to perform the interaction [10]. Some works have identified the problem of how to involve humans within the adaptation loop of an autonomous system, but there is no engineering proposal that helps software designers to design this human participation [8, 1].

This work aims at identifying the technological challenges for *HiL-𝒜CPS*s and breaking new ground for design solutions in the Software Engineering and Human-Computer Interaction (HCI) fields. It analyzes the foundations for performing Human-Autonomous System integration, and it presents a conceptual framework to characterize the cooperation and integration between humans and *𝒜CPS*s. The goal of this framework is to provide a design theory to analyze and design the involvement of humans in *𝒜CPS*s in order to improve system performance and understandability and to avoid developing intrusive and annoying systems. We provide a method and a notation to apply the framework and describe the Human-*𝒜CPS* integration. Also, we propose an execution model that provides the operational support for guiding the implementation of the Human-*𝒜CPS* integration described with the conceptual framework. Finally, this paper instantiates the proposal to design and develop a functional prototype of an autonomous car with *HiL-𝒜CPS*s capabilities. The prototype was validated using a set of tests that checked whether the car prototype behaves as defined in its specification. Moreover, the prototype can be viewed as a tool for helping to improve Human-*𝒜CPS* integration designs since it serves to test and refine them until a suitable interaction design is achieved.

The rest of the paper is structured as follows. Section 2 shows the related work. Section 3 identifies the challenges to involve humans in *𝒜CPS*s and defines three design principles to follow based on these challenges. In Section 4, we introduce the case study of the autonomous car, which is used throughout the paper. Section 5 analyzes how to open the *𝒜CPS* control loop to human participation. Section 6 defines the conceptual framework for designing human collaboration within *HiL-𝒜CPS*. Section 7 introduces an execution model that defines how this human collaboration should be operationalized. Section 8 validates the proposal by means of a functional prototype of the autonomous car described in the case study. Finally, Section 9 presents conclusions and future work.

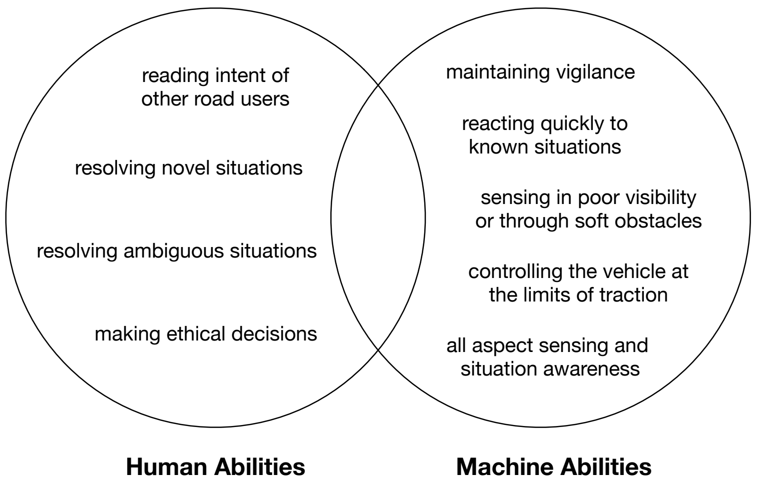

reading intent of other road users
resolving novel situations
resolving ambiguous situations
making ethical decisions
maintaining vigilance
reacting quickly to known situations
sensing in poor visibility or through soft obstacles
controlling the vehicle at the limits of traction
all aspect sensing and situation awareness
Human Abilities
Machine Abilities

Figure 1: Partially overlapping areas of human and machine abilities in the area of road vehicle systems [12]

## 2. Related work

Human Integration within *CPS*s is a matter that concerns several disciplines and applies to many domains. Thus, the related work covers a wide-spread of works. In order to better organize this related work, first we focus on previous works that analyze the role of humans in Autonomous Systems, which allows to contextualize our proposal. Then, we introduce some works that deal with how to integrate humans into *CPS*s, identifying problems and challenges. Finally, we review some works that address human integration from the point of view of the Human-Computer Interaction.

### *2.1. The role of humans in Autonomous Systems*

The system autonomy represents the capability of the system to make decisions, act or understand its context "by itself" without human participation [11]. According to [4], we are gradually moving into a world of autonomous systems where systems will have different autonomy levels. Full autonomy is the capacity of systems to act in whatever situation and for all their capabilities in an autonomous way. However, this scenario will only be achieved in the very long term, as stated in [4]. Meanwhile, autonomous systems will require human support to guarantee their complete and correct behavior in all situations. Machines can react faster than humans when they are able to properly evaluate the situation and have a programmed response. Humans have the capability to assess and respond to novel situations that may be beyond the capabilities of a system. Therefore, a dynamic sharing of control will provide the scope for the greatest total safety [12].

In the middle of the last century, [13] stated which processes, tasks, and functions can be more efficiently performed by a human or by a machine. This work identified functions that can be better performed by humans (detection, perception, judgement, induction, improvisation or long-term memory, etc.) versus others that can be better solved by machines (computing power, replication capacity, operation simultaneity, short-term memory, etc.). According to this work, the functions that are better performed by machines should be automated, while the other functions should be allocated to the human operator. Despite the contemporary technological limitations and the paradigm shift in computation, this work continues to be a reference [14, 15]. For example, in the vehicle automation domain, [12] identified the superior abilities of humans over machines and vice versa (see Fig. 1). In the autonomous car domain, [12] gives an example of a cyclist that has fallen at speed fast moving object difficult to characterize by the system and the system would likely to find ambiguous and difficult to resolve, but where the human would have little trouble in assessing the situation. Other examples in this domain in which the car cannot perform the autonomous driving without a human is in the case of driving in a city, driving with adverse weather, or in case that any sensor used in the autonomous driving fails [16].

The roadmap to achieving fully autonomous systems involves a progressive shift in the control and supervision by humans and machines [17]. The shift in the responsibility for planning, executing, and actively monitoring to a

Table 1: Sheridan & Verplank's (1978) Levels of Automation [19]

| Level | Description |
|---|---|
| 1 | Human does the whole job up to the point of turning it over to the computer to implement |
| 2 | Computer helps by determining options |
| 3 | Computer helps to determine options and suggests one, which human need not follow |
| 4 | Computer selects action and human may or may not do it |
| 5 | Computer selects action and implements it if human approves |
| 6 | Computer selects action, informs human in plenty of time to stop it |
| 7 | Computer does whole job and necessarily tells human what it did |
| 8 | Computer does whole job and tells human what it did only if human explicitly asks |
| 9 | Computer does whole job and decides what the human should be told |
| 10 | Computer does whole job if it decides it should be done, and tells the human (if it decides) |

supervisory role allows for reduction in the cognitive responsibilities of the human operator and the stress caused by these tasks, and it also provides the possibility of enhancing time response, security levels, and support for multiple tasks in parallel [18].

Several attempts have been made to create taxonomies to identify collaborative scenarios where humans and systems could cooperate to complement tasks, processes, and functions. Both general scope proposals and specific domain proposals (space, air or land unmanned vehicles, machines and robots in factories, etc.) have been defined. Each approach is made up of different numbers of automation levels (approaches that provide four autonomy levels to others that provide 10 or even more). Each level $\mathcal{L}$ is associated to different actions that should be allocated to humans to assist the system. The most widely used taxonomy was introduced by Sheridan and Verplank in [19] (see Table 1). This taxonomy defines 10 levels of automation that refer to the locus of control (human or system) and it states how the information is presented to the human. In low automation levels (below $\mathcal{L}5$), control is allocated to the human and the system provides little assistance to him/her. Between $\mathcal{L}5$ and $\mathcal{L}7$, the control is collaborative. In higher levels (above $\mathcal{L}7$) control is allocated to the system and the human only performs supervision tasks and a few actions.

Simultaneously, in the fields of autonomous spacecraft and vehicles, several specific taxonomies have been proposed based on identifying the control over the primary vehicle functions by the driver versus the unmanned vehicle (UMV) [20, 16, 21]. In these taxonomies, as the level of automation increases, the driver's role shifts from primary control, passing through supervised control to full autonomy.

### 2.2. *Efforts to integrate humans in Autonomous Systems*

Several approaches for integrating humans within *CPS*s are surveyed in Nunes et al. [1]. In that survey, the authors carry out an analysis on existing solutions, reviewing technologies for supporting humans in *CPS*s, applications in this domain, and the social networking impact on *CPS*s. They present a classification of the general roles of humans in future *CPS*s. Several authors have also highlighted the need to distinguish the different types of user participation on the system (roles) [22], [23], [24]. The role taxonomy presented in [1] is based on the previous work of Munir et al. in [25], where a taxonomic foundation for human-in-the-loop in *CPS*s applications is introduced. In that work, Munir et al. highlighted the incorporation of human behavior into the methodology of feedback control as a crucial challenge to be addressed in the research of human-in-the-loop in *CPS*s.

In [8], Cámara et al. provided an extension of the Rainbow framework to reason about the user's integration in self-adaptation. However, they only focused on the user role as executor of adaptations. Moreover, the authors do not provide design guidelines or tools to design this integration. Shin et al. in [26] presented a proposal of resolution of conflicts by combining the automatic resolution with the resolution directed by the user according to her/his preferences. This solution only considers two types of participation: automatic and manual.

The problems that can arise from considering the human as an active participant in a shared or switched control loop have also been studied in the literature. Cranor in [9] proposes a framework for reasoning about human-in-the-loop in secure systems. This framework can be used by system designers to identify problem areas before a system is built and to proactively address deficiencies. However, it is domain dependent (only applicable on secure systems). Nothwang et al. in [27] investigate the contributors to success/failure in current human-autonomy integration frameworks and propose guidelines for the safe and resilient use of humans and autonomy with regard to performance, consequence, and the stability of human-machine switching. This is still an unresolved issue in the work on the human-machine collaboration. Recently, Dorn et al. in [28] introduced a framework to reason about the effects of changes at the software level on the interactions of users and vice-versa. This approach focuses on a collaborative topology that is capable of offering a collaborative adaptation process to allow more sophisticated adaptations. However, it does not explicitly introduce the user into the adaptation process. In [29], Evers et al. provided a solution to take users into account in self-adaptive interfaces based on their interaction practices. However, the solution that was presented only deals with isolated user participation types and does not consider the full range of types of participation that the user may have.

In the field of autonomous cars, in the last years many works have emerged to deal with the problem of how to involve humans in the car system. In [30], Mirnig et al. overcome the problem of the control transition in autonomous cars, which is still an unresolved problem. That work focuses on interaction solutions for control transitions, which provides a more abstract perspective. Different strategies for keeping drivers aware of the car's status are analyzed in Noah et al. [31]. Solving this question directly benefits the hand-off issue. The suitability of the feedback provided to drivers is analyzed by Kunze et al. in [32]. The authors throughly explore the impact of the kind of feedback on user experience.

### *2.3. Human integration in the Human-Computer Interaction area*

In the area of HCI, advances increase the strength of *𝒜CPS*s with Human Integration. In [7], Farooq et al. discuss the end of the era of human-computer interaction leading to the era of human-computer integration. To achieve this vision, advanced technologies for human-machine communication are required. Schirner et al. [33] see the future of HiLCPSs putting aside traditional interfaces (such as the keyboard, mouse, or joystick) to shift to transparent interfaces that use existing electrophysiological signals such as electroencephalography (EEG), electrocardiography (ECG), and electromyography (EMG). Lloyd et al. [34] introduced the concept of a co-adaptive human-computer interaction system where the system benefits from the human by recognizing his/her mental states. In [35], Huang et al. presented the preliminary results of creating a brain mouse to command actions of a software system based on human intention. The goal was to incorporate mental state of humans in systems so that they can "feel" and "anticipate" user intentions and put human-in-the-loop. In addition to these cited works, research on the quality properties of human-computer interactions constitutes an important source for our proposal. Works of Lee et al. [36] and Stanton et al. [37] encourage the development of systems that keep the human informed, provide him/her with feedback and reduce cognitive responsibilities and stress. Related to these works, other ones that also provide insights on this issue are the work of Nahavandi in [38], which explores trust as a main attribute of systems to be accepted in our daily lives, and the work of Lim et al. in [39], which discusses intelligibility in context-aware applications.

The analysis of this related work confirms that collaborative integration of humans in autonomous systems will continue to be a challenge in the long term. Next section identifies the main challenges in design of *HiL-𝒜CPS*s.

## 3. Challenges in *HiL-𝒜CPS*s

Based on the need for human participation in *𝒜CPS*s and assuming that humans are required to help or to assist in the functionality of these kinds of systems, in this section we identify three main design principles that impact the design of efficient human integration.

The first challenge that must be confronted when involving humans in *𝒜CPS*s is to complement the functionality of *𝒜CPS*s by means of the collaborative work between human and system. This collaborative work carries out the tasks that *𝒜CPS*s cannot execute on their own or that humans would rather participate in. The *𝒜CPS* together with the human (*HiL-𝒜CPS*) must perform the "whole" functionality for which it was developed. This collaborative work is expected to be performed under the functional requirements of *correctness* (the degree to which the system provides the correct results with the needed level of precision) and *completeness* (the degree to which the set of tasks covers all the specified objectives).

In addition, we seek to develop *𝒜CPS*s that perform this human involvement in a natural, robust, and non-intrusive way. Quality factors of this kind must also be taken into account. There are works [40, 41, 42, 43] that have identified

Table 2: Quality properties of Human-System Integration

| Property | Description |
|---|---|
| *Intelligibility* | The system's ability to communicate to the end user how the interaction is conducted in a meaningful and representative way. |
| *Transparency* | The extent to which the user can understand system actions and/or s/he has a clear picture of how the system works. |
| *Observability* | The system's ability to make perceivable its state in a relevant way for the end user. |
| *Controllability* | The extent to which a user can control the system. It involves that the user brings about or prevents specific actions or states of the system if s/he has the goal of doing so. |
| *Obtrusiveness* | The extent to which the system places demand on the user's attention which reduces the user's ability to concentrate on her/his primary tasks. |
| *Perceptibility* | The system's ability to perceive domains or adaptation actions achieved by the end user. |
| *Feedback* | Any entity's ability to provide other entity with any information in return to an action executed by the other entity. |
| *Feedforward* | Any entity's ability to provide other entity with any information before this last entity executes any action. |

quality properties that are relevant to design user interactions for intelligent and autonomous systems. Table 2 depicts a description of these relevant properties.

By analyzing the above requirements, we state that the human involvement in *HiL-𝒜CPS*s introduces an important number of challenges to be solved in order to achieve seamless and solid participation, especially when considering situations where human attentional, cognitive, and physical resources are limited. These challenges lead to three *design principles*:

1. **Complement *𝒜CPS* functionality**. Humans are integrated in an intimate collaboration with the system to assist in the execution of certain tasks in a correct and complete manner. In order to involve humans, *𝒜CPS*s must manage the control flow from the system to the user, and vice-versa [30]. This shared-control relationship defines a space for human-system integration, in which the control of the task is transferred dynamically between the human and the system. Controllability is a key issue since we must manage to what extent a user can get in control of the system. Also, perceptibility is a required system property since the system must perceive the user's actions and reactions. For example, in the case of the autonomous car, when the car cannot drive in a city, the human needs to complement this functionality.
2. **Achieve understandability**. A human must understand how the *𝒜CPS* operates and what is happening at the current moment or even the moment before, otherwise, the human may mistrust the system. It is necessary to make use of mechanisms that provide relevant feedback and feedforward to the human. Also, feedback and feedforward actions allow systems to improve observability and intelligibility and make systems more transparent to the users. For example, the car should give the adequate feedback to the human in order that the human can understand the car's behavior and participate in a correct way when required.
3. **Manage user attention**. Getting or maintaining a human's attention to *𝒜CPS*s is not a trivial task. The human can be distracted or focused on another tasks at the moment the *𝒜CPS* could require her/his participation. The human can even lose attention to the task while collaborating with the *𝒜CPS*. The *𝒜CPS* must perform effective actions in order to get the human involved and to help him/her to maintain a suitable level of attention. However, at the same time, the system must avoid disturbing or overwhelming the user with unnecessary actions that can require too much attention or cause undesired results or bad user experiences. Therefore, the *𝒜CPS* must perform actions to get human attention in accordance with the level of human involvement require, using the correct level of obtrusiveness for this task. For example, the car should get human attention when human has to complement its functionality.

In summary, we address human-system integration to complement *𝒜CPS* functionality with shared control be-

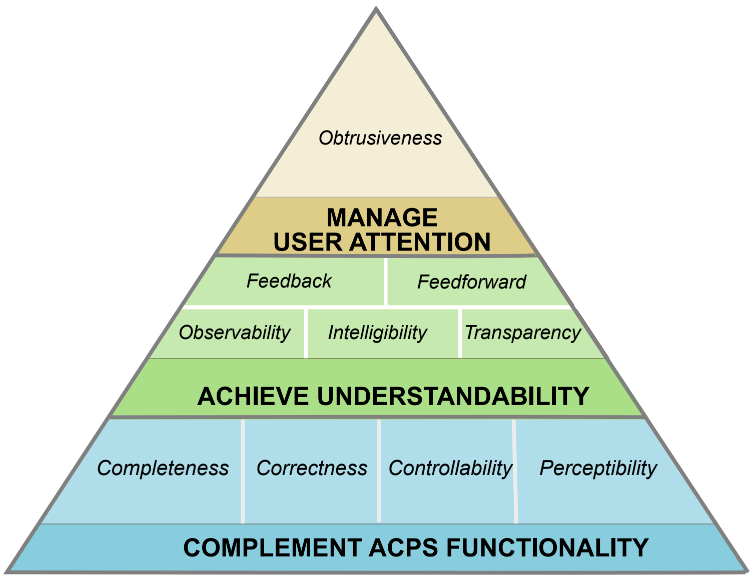


Figure 2: Design principles of human involvement

tween humans and systems, thus guaranteeing *functional suitability* (correctness and completeness) as well as human-automation interaction properties that involve human reliability and system obtrusiveness. Proper human involvement in an *𝒜CPS* is constructed by layering the three principles as shown in the pyramid in Fig. 2. The pyramid expresses the *HiL-𝒜CPS* design principles arranged in a hierarchy. At the bottom of the pyramid is the basic principle *Complement 𝒜CPS functionality*. This is the most important principle since, in order to integrate humans into systems, *𝒜CPS*s must manage the control flow between the human and the system. In the middle of the pyramid is the *Achieve understandability* principle, since in order for humans to be able to perform their actions, they must understand how to participate. During this human-*𝒜CPS* collaboration, it is desirable to avoid disturbing the user in order to achieve a good user experience. Therefore, the *Manage user attention* principle is placed at the top of the pyramid. The conceptual framework proposed in this paper aims at providing a basis for designing techniques and methods that allow designers to achieve these three principles.

## 4. Case study: the autonomous car

The best way to intuitively understand the proposal is through a real case study. One of the most well-known and popular *HiL-𝒜CPS*s today is the case of autonomous cars. The autonomous car technology is ready and a many car units are already driving autonomously on the roads (with minimal driver intervention) under controlled situations (e.g. “motorways in clear weather”). This level of autonomy is expected to increase gradually in the next few years [16, 21], but, in the meantime, the driver must stay attentive to the steering wheel and to the road in what is being called *delegated driving*. Autonomous driving is expected to reach maturity by 2030 [21], when cars reach *ℒ5* and abandon delegated driving to move on their own without a driver. Until then, all autonomous cars are required to drive with an assistance driver (*Human-in-the-Loop*).

One representative example of these autonomous cars is the Google car, which is ranked at *ℒ3* [44] and is one of the first industrial autonomous cars. This car is capable of traveling thousands of kilometers without human intervention under restricted conditions: specially marked roads and good weather. The driver is required to be prepared to intervene at any moment in case of conflictive situations [45]. Although the car is designed to have full autonomous driving capabilities under certain circumstances, it still has some limitations that require human supervision and control in complex driving situations or emergencies [46].

## *4.1. Autonomous car limitations*

In the last few years, the news about some fatal accidents involving autonomous cars has been widely spread, intensifying the debate on safety to levels that threaten to harm the adoption of the technology. In 2016, a Tesla car crashed into a truck while driving in $\mathcal{L}3$ AutoPilot mode. In 2018, an Uber/Google Car vehicle struck a pedestrian that was crossing the road, also while driving in $\mathcal{L}3$ AutoPilot mode. NTSB reports of these accidents [47, 48] stated that, among other technical questions, the cars failed in providing good integration of the human with the driving tasks. Some of the known limitations are:

- **"Hand-off" problem:** This describes a situation in which the car can no longer perform the automated driving task and requires the driver to take control of the car in a short period of time. It must deal with situations in which the driver may be distracted by texting, reading email, talking, etc. [49].
- **Inadequate (or lack) of feedback:** This refers to scenarios in which the car does not properly notify the human of what is happening. It can cause the driver to misunderstand why the car is performing certain actions and potentially become distrustful of the system's reactions. Consequently, the driver's situation awareness could be inadequate, resulting in control problems in critical situations [50]. Feedback should be transparent enough so that the driver not only knows what the system is doing but is also prepared to (quickly) intervene and override the car's behavior (if required).
- **Issues of mental workload:** With autonomous cars, the driver changes the task from only monitoring the situation to also monitoring the automation functions. It has also been suggested that a proliferation of driver support systems could overload the driver [50] (fatigue, stress, attention, etc.). For this reason, all driver interaction must be in a clear and timely fashion in order not to overload users.

This is not a full list of the known limitations. These are the limitations that are closely related to human integration within autonomic functions. There are other limitations that are more related to functional or technical aspects. The following section introduces the strategy that we present to mitigate these problems.

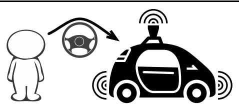
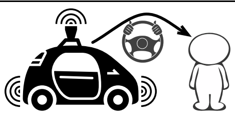
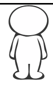
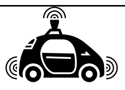
## *4.2. Facing autonomous car limitations*

The problems described above are aligned with the design principles introduced in Section 3. The hand-off problem is about how to involve the human in the driving task so that s/he is ready to take over the car at any moment if required. This problem can be mitigated with the *Complement $\mathcal{A}$CPS functionality* design principle, which deals with the shared-control issue between the user and the system. However, this problem also has the challenge of getting the human's attention in any human situation (s/he may be distracted). This challenge can be faced by applying the *manage user attention* principle. To perform a safe and effective control transition, the human must also understand the car's behavior, which can be overcome with the *achieve understandability* design principle. The inadequate (or lack of) feedback problem can be faced with the *achieve understandability* design principle, which encourages designers to enhance system understandability by providing feedback and feedforward with explanations about what the system is doing or what is happening. Finally, the problem of mental workload could be mitigated if designers apply the *manage user attention* design principle to achieve getting human attention but avoiding disturbing and overwhelming him/her.

The case of the autonomous car at $\mathcal{L}3$ is used as a case study in this work. In order to illustrate our proposal seven tasks that require human integration have been selected among the tasks that are usually performed in an autonomous car. These tasks are the following:

- *$\mathcal{T}1$ Supervised Autonomous Driving*: represents the collaborative work that the system and the human do in AutoPilot mode. The AutoPilot is responsible for performing the automated driving task (lane keeping with adaptive cruise control). However, the $\mathcal{L}3$ AutoPilot mode requires a driver to always be in attentive mode and ready to take over (if required). This scenario is used to illustrate the Tesla (2016) and Uber (2018) accidents. Most of the reports related to these two accidents agree that the drivers were not attentive to the driving task at the time of the accidents. Even though the cause of the accidents can be blamed on the drivers' misuse of the AutoPilot mode, we consider that a good design of human participation in this task would have reduced the likelihood of these accidents.

Table 3: Control Transfer Tasks in the Autonomous Car

| **Initiated by** |  **Handover** |  **Takeover** |
|---|---|---|
|  | *$\mathcal{T}$3 Handover* | *$\mathcal{T}$4 Driver Reassumming Control* |
|  | *$\mathcal{T}$5 Emergency Handover* | *$\mathcal{T}$6 Takeover*<br>*$\mathcal{T}$7 Emergency Takeover* |

- *$\mathcal{T}$2 Supervised Manual Driving*: represents the collaborative work that the system and the human do in Manual mode. The human is responsible for performing the driving task. However, the $\mathcal{L}3$ Manual mode requires the system supervises the state of the human (and takes the appropriate actions when necessary).

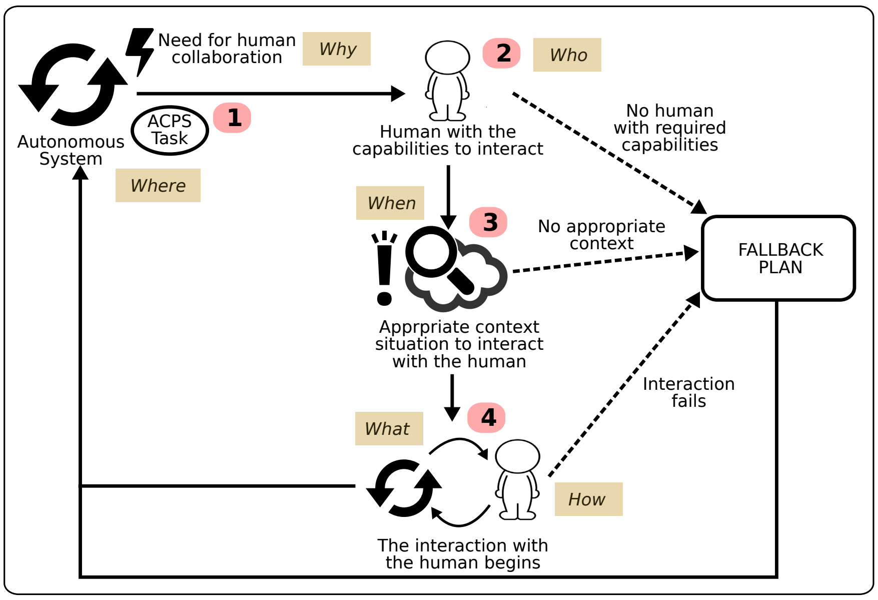
These two tasks are the basic ones that are performed in an autonomous car in order to accomplish the main goal of the car, which is safety traveling between destinations. Each task represents a different driving mode. In order to perform the control transition between autonomous and manual mode, additional tasks are required. Based on [30], the transition of control from the autonomous system to driver is called *takeover*, while the transition from driver to the system is called *handover*. Both takeover and handover can be initiated by the system or by the human. This casuistic of transitions leads to the quadrants shown in Table 3, where the different tasks to perform the transition of control are placed. These tasks are the following:

- *$\mathcal{T}$3 Handover*: the driver gives the order to the system for it takes over control of the car.
- *$\mathcal{T}$4 Driver Reassumming Control*: the human requests control of the car.
- *$\mathcal{T}$5 Emergency Handover*: the system realizes that the human cannot continue driving (e.g., it realizes the driver has fallen asleep) and takes control of the car in a rush.
- *$\mathcal{T}$6 Takeover*: the system transfers control of the car to the human in a situation that does not require hurry (e.g. the car is approaching a city and the car cannot continue autonomously driving).
- *$\mathcal{T}$7 Emergency Takeover*: the system must leave the car in a safety situation while transfers control of the car to the human. This task takes place in emergency situations (e.g. a sensor fails and the system cannot continue autonomously driving).

Note that based on [16] we distinguish two types of takeover initiated by the system: one corresponds to a situation that does not require a rushed transition and the other represents an emergency situation. This distinction is important as it impacts on the design of the transition.

## 5. Opening the loop to human integration

This section analyzes how humans can be integrated within $\mathcal{A}CPS$s taking into account the design principles identified in Section 3. This analysis aims at recognizing the main aspects that have to be considered in the design of human integration and extending the behavior of $\mathcal{A}CPS$s (based on a closed feedback loop) in order to integrate humans into some of their tasks, which is called *"opening the loop"* .



Need for human collaboration
Why
2
Who
Autonomous System
ACPS Task
1
Human with the capabilities to interact
No human with required capabilities
Where
When
3
No appropriate context
FALLBACK PLAN
Apprpriate context situation to interact with the human
Interaction fails
What
4
How
The interaction with the human begins

Figure 3: Extended *𝒜CPS* behavior

### 5.1. *Main aspects of the HiL-𝒜CPS design*

We applied the *"Six Serving Men Questions"* analysis method [22] to study the human integration problem. The questions helped us to identify the aspects to be taken into account when designing an *HiL-𝒜CPS*. The analysis through the questions is presented below and illustrated in Fig. 3.

- ***Why** is human integration necessary within the 𝒜CPS?* A human should be integrated into an *𝒜CPS* in order to complement its functionality for two possible reasons: (1) the *𝒜CPS* by itself cannot solve a specific task completely or correctly; or (2) the human requests the control of the system because of her/his wish to collaborate in a specific task. For example, in the case of an autonomous car at *ℒ3*, the car requires the human's collaboration when it approaches a city, since it cannot continue driving autonomously. Another reason for integrating a human into this car would be the human's request for control of the vehicle during autonomous driving by turning the steering wheel when s/he faces a dangerous situation.

- ***Where** is the human integrated within the 𝒜CPS?* This question is related to which tasks could require human integration because the system cannot autonomously perform them or which tasks allow the control of the system to be transferred when the human requests it. Human integration into these tasks *complements 𝒜CPS functionality*. Examples of these tasks for the case of the autonomous car are enumerated in Section 4.2. For example, in the case of *𝒯1 Supervised Autonomous Driving*, it requires human integration for safety reasons.

- ***Who** is the human that can be integrated in the task?* The task needs a human with a specific kind of capability, knowledge, background, etc. to correctly *complement the 𝒜CPS functionality*. For example, *𝒯4 Driver Reassumming Control* requires a human with *driving* capabilities to achieve the control transition.

- ***When** is it feasible to do the collaboration?* This question is related to which context situations maximize an optimal task execution or which situations allow the task to be performed in a safe manner. An optimal context situation allows to correctly *complement the 𝒜CPS functionality* since if the task were executed in an inappropriate situation, the success of the task could be threatened. For example, if the human is not seated in the driver's seat when the car requires her/his collaboration to take control of the car, the task cannot be performed since the human is not ready to drive. In those situations when the system, its environment, or the human are not prepared at the moment of the collaboration, it is necessary to provide the human with relevant feedback to inform him/her of the situation (understandability) and to execute actions to *acquire the correct human attention* in order to engage him/her in the task.

- ***What** do the human and the system have to do collaboratively?* In other words, which work have to do the human and the system in order to achieve the task's goal? For example, *𝒯6 Takeover* involves that *the system notifies the human that s/he is required to take over control of the car*, *the human confirms the takeover* (that is, the human makes the system aware of s/he accepts the takeover) and *the system transfers control to the human*. But also, some feedback or feedforward information has to be provided to the human to *achieve understandability*. For example, in the previous task, after the system transfers control of the vehicle to the human, *the system should inform him/her about the achieved driving mode*. Moreover, it is important to introduce mechanisms to *manage the user's attention* in order not to overwhelm him/her when it is not necessary. For example, if the human is already paying attention to the driving task at the moment that s/he is requested to take over control, the system should not warn him/her in a noteworthy way.

- ***How** is the human-HiL-𝒜CPS collaboration performed?* Tasks usually operate under specific conditions or limitations that must be fulfilled throughout the execution of the task. These limitations help the *HiL-𝒜CPS* to correctly *complement the 𝒜CPS functionality*. For example, the control of the car must be taken over by before 10 seconds after the system notifies the human. Otherwise, the task will be considered as failed.

### *5.2. Extending 𝒜CPSs to become HiL-𝒜CPSs*

Traditionally, canonical models of human-computer interaction are based on a feedback loop in which information flows from a system (e.g., a computer or a car) through a person and back through the system again [51]. The person acts to achieve a goal in an environment (provides input to the system); the person measures the effect of her/his action on the environment (interprets output from the system, feedback) and then compares the result with the goal. This is a simple first-order cybernetic system. Taking this traditional feedback loop into consideration and based on the aspects identified to integrate the human into the *𝒜CPS*, we propose a redesign of the *𝒜CPS* behavior in order to include the steps required for performing human collaboration. These steps are depicted in Fig. 3:

**Step 1: Monitoring**. The *𝒜CPS* is continuously monitoring its environment to identify situations in which human integration is required. In this case, it raises an event that triggers the need for a human to provide help or to carry out a task.

**Step 2: Requesting human**. The system requests a human with specific capabilities to perform the task.

**Step 3: Checking context**. The system verifies that the conditions for integrating the human into the task are accomplished.

**Step 4: Human collaboration**. The system and the human initiate their collaborative work in a shared-control manner.

Some of these steps could hinder the correct execution of the task. For example a human with the required profile may not be available; or the human may not be prepared to collaborate in the task; or the human interaction may not be as expected. In those cases, the task should leave the system in a consistent and safe state. After the human collaboration, the system returns to the autonomous mode until a new collaboration is required.

This section has identified the aspects to be taken into account when designing *HiL-𝒜CPS*s. The outcomes of this section are used in Section 6 to define a conceptual framework for the design of human integration.

## 6. Conceptual framework to design human collaboration within *HiL-𝒜CPS*s

This section presents a conceptual framework to help in designing Human-in-the-loop integration within an *𝒜CPS*. This framework states the concerns that an interaction designer must deal with in order to design proper human integration within *𝒜CPS*, and it provides a design theory that explains the foundations for *HiL-𝒜CPS* design. The conceptual framework is constituted by a set of concepts that define the collaborative work between the human and

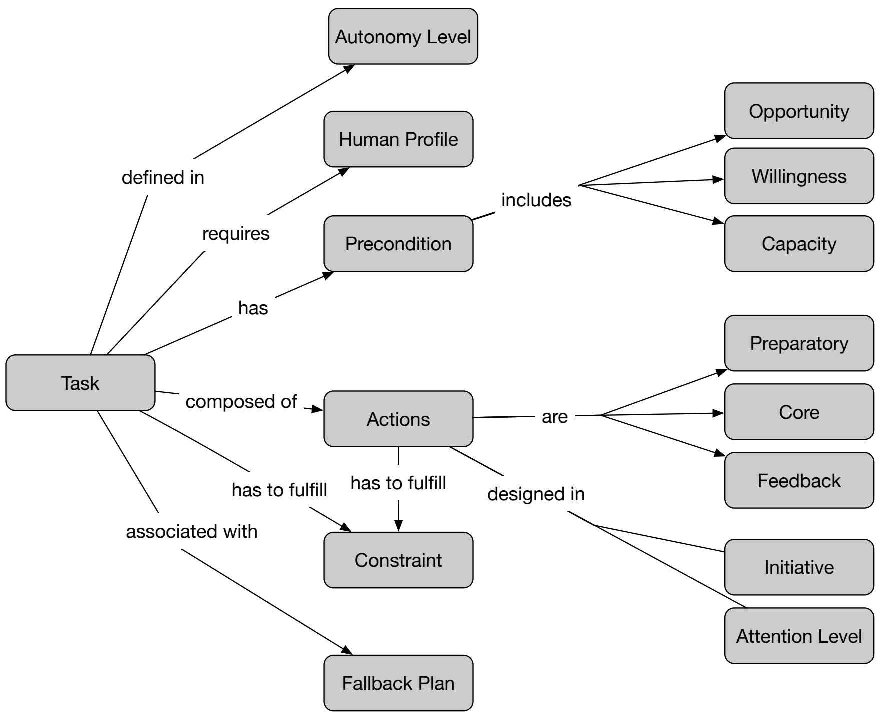


Figure 4: Concepts of the *HiL-𝒜CPS* framework

the *𝒜CPS*, which are introduced in Section 6.1 and Section 6.2. Section 6.3 explains how to apply this conceptual framework.

To build the framework, we identified and defined the key elements, terms, concepts, and relationships that are essential for overcoming the challenges identified in Section 3 (*Complement 𝒜CPS functionality*, *Achieve understandability*, and *Manage user attention*). See Fig. 4 for a summary of all of the identified concepts. The following subsections address those challenges in detail.

### *6.1. Complement 𝒜CPS functionality*

In Section 3, the first challenge stated that *"Humans must be integrated in an intimate collaboration with the system in order to assist in the execution of certain tasks in a correct and complete manner"*. The concepts required to address this question are identified in the following subsections.

#### *6.1.1. Tasks*

Tasks describe what a human is required to do in collaboration with the system in order to complement its functionality (the *Task* concept is the root concept in Fig. 4). A task requires a human with specific capabilities, knowledge, and background in order to be executed. These human features constitute a **human profile** (Fig. 4 shows the *Human Profile* concept and the *requires* relationship between *Task* and *Human Profile*).

The humans that fit the profile/s associated with the task are the only ones that can assist in performing that task. For example, in the case of the autonomous car, the *𝒯6 Takeover* task is associated with the *driver* profile, meaning that only a person with driving capabilities (s/he has a driving license) can help in performing that task. Fig. 5 illustrates the task throughout a notation that allows describing collaborative tasks. This diagram is named *task diagram*. It graphically represents a task by an oval linked to a human picture that represents the *driver* profile. A curved arrow describes the event that triggers the task. As we introduce new concepts, we will present the rest of the notation of the diagram.

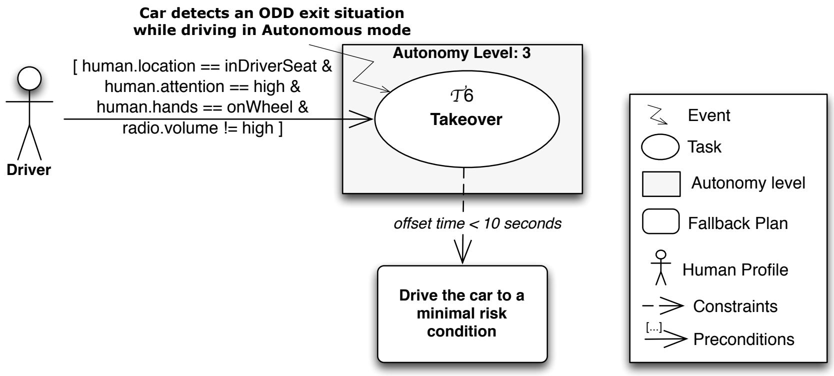

Car detects an ODD exit situation while driving in Autonomous mode
Autonomy Level: 3
[ human.location == inDriverSeat & human.attention == high & human.hands == onWheel & radio.volume != high ]
Driver
𝒯6 Takeover
offset time < 10 seconds
Drive the car to a minimal risk condition
Event
Task
Autonomy level
Fallback Plan
Human Profile
Constraints
[...]
Preconditions

Figure 5: Task Diagram: *𝒯6 Takeover* task

### *6.1.2. Human roles*

Humans can participate in tasks by playing different roles. The role that a human performs on a task determines the type of the task. Taking the capacities introduced in [13, 11], we abstracted the the roles that a human can play in *𝒜CPS* and proposed the following task types related with those roles:

1. **Monitoring tasks:** The human is in charge of performing actions to provide information or identify situations in the context of the running task (acting as sensors). Also, the human is in charge of providing data that is difficult or can not be monitored, or to correct and validate anomalous values that s/he identifies.
2. **Decision-making tasks:** The human can incorporate input into the decision-making process to provide better insight about the best way of performing that task. Therefore, the human has to decide between a set of options that the system has predefined or to decide by herself.
3. **Control and execution tasks:** The human is responsible for managing, executing, and leading the task execution and control (totally by herself or assisted by the system).

Tasks are defined within an **autonomy level** (represented by the corresponding concept in Fig. 4, which is related to the *Task* concept throughout the *defined in* relationship), which impacts on the role that the human can play in the task. In other words, a task can be placed in a different autonomy level and can require a different degree of human collaboration. In Fig. 5, a rectangle that contains the task is labeled with the autonomy level. This example shows a task ranked with *ℒ3* autonomy level.

### *6.1.3. Context situation*

An appropriate context situation (defined by context properties) is required to execute a task. This context situation can be considered as the **precondition** of the task (this concept is represented in Fig. 4 with the *has* relationship between it and the task). The precondition represents the context in which the system, its environment, and the human are prepared to execute the task under appropriate conditions in order to achieve proper task performance [52]. For example, when the *𝒜CPS* initiates *𝒯6 Takeover* task, the volume on the radio should be different from high to maximize the success of a safe takeover.

The *context* definition that Dey introduced in [53] is used to specify the semantics of the context concept: *"any information that can be used to characterize the situation of an entity. An entity is a person, place, or object that is relevant to the interaction between a user and an application, including the user and applications themselves"*. From this definition it is possible to derive the relevant properties of the context as the state of the physical environment (e.g., temperature, noise, etc.), the state of the resources regarding the system surroundings, the state of the system itself and the user's situation (e.g., its location, activity, etc.). The user's situation acquires special importance for achieving the *managing user attention* challenge. Section 6.2 (where this challenge is addressed) deals with this issue.

As can be seen in Fig. 5, the arrow between the human profile and the task indicates the relationship between those concepts and this arrow is labeled with the precondition (if exists) that enables the task execution. In the example of that figure, there is a precondition that must be fulfilled in order the driver to be able to safely participate in that task. The precondition includes the condition *the volume of the radio not high* among others, which are further explained in Section 6.2.2.

### *6.1.4. Constraints*

A task can have associated a **constraint**. Constraints are conditions that must be always fulfilled during the execution of the task (see the corresponding concept and the *has to fulfill* relationship between *Task* and *Constraint* in Fig. 4). They describe a required system state or a limit on task executing (for instance, an execution time limit). For example, in the case of $\mathcal{T}6$ *Takeover* task, it is associated with the constraint *before 10 seconds*, because if the human does not take control before 10 seconds, the $\mathcal{A}$*CPS* should cancel the task (see in Fig. 5 the arrow labeled with the constraint that leaves the task).

### *6.1.5. Fallback plan*

Having human-in-the-loop poses challenges related to the unexpected behavior of humans. Therefore, the design of tasks with human integration requires a safe guard, or a **fallback plan**, to be executed when the task is not working as planned (as Fig. 4 shows by means of the *fallback plan* concept and the *associated with* relationship). When executing a task, a fallback plan can be thrown due to:

- the non-presence of a human with the specific human profile,
- the incapacity to achieve the appropriate context situation (in fact, it occurs when the appropriate context situation is not accomplished within the limits forced by the constraint), or
- the execution of actions is not as expected (that is, it does not fulfill some constraint).

Fallback plans are, in turn, tasks. Some of the fallback plans are also autonomous tasks that may or may not require human integration. The fallback plan is represented in Fig. 5 by means of a rounded rectangle and is associated with a task with a dashed arrow in which the task constraint is specified. In the example of that figure, the fallback plan *Drive the car to a minimal risk condition* is specified with a constraint of 10 seconds for the task.

### *6.1.6. Actions*

Tasks can be decomposed into more fine-grained units named **actions** (see the *Action* concept and *composed of* relationship in Fig. 4). Each action represents a perceivable interaction performed either by the system or by the human, or an internal function carried out by the system. Actions are inter-related to determine a partial **order** in which they must be processed. Actions can also be associated with a constraint. In the same way that we have introduced in 6.1.4, the constraints of an action must be always fulfilled during the execution of the action (see the *has to fulfill* relationship between *Action* and *Constraint* in Fig.4).

For example, $\mathcal{T}6$ *Takeover* task can be decomposed into three actions: (1) the system notifies the human that s/he is required to take over control of the car, (2) the human confirms the takeover, and (3) the system transfers the control to the human. We define these three actions as the **core** actions that allow the human and the $\mathcal{A}$*CPS* to collaborate and to perform the task (it is one of the types of actions as Fig. 4 shows). We propose a graphical notation to represent the actions that compose a task. We call this notation the *action diagram*. Fig. 6 illustrates the actions that define $\mathcal{T}6$ *Takeover* task. Core actions are represented by a green circle. As we introduce new concepts, we will present the rest of the notation of the diagram.

## *6.2. Manage user attention & achieve understandability*

Second and third challenges in Section 3 state that *"$\mathcal{A}$CPSs must perform actions to get human attention in accordance with the level of human involvement required by using the correct level of obtrusiveness"* and *"a human must understand how the $\mathcal{A}$CPS operates and what is happening at the current moment or even the moment before"*. The concepts required to address these two issues are further explained in the following subsections.

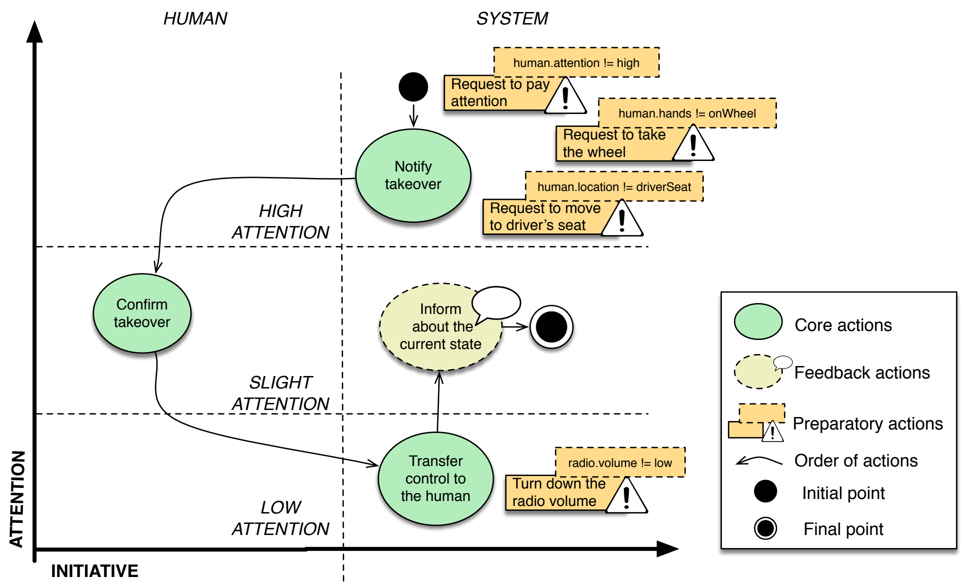


Figure 6: Action Diagram: $\mathcal{T}6$ *Takeover* task

### *6.2.1. Human factors*

In this work, the situation or context of the user acquires a special importance to determine whether or not the human is "prepared to" or "can be" integrated into the task to achieve the proper task performance. The "*opportunity-willingness-capability*" (OWC) model [52] is used to define the required characteristics of the human factors. The OWC model defines a set of human elements that constraints tasks performance:

- *Opportunity:* This identifies the set of variables that humans need to fulfill in order to attempt a task. These variables are related to the human's situation, such as the user's location, the activity that is being carried out, etc. In the autonomous car, possible variables and states for this element are *human.location = inDriverSeat*, *human.hands = onWheel*.

- *Willingness:* This indicates the human predisposition to perform the task. This factor is related to the human attention, stress level, load, motivation, or how busy the human is at the time. It is important to note that the user will not be able to perform the task if s/he does not have the necessary attentional resources available that the task requires. Possible variables and states for this element are *human.attention = high*, *human.stressLevel = low*.

- *Capacity:* This defines the human's skills and abilities that are necessary to successfully execute the task. It is related to the human's knowledge, level of experience or training, cognitive or physical skills, required devices, emotional states that may reduce the human's ability, etc. Possible variables and values for this element are *human.experience = high*, *human.hasSmartPhone = true*.

As shown in Fig. 7, to achieve proper human integration, the current human state must be the intersection of possible human states for which OWC can be evaluated as true.

OWC conditions are included in the precondition of a task (see the *includes* relationship from *Precondition* to the three concepts *Opportunity*, *Willingness* and *Capacity* in Fig. 4), which is evaluated before executing the task. Fig. 5 shows the precondition of $\mathcal{T}6$ *Takeover* task, which includes three conditions related to human factors (the human's location, human's attention and human hands' position).

The human elements that are included in a precondition have to be perceivable by the system somehow. In the autonomous car example, a RFID reader reads the RFID tags that the humans wear and evaluates if the human is in the driver's seat (the human location). The human attention is evaluated by means of a small cam that determines

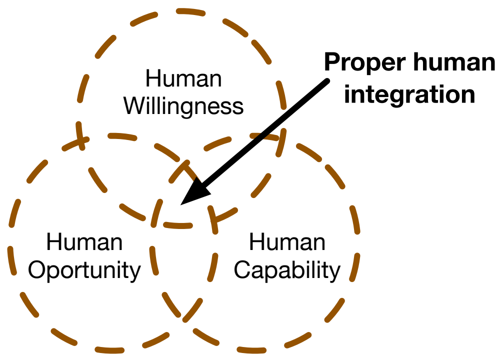


Figure 7: Graphical representation of proper human integration

the eyes position of the driver. Finally, a contact sensor that determines if the human has her/his hands on the wheel evaluates the human hand's position.

### 6.2.2. *Complementary actions*

In addition to the actions that perform the task functionality, tasks include actions that aim at preparing the system, the environment, and the human to perform the task in the required conditions. As mentioned, these conditions are inferred from the precondition. We call these actions ***preparatory actions*** and are executed before intrinsic actions. Thus, when a task initiates, if the precondition is not satisfied (i.e., the context is not suitable for performing the task), this task will execute the preparatory action/s corresponding to the none satisfied condition/s.

Consider again $\mathcal{T}6$ *Takeover* task. When the $\mathcal{A}CPS$ runs that task, if the human has no situational awareness, or s/he is in the passenger seat, or her/his hands do not take the wheel, or the volume on the radio is too loud, the three actions defined above for the task are insufficient to guarantee a safe takeover (the human is not likely to be aware of the notification about taking control of the car). The context situation requires additional actions in the task. Therefore, the task should include four preparatory actions: (1) the system warns the human to pay attention; (2) the system asks the human to move to the driver's seat; (3) the system asks the human to take the wheel; and (4) the system turns down the radio volume. Taking into account the current situation, the task can select which preparatory actions should be executed.

Moreover, tasks can include actions of another kind that aim at achieving the second challenge *achieve understandability*. These actions are intended to provide (1) feedback regarding what the task is doing, and (2) feedforward so that the human properly receives knowledge to collaborate. We call these actions ***feedback actions***. For example, in $\mathcal{T}6$ *Takeover* task, once the human takes control of the car, the system informs the driver of that situation. Also, feedback actions can depend on specific human factors in order to avoid obtrusiveness. A condition can be attached to the task to determine whether the action must (or must not) be executed.

Preparatory and feedback actions can be associated with constraints in the same way we have introduced in Section 6.1.4 (these are the other two types of actions as Fig. 4 shows).

Fig. 6 shows the action diagram that includes the representation of core actions together with preparatory and feedback actions. Preparatory actions are represented by an orange square with a dashed square above it when they are associated with a precondition; feedback actions are represented by a yellow dashed circle. The order of the actions is not represented between preparatory actions and the other actions, since preparatory actions are executed first and only when the precondition is not fulfilled. Initial and final actions are marked with initial and final points. The semantics of the quadrants represents the obtrusiveness level, and are explained in detail in the following section.

### 6.2.3. *Attention and initiative*

Actions usually entail an interaction between the human and the system. The actions that involve an interaction should be designed to manage human attention resources in order to get the human's attention while at the same time avoiding to overwhelm the human. The HCI has proposed solutions to manage of human attention based on levels of automation and workload [54, 10, 55]. We analyzed and designed the actions from two points of view: 1) who initiates the interaction (the system or the human); and 2) the attention resources needed to assist the interaction (attentional demand imposed on the human) (see the *designed in* relationship between *Action* and the two concepts *Initiative* and *Attention* in Fig.4). We consider this specification of the **initiative** and the **attention** (attentional demand) to be adequate since these are factors that vary independently. For example, an interaction of feedback (initiated by the system) may require more attention or less depending on the importance/criticality of the information to be offered. In the same way, a preliminary interaction could be initiated by the system (if it tries to capture the user's attention) or by the human (if the system waits for an answer from the human to confirm that s/he is prepared). Establishing an initiative and an attention level help designers in selecting the most suitable interaction mechanisms.

In Fig. 6 the vertical axis places the attention dimension and the horizontal axis displays the initiative dimension. In this example, the initiative dimension is divided into two segments according to who initiates the action: the *human* or the *system*. The attention dimension is divided into three segments: *low attention* (if the user is required to make some effort to perceive the interaction), *slight attention* (if the action implies slight demand of the human attentional resources), and *high attention* (if the user is required to be fully conscious of the interaction). In each quadrant, the designers should locate the different actions for performing the task. The first action in the figure is *Notify takeover* action that demands high attention and is performed by the system. The second action, *Confirm the takeover* action, demands slight attention and is performed by the human. Then, there are two actions performed by the system: *Transfer control to the human* action that just demands low attention (from the human) and *Inform about the current state of the car* action with a slight level of attention. The level of attention associated with the action will guide the selection of the concrete interaction mechanism in further development phases, such as proposed in [56]. The levels that demand more attention must be associated with interaction modalities more obtrusive or that draw more attention. Conversely, levels that demand less attention must be associated with discreet interaction modalities.

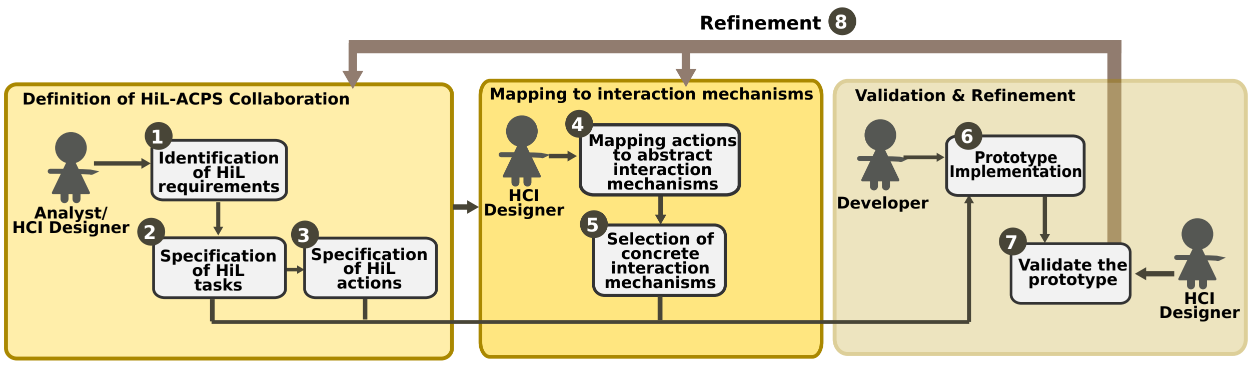
### *6.3. Applying the conceptual framework to build HiL-$\mathcal{A}$CPS designs*

The concepts presented in Sections 6.1 and 6.2 constitute the conceptual framework that supports the specification of *HiL-$\mathcal{A}$CPS* designs. This subsection provides a guide for applying the framework in order to build these designs. To obtain an *HiL-$\mathcal{A}$CPS* design, the following steps must be taken (in the following order):

1. **Identify and define a set of collaborative tasks.** It is necessary to determine which tasks of the *$\mathcal{A}$CPS* require collaboration between the system and the human. Each collaborative task is defined within an autonomy level. This task definition involves specifying (in any order):
   - the *human profile*, which participates in the task by playing a specific human role
   - the *precondition*, which uses human factors for its specification
   - the *fallback plan*, which can have an associated *constraint* that, if not satisfied, triggers the fallback plan

   This specification is graphically represented by means of the task diagram.
2. **Decompose each collaborative task into actions.** The following actions must be identified (in the following order):
   - the *core actions* that are necessary to reach the task goal
   - the *preparatory actions* that are required to get the proper human attention to fulfill the precondition associated to the task
   - the *feedback actions* that allow the user to understand how the system is operating
3. **Associate each action with an obtrusiveness level.** The specification of the actions and their obtrusiveness level is graphically represented by means of the action diagram.

The proposed specification of *HiL-$\mathcal{A}$CPS* designs, which is the main concern of our proposal, is part of a methodological approach to specify and build *HiL-$\mathcal{A}$CPS*. Figure 8 illustrates the complete proposal. Specifically, Steps 2 and 3 deal with the specification of the *HiL-$\mathcal{A}$CPS* collaboration by applying the proposed conceptual framework. Once the *HiL-$\mathcal{A}$CPS* collaboration is built, the specified actions should be mapped to abstract interaction mechanisms (Step 4), according to the level of attention attached to the actions. Then, the selection of the concrete interaction mechanisms (Step 5) is necessary to implement the collaborative tasks (since it establishes the specific devices and/or



Refinement 8
Definition of HiL-ACPS Collaboration
Analyst/ HCI Designer
1 Identification of HiL requirements
2 Specification of HiL tasks
3 Specification of HiL actions
Mapping to interaction mechanisms
HCI Designer
4 Mapping actions to abstract interaction mechanisms
5 Selection of concrete interaction mechanisms
Validation & Refinement
Developer
6 Prototype Implementation
7 Validate the prototype
HCI Designer

Figure 8: The '*big picture*' of the proposal

interaction mechanisms that support the interaction between humans and systems). Steps 4 and 5 are out of the scope of the present work due to space constraints (an example of this mapping can be seen in [56]). Finally, a key step of this proposal is validation and refinement (Steps 6, 7 and 8) where system prototypes are built (following the specifications of the conceptual framework) to validate the design carried out in the previous steps. The validation allows: the refinement of the design (HiL tasks, HiL actions, attention levels, feedback messages, etc.), and the consolidation of the specific interaction mechanisms that are most suitable.

In conclusion, the conceptual framework has been designed to provide models and techniques to allow: 1) the specification of tasks in which a human is required to participate in order to complement the functionality of the *ACPS*s; and 2) the definition of an appropriate interaction to allow a human to execute a task in collaboration with the system, in order to minimize system obtrusiveness and maximize system understandability.

## 7. Operationalization of the *HiL-ACPS* collaboration

This section presents an execution model that defines how the human collaboration should be executed when developing fully functional (or fully operative) solutions. This execution model is based on the extended *HiL-ACPS* behavior presented in Section 5.2 and establishes the control flow that underpins the execution of collaborative tasks.

The process of involving the human can be abstracted by the following steps (See the flow chart in Fig. 9):

**Step 1: Monitoring**. The *ACPS* is continuously monitoring its environment to identify situations in which it should take an action. If it detects that a collaborative task needs to be executed, it raises an event that triggers the need for the human's help in carrying out that task.

**Step 2: Is the human profile available?** The presence of a human that fits the human profile is verified:

- If the human exists, the system goes to Step 3.
- If the human does not exist, the system goes to Step 6.

**Step 3: Is the context suitable for collaboration?** The context conditions that determine the suitability of the context for a proper human collaboration are verified:

- If the context conditions are fulfilled, it means that the context is suitable to perform the execution of the collaborative task. The system goes to Step 5.
- If some context conditions are not fulfilled, it means that the context is not optimal to perform the task execution. The system goes to Step 4.

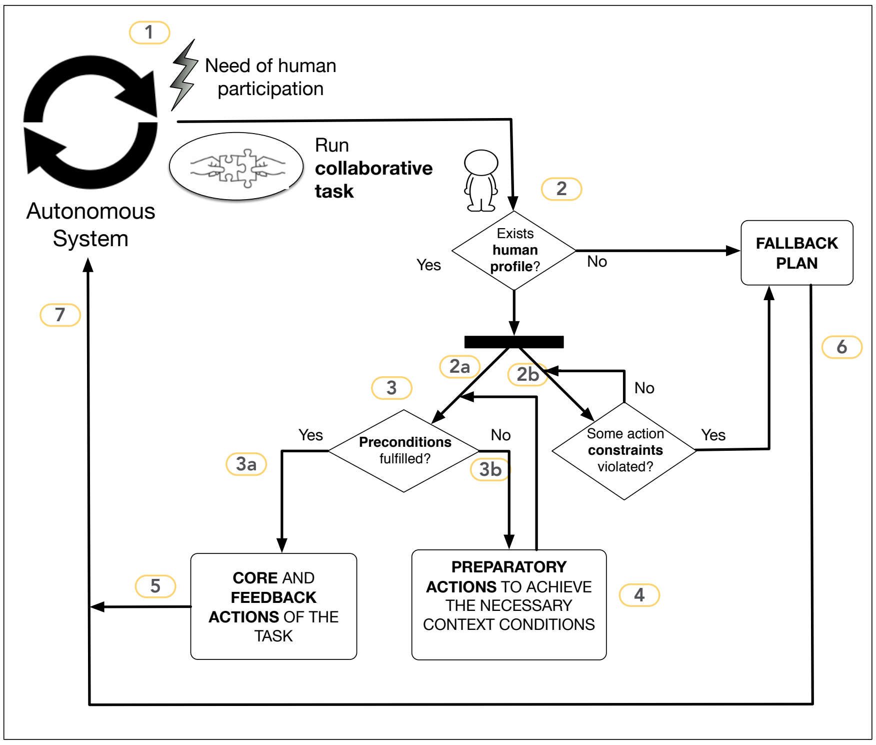

1
Need of human participation
Run collaborative task
Autonomous System
2
Exists human profile?
Yes
No
FALLBACK PLAN
7
6
2a
2b
3
No
Preconditions fulfilled?
Yes
No
Some action constraints violated?
Yes
3a
3b
CORE AND FEEDBACK ACTIONS OF THE TASK
PREPARATORY ACTIONS TO ACHIEVE THE NECESSARY CONTEXT CONDITIONS
5
4

Figure 9: Execution model of the *ACPS*

**Step 4: Preparing the context for optimal configuration**. The system executes actions (using interaction mechanisms) to change the context in order to achieve the suitable context conditions (e.g., requiring human attention and/or preparing the environment in a way that enables the human to carry out the task). The system goes to Step 3.

**Step 5: Human collaboration**. The system and the human perform the task collaboratively. To do this, the actions that make up the task are executed in the order specified in the action diagram. These actions can be:

- *System-initiated actions*. The system interacts with the human through an output interaction mechanism that supports the attention level specified for the action.
- *Human-initiated actions*. The human interacts with the system by means of an input interaction mechanism that supports the attention level specified for the action.

After the execution of some core actions, the system can perform feedback actions that are designed to support a specific attention level. Feedback actions inform the humans without overwhelming them, thus facilitating understanding of the executed/performed actions.

When all of the specified actions are executed, the system goes to Step 7.

**Step 6: Fallback plan – something is not working as planned**. The system executes a fallback plan or alternative actions that leave the system in a safe state. The system goes to Step 7.

**Step 7: Returning to autonomous mode**. Once the task is completed, the system automatically switches to autonomous mode (until it detects a new situation that requires human participation).

Each action of the task must be strictly carried out without violating the task constraints (temporary, contextual, etc.). The *HiL-ACPS* is responsible for checking them at any moment during the execution of those actions (Steps 4 and 5). If these constraints are not satisfied at any time, Step 6 (the fallback plan) must be executed.

The execution model steps provide the operational support for guiding the implementation of the collaborative tasks in *HiL-$\mathcal{A}$CPS* systems. The execution model is also essential in the validation of the proposal that is presented below since it guides the implementation of the prototype of the autonomous car.

## 8. Evaluation of the proposal

The goal of this evaluation is to demonstrate that this solution can be used to properly design *HiL-$\mathcal{A}$CPS* systems in which the human co-works with the autonomous system in a way that the three design challenges are fulfilled. The evaluation is focused on two main questions:

1. Can a functional *HiL-$\mathcal{A}$CPS* solution actually be implemented according to the collaborative task specifications?
2. Does the functional *HiL-$\mathcal{A}$CPS* solution meet its specification in order to achieve the three design challenges?

Therefore, we have looked for evidence that functional *HiL-$\mathcal{A}$CPS*s can be implemented according to the specifications built by applying the proposed conceptual framework and that these systems are capable of complementing the ACPS functionality with users, provide feedback, and avoid obtrusiveness as defined in the specification. In order to assess these questions, we have designed and implemented a prototype of the autonomous car described in Section 4. To do this, we used the conceptual framework and the execution model to design and implement the collaborative tasks of the autonomous car. Then, we carried out a set of tests to check whether the car prototype behaves as defined in the specification of the tasks. The goal of the tests was to determine whether the collaborative tasks provide support to the design challenges in the way specified.

The test process involved the following steps, which are described in detail in the following subsections:

1. *Build the prototype of the autonomous car.* A prototype of the autonomous car case study was built, applying the proposed approach.
2. *Design of the test.* A set of test cases was designed to determine whether the behavior of the collaborative tasks of the autonomous car is compliant with their specifications.
3. *Execution of the test.* The car prototype was used to simulate driving with human collaboration and to check the behavior of the car in the test cases.
4. *Analyze the results.* We analyzed the results in order to determine whether the execution of the tasks satisfied the three design challenges identified.

### *8.1. Build the prototype of the autonomous car*

In this subsection we apply the proposal to implement a functional prototype of the autonomous car, which is described as the case study. This prototype is mainly focused on providing both integration and collaboration mechanisms within the human and the autonomous tasks. Following subsections introduce the design of the prototype and its implementation.

#### *8.1.1. Design of the prototype*

This subsection focuses on describing the design of *$\mathcal{T}$1 Supervised Autonomous Driving* and *$\mathcal{T}$6 Takeover* using the *action* diagram notation and *task* diagram notation; the design of the rest of the collaborative tasks is performed in the same way and is not shown due to space limitation.

- *$\mathcal{T}$1 Supervised Autonomous Driving*: Figure 10 shows the specification of this task. The task requires a *driver* to be the human profile. The *$\mathcal{T}$1* precondition requires a car scenario in which the driver is located in the driver's seat, with his/her hands on the wheel, in an attentive mode (facing forward). The fallback plan defined for the task is: *Drive the car to a minimal risk condition*. In addition, this task has defined a constraint to execute the fallback plan if the driver gets distracted or removes his/her hands from the wheel for a period of 5 seconds ([47], concluding that this situation is not desirable, even for a short period of time). The specific collaboration between the human and the car in the *$\mathcal{T}$1 Supervised Autonomous Driving* task is described in Fig. 11. As the figure illustrates, three preparatory actions will be performed (in *high attention* level) if the driver is not able to

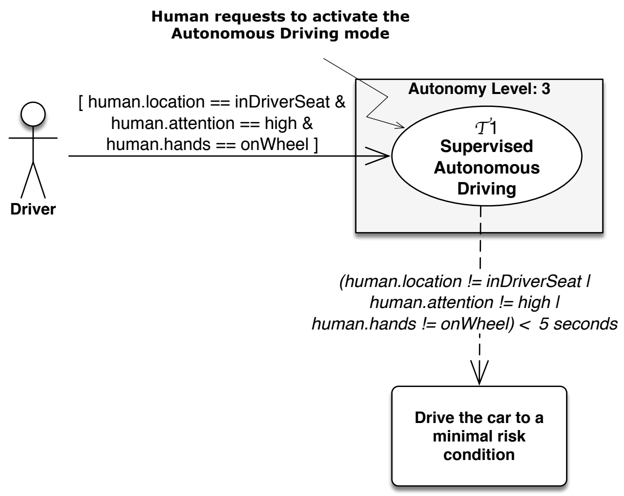

Human requests to activate the Autonomous Driving mode
Autonomy Level: 3
[ human.location == inDriverSeat & human.attention == high & human.hands == onWheel ]
Driver
$\mathcal{T}1$ Supervised Autonomous Driving
(human.location != inDriverSeat | human.attention != high | human.hands != onWheel) < 5 seconds
Drive the car to a minimal risk condition

Figure 10: Task Diagram: *$\mathcal{T}1$ Supervised Autonomous Driving* task

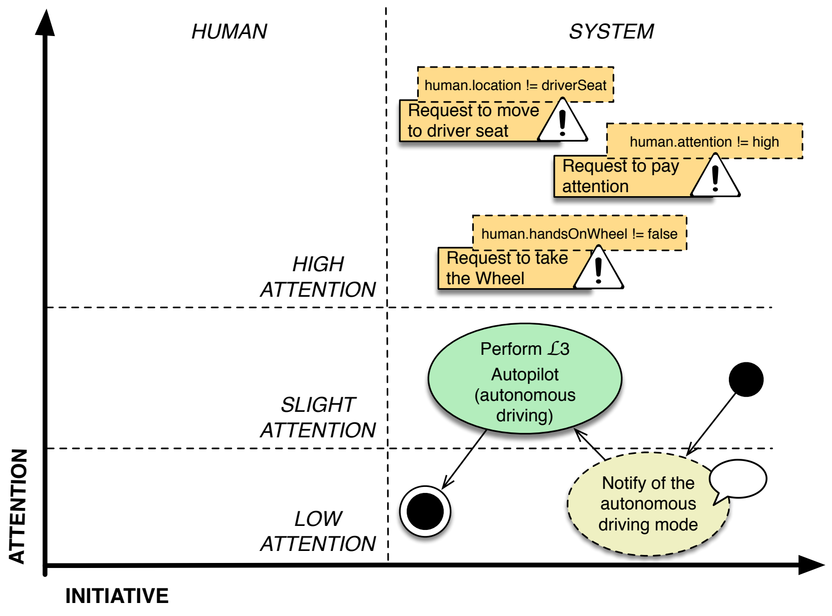

HUMAN
SYSTEM
human.location != driverSeat
Request to move to driver seat
human.attention != high
Request to pay attention
human.handsOnWheel != false
Request to take the Wheel
HIGH ATTENTION
SLIGHT ATTENTION
LOW ATTENTION
Perform $\mathcal{L}3$ Autopilot (autonomous driving)
Notify of the autonomous driving mode
ATTENTION
INITIATIVE

Figure 11: Action Diagram: *$\mathcal{T}1$ Supervised Autonomous Driving* task

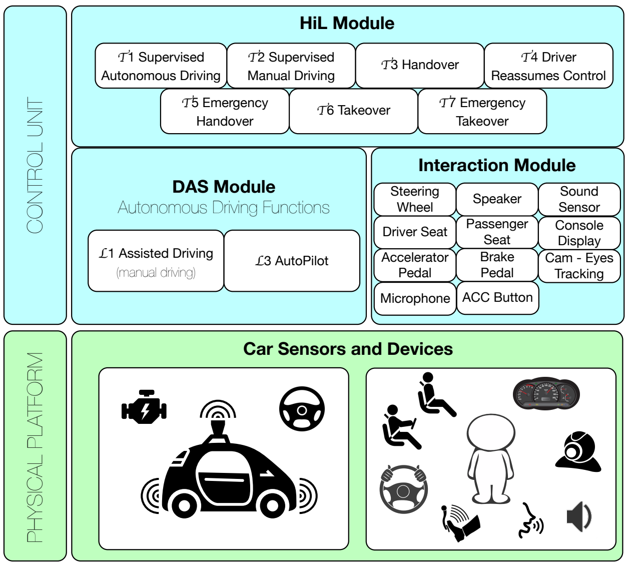


Figure 12: Car Simulator Architecture

properly collaborate with the task. Once the human is ready to collaborate, the car informs the driver about the AutoPilot engagement in *low attention* level. Then, it engages the *ℒ3 AutoPilot* function provided by the *DAS Module*.

- *𝒯6 Takeover*: Figure 5 shows the specification of this task. The required human profile is also a *driver*. *𝒯6* requires the same precondition as the *𝒯1* task does, but also requires the volume of the radio not being high. However, the *𝒯6* task defines a constraint to execute the same fallback plan if the human (driver) does not get control of the car within 10 seconds. Figure 6 shows the specification of the action diagram for this task. It contains three preparatory actions of the system for the *high attention* level. It also has another preparatory action *Turn down the radio volume*, in the *low attention* level. Once the human driver is ready to collaborate, the system sends him/her a *Notification to take control*. Then, when the *Human confirms the car takeover*, the car will *Transfer control to the human* and confirm this through a feedback action in the *slight attention* level.

Once the collaborative tasks of the case study were specified, they were implemented within the car prototype as explained below.

### *8.1.2. Implementation of the prototype*

Figure 12 shows an overview of the prototype architecture, which is made up of two subsystems: a *Physical Platform*, which monitors the car, its environment, and the human; and a *Control Unit*, which is responsible for the autonomous functions, the human interaction mechanisms, and the collaborative tasks.

The **Physical Platform** is composed of a set of sensors and devices that allows the *Control Unit* to know the current situation of the driving environment at any time, and to detect events or situations that happen to the system. This platform also provides functional devices that allow the autonomous functions to be simulated. An Arduino MEGA 2560 acts as the physical platform device controller of the car (shown in Fig. 15). This controller is responsible for: providing the *Control Unit* with the state of the different *sensors and devices* of the car, notifying of physical events (relevant changes in those sensors and devices); and actuating on controllable devices. The car simulator uses the following devices:

- a set of RFID tags representing different human profiles,
- RFID readers for the driver's seat and the passenger's seat, each of which monitors which human is seated by detecting the RFID tags,

- a contact sensor that determines if the human has his/her hands on the wheel.
- two small potentiometers that monitor the pressure of both the brake and the accelerator pedals,
- a sound sensor that senses the sound level inside of the car,
- a small cam that is used to "see" the eye position of the driver (this cam is actually connected to the Raspberry Pi),
- a speaker to provide (voice) feedback,
- a microphone to recognize voice input,
- a physical button to switch the autonomous driving mode,
- a car console to show (visual) feedback.

This controller was developed using the Arduino Processing/C language. It mainly executes a loop that: processes incoming requests (via Serial communications) to obtain the state and measures of the sensors (*who is seated at the driver seat?*); checks the current state or measures of any connected device (*what is the current sound level?*); and sends notifications (via Serial communications) if a relevant change in any device occurs (*the driver has put his/her hands on the wheel*).

The **Control Unit** of the car performs both the automated driving tasks and the collaborative tasks between the driver and the car. It is executed by a Raspberry Pi 3 Model B+. This prototype was developed using Java with OSGi and connects via UART-Serial to Arduino. It executes the *DAS Module*, the *Interaction Module*, and the *HiL Module*. We describe these modules in detail below.

- **The DAS Module**. This module implements the skeleton of the automatic functions of the car ($\mathcal{L}$*1 Assisted Driving* and $\mathcal{L}$*3 AutoPilot*) and offers an interface to allow the HiL Module to activate or deactivate them. The details of the internal implementation of these functions is out of the scope of this prototype. The module is also responsible for detecting situations in which the autonomic tasks cannot be fully performed, according to the identified autonomous driving limitations (see Section 4.1). In these situations, the module will require the HiL module to activate a collaborative task. For instance, when the car is driving in $\mathcal{L}$*3 AutoPilot* mode and is about to enter a city, the DAS module identifies that the car cannot continue the autonomous driving and requires the *HiL Module* to trigger the corresponding collaborative task (in this case, $\mathcal{T}$*6 Takeover*).

- **The Interaction Module**. This module is responsible for implementing the communication mechanisms with the human, which include both explicit interactions (sounds and messages) and implicit interactions (human gestures and actions). To perform this bi-directional communication between the human and the car, this module makes use of physical devices from the *Physical Platform* such as speakers, a small cam (to track the driver's eye position), a steering wheel contact sensor, a sound sensor, etc.

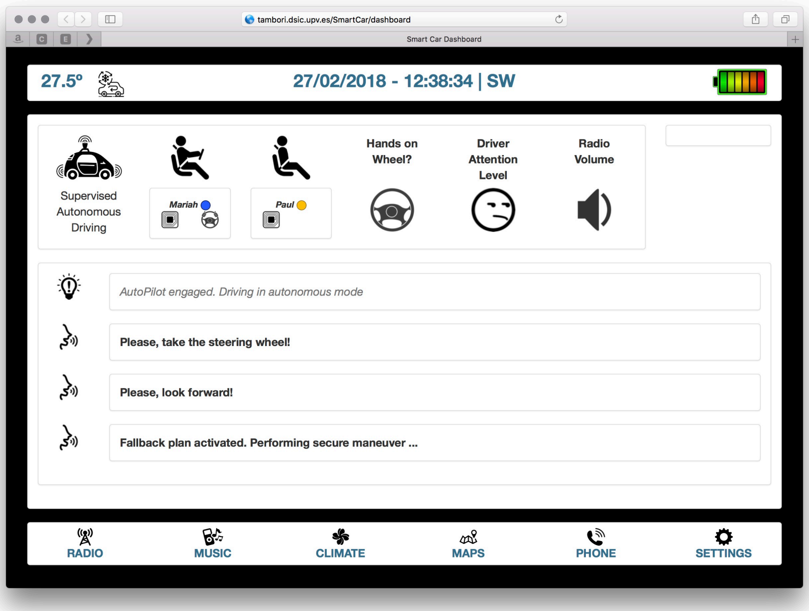
  The specific interaction mechanisms available in the prototype that support each attention level are the following[1]:

  **High Attention Level.**

  - Output interaction modalities: voice feedback using speakers and a highlighted text message with an icon on the car dashboard.
  - Input interaction modalities: haptic input using a physical button and voice recognition by means of a microphone.

  **Slight Attention Level.**

[1]Note that output interaction modalities are used for system-initiated interactions and input interaction modalities are used for human-initiated interactions



AutoPilot engaged. Driving in autonomous mode

Please, take the steering wheel!

Please, look forward!

Fallback plan activated. Performing secure maneuver ...

Figure 13: Car simulator dashboard web interface

  - Output interaction modalities: a sound/'beep' on the speakers and a highlighted text message on the car dashboard.
  - Input interaction modalities: pressure detection of the accelerator and brake pedals as well as the steering wheel.

**Low Attention Level.**

  - Output interaction modalities: a text message on the car dashboard.
  - Input interaction modalities: gesture recognition using the small cam.

The *Interaction Module* also monitors the notifications that come from the *Physical Platform* devices that are related to the human's context situation and communicates them to the *HiL Module*.

Finally, the interaction module implements a small web application (using HTML/JQuery and Bootstrap) that simulates a car dashboard (see Fig. 13). This dashboard displays the messages and the current state of the car (i.e., the task or functionality that is being performed, who is in the car, the driver's attention level and the environmental sound volume).

- **The HiL Module**. This module contains the implementation of the collaborative tasks. To illustrate this implementation, we show the details of the $\mathcal{T}6$ *Takeover* implementation (Fig. 6 shows the action diagram and Fig. 5 shows the task diagram for this task):

  - The task starts by checking if the required **human profile** (a driver) is available. This condition is evaluated by monitoring the car seat devices (which use RFID tags to identify the human profile). If the human is not available, the fallback plan is executed; if the human is available, the task continues its execution.
  - Next, the four **preconditions** of the task are verified:
    1. Is there a human in the driver's seat? The RFID tag of the driver's seat is used to validate this condition. If this condition is false, the **preparatory action** *Request the human to move to the driver's seat* is executed using the voice feedback on the speakers and showing a message with an icon on the dashboard (since the action has been specified at the high attention level).

2. Is the human's attention high? The small cam that tracks the driver's eyes is used to evaluate this condition. If this condition is false, the **preparatory action** *Request the human to pay attention* is executed using the voice feedback on the speakers and showing a message with an icon on the dashboard (since the action has been specified at a high attention level).
3. Are the hands of the human on the wheel? The contact sensor of the steering wheel is used to validate this condition.If this condition is false, the **preparatory action** *Request the human to take the wheel* is executed using the voice feedback on the speakers and showing a message with an icon on the dashboard (since the action has been specified at a high attention level).
4. Is the radio volume high? The sound sensor volume of the speaker is used to validate the condition. If this condition is true, the **preparatory action** *Turn down the radio volume* is executed showing a message on the dashboard (since the action has been specified at the low attention level).

When the four preconditions are fulfilled, the task continues to the next step.

– Then, the task implements the collaboration between the car and the human to perform a safe takeover. This is done by executing the specified actions in the action diagram as follows:

1. The task executes the **core action** *Notify the takeover* using the voice feedback on the speakers and showing a message with an icon on the dashboard (since the action has been specified at the high attention level).
2. The human then initiates the **core action** *Confirm takeover* by actuating the steering wheel or the brake pedal (since the action has been specified at the slight attention level). When any of these occur, the action is completed.
3. The task then continues to execute the **core action** *Transfer control to the human*. A text message is shown on the dashboard (since the action has been specified at the low attention level) in order to make the human aware of the action.
4. Finally, the task executes the **feedback action** *Inform the driver of the current state* using a beep from the speakers and showing a message on the dashboard (since the action has been specified at the slight attention level) in order to provide the appropriate understanding of what is currently happening.

During the execution of these actions, the preconditions are continuously being evaluated. If some precondition becomes false, the corresponding preparatory action is executed.

– Simultaneously with these steps, the **constraint** *offset time<10 sec* is continuously being evaluated. If the constraint is violated, the task is cancelled and the **fallback plan** is executed.

This implementation leads to a car prototype that performs its complete functionality by co-working with the driver by means of the collaborative tasks. During the task execution, the driver receives system notifications through one of the output interaction mechanisms (speakers or dashboard) and actuates through one of the input interaction mechanisms (the wheel or the pedals). At all times, the car is aware of the human context situation (the human's location, the human's attention, and the human's hand position) by using the different physical sensors and devices.

### *8.2. Design of the test*

Once the prototype was implemented, we built a set of tests to evaluate it. Since the goal is to evaluate the three design challenges, the following questions need to be assessed:

- **Q1 - Complement ACPS functionality**. Do the collaborative tasks of the autonomous car prototype achieve their goal and finalize?
- **Q2 - Provide Understandability**. Do the collaborative tasks of the autonomous car prototype provide the user with feedback according to what was specified in the models?
- **Q3 - Manage obtrusiveness**. Do the collaborative tasks of the autonomous car prototype perform the interactions/actions at the specified level of attention?

Table 4: Test Case for *𝒯1 Supervised Autonomous Driving*

| **Id** | **Class** | **Precondition** | **Test Steps** |
|---|---|---|---|
| `TC01` | *HC&FC* | system.mode = manual driving, human.attention = high, human.hands = onWheel, human.position = inDriverSeat | The human requests the system to activate the autonomous driving mode, The human takes the wheel |
| `TC02` | *HC&NFC* | system.mode = manual driving, human.attention = low, human.hands != onWheel, human.position = inDriverSeat | The human requests the system to activate the autonomous driving mode, The human takes the wheel, The human pays attention |
| `TC03` | *NCH&FC* | system.mode = manual driving, human.attention = high, human.hands = onWheel, human.position = inDriverSeat | The human requests the system to activate the autonomous driving mode |
| `TC04` | *NCH&NFC* | system.mode = manual driving, human.attention = low, human.hands != onWheel, human.position = inDriverSeat | The human requests the system to activate the autonomous driving mode |

In order to obtain answers to these questions, we defined a battery of test cases to effectively test the collaborative tasks. To ensure high testing coverage, the black box testing technique *equivalence partitioning* [57] was used to define the test cases. This allowed us to test the car functionality without peering into its internal structures. Equivalence partitioning divides the input domain of a program into equivalence classes or groups that are expected to exhibit similar behavior. In order to define these groups, two variables that impact the behavior of the collaborative tasks were considered: the human collaboration and the system context (which includes the human's context situation).

The following equivalence classes were defined:

- *Class HC&FC (Human Collaboration and Favorable Context).* The human reacts to the system's requests in a situation where the system state is ready to perform the task.
- *Class HC&NFC (Human Collaboration and Non-Favorable Context).* The human reacts to the system's requests in a situation where the system state is not ready to perform the task.
- *Class NCH&FC (Non-Collaborative Human and Favorable context).* The human does not react to the system's requests in a situation where the system state is ready to perform the task.
- *Class NCH&NFC (Non-Collaborative Human and Non-Favorable context).* The human does not react to the system's requests in a situation where the system state is not ready to perform the task.

One test case had to be designed to explore each equivalence class, which resulted in four test cases. The four test cases had to be applied for each collaborative task, so we had a total of 28 (4 x 7) test cases.

The following subsections describe how the input data and expected results were specified for the test cases.

### *8.2.1. Input data*

First, the input data for each test case was defined based on its equivalence class. Table 4 shows the test cases for *𝒯1 Supervised Autonomous Driving*. The table shows the data for the four equivalence classes; each class maps to a row. The four columns of the table show: 1) the *id* that identifies the test; 2) the *class* of the test; 3) the *precondition* that specifies the setup that is needed for executing the test; and 4) the *test steps*, which contain a list of steps that the human and the system should perform. The list of steps is sorted according to the order in which they should be executed. This data was used to execute the tests that belong to *𝒯1 Supervised Autonomous Driving*.

Table 5: Expected Results for $\mathcal{T}$*1 Supervised Autonomous Driving*

| id | Question | Expected Results |
|---|---|---|
| **TC01** | **Q1** | - The system starts $\mathcal{T}$*1 Supervised Autonomous Driving*,<br>- Autonomous driving is successfully performed |
| | **Q2** | - The system notifies the human the autonomous driving mode |
| | **Q3** | - The system notifies the human of the autonomous driving mode by showing a text message on the dashboard |
| **TC02** | **Q1** | - The system starts $\mathcal{T}$*1 Supervised Autonomous Driving*,<br>- Autonomous driving is successfully performed |
| | **Q2** | - The system requests the human to take the wheel,<br>- The system requests the human to pay attention,<br>- The system notifies the human of the autonomous driving mode |
| | **Q3** | - The system requests the human to take the wheel and to pay attention using a voice feedback on the speakers and by showing a highlighted text message with an icon on the dashboard,<br>- The system notifies the human of the autonomous driving mode by showing a text message on the dashboard |
| **TC03** | **Q1** | - The system starts $\mathcal{T}$*1 Supervised Autonomous Driving*,<br>- Autonomous driving is successfully performed |
| | **Q2** | - The system notifies the human of the autonomous driving mode |
| | **Q3** | - The system notifies the human of the autonomous driving mode by showing a text message on the dashboard |
| **TC04** | **Q1** | - System starts $\mathcal{T}$*1 Supervised Autonomous Driving*,<br>- The fallback plan is activated |
| | **Q2** | - The system requests the human to take the wheel,<br>- The system requests the huaman to pay attention |
| | **Q3** | - The system requests the human to take the wheel and to pay attention using a voice feedback on the speakers and by showing a highlighted text message with an icon on the dashboard |

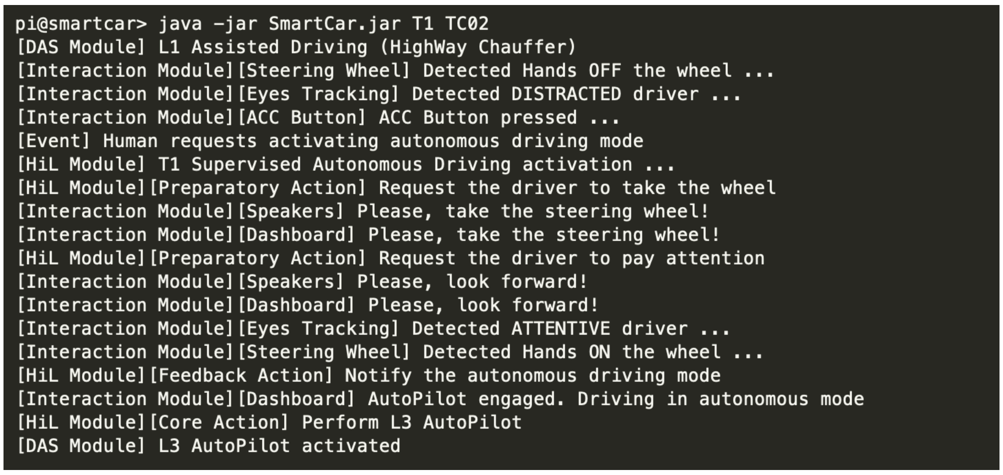
### *8.2.2. Expected results*

The expected results were specified based on the specific question under study (Q1, Q2 or Q3). Table 5 shows the expected results for $\mathcal{T}$*1 Supervised Autonomous Driving*, which were defined manually according to the specifications defined for the autonomous car. Similar tables were defined for the rest of the collaborative tasks.

## *8.3. Execution of the test*

The tests cases were executed by a user playing the driver role and interacting with with the autonomous car prototype. The drivers were asked to start in a given situation and to react or not to react to what the car was about to ask them, according to the test case that was about to be executed.

For instance, let's consider the execution of the $\mathcal{T}$*1 Supervised Autonomous Driving* - `TC02` test case. The driver was asked to start with her hands off the wheel not facing forward and to react to the car requests. The scenario began with the car prototype in $\mathcal{L}$*1* "Assisted Driving". The human requested the system to activate the autonomous driving mode and $\mathcal{T}$*1 Supervised Autonomous Driving* was initiated. Then, since the task precondition was initially unsatisfied, the system asked the driver to take the wheel (*Request the driver to take the wheel*) and asked the driver to pay attention (*Request the driver to pay attention*) by means of voice feedback on the speakers and by showing a message with an icon on the dashboard. As requested, the driver took the wheel and faced forward. The system notified the human the current autonomous driving mode (*Notify the current autonomous driving mode*) by means of a text message on the dashboard. Then, the system activated $\mathcal{L}$*3* "AutoPilot" (*Perform* $\mathcal{L}$*3 "AutoPilot"*) without providing any new notification or action to the driver. Fig. 14 shows the execution trace related to this scenario. Note that this result matches the one expected (see row 2 of Table 5).



```
pi@smartcar> java -jar SmartCar.jar T1 TC02
[DAS Module] L1 Assisted Driving (HighWay Chauffer)
[Interaction Module][Steering Wheel] Detected Hands OFF the wheel ...
[Interaction Module][Eyes Tracking] Detected DISTRACTED driver ...
[Interaction Module][ACC Button] ACC Button pressed ...
[Event] Human requests activating autonomous driving mode
[HiL Module] T1 Supervised Autonomous Driving activation ...
[HiL Module][Preparatory Action] Request the driver to take the wheel
[Interaction Module][Speakers] Please, take the steering wheel!
[Interaction Module][Dashboard] Please, take the steering wheel!
[HiL Module][Preparatory Action] Request the driver to pay attention
[Interaction Module][Speakers] Please, look forward!
[Interaction Module][Dashboard] Please, look forward!
[Interaction Module][Eyes Tracking] Detected ATTENTIVE driver ...
[Interaction Module][Steering Wheel] Detected Hands ON the wheel ...
[HiL Module][Feedback Action] Notify the autonomous driving mode
[Interaction Module][Dashboard] AutoPilot engaged. Driving in autonomous mode
[HiL Module][Core Action] Perform L3 AutoPilot
[DAS Module] L3 AutoPilot activated
```

Figure 14: Console of the Supervised Autonomous Driving for `TC02`

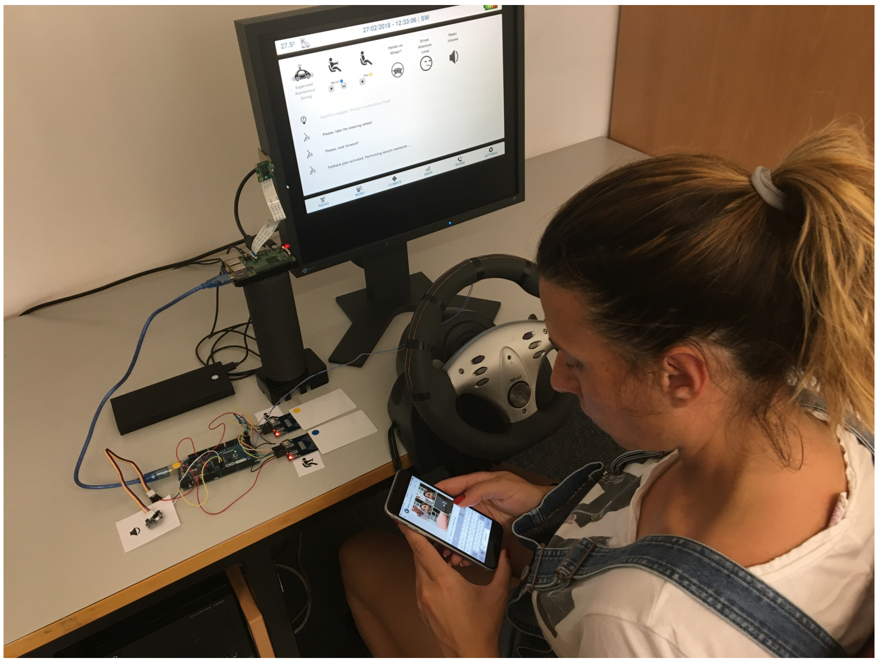
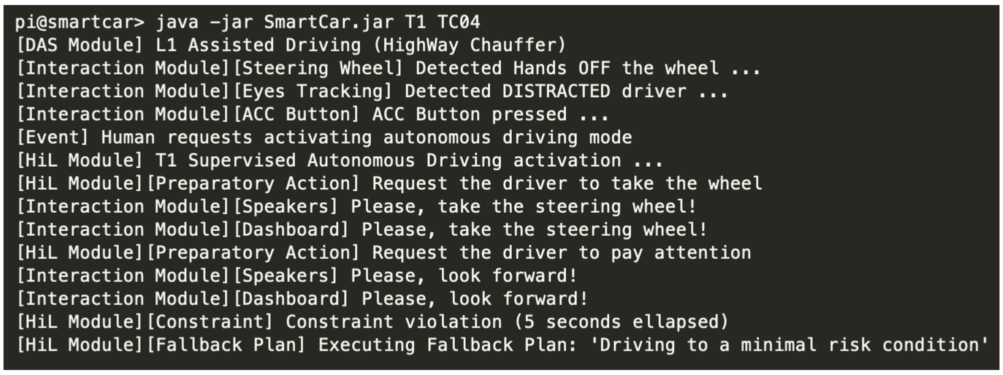


Figure 15: Scenario of the Supervised Autonomous Driving for `TC02`



```
pi@smartcar> java -jar SmartCar.jar T1 TC04
[DAS Module] L1 Assisted Driving (HighWay Chauffer)
[Interaction Module][Steering Wheel] Detected Hands OFF the wheel ...
[Interaction Module][Eyes Tracking] Detected DISTRACTED driver ...
[Interaction Module][ACC Button] ACC Button pressed ...
[Event] Human requests activating autonomous driving mode
[HiL Module] T1 Supervised Autonomous Driving activation ...
[HiL Module][Preparatory Action] Request the driver to take the wheel
[Interaction Module][Speakers] Please, take the steering wheel!
[Interaction Module][Dashboard] Please, take the steering wheel!
[HiL Module][Preparatory Action] Request the driver to pay attention
[Interaction Module][Speakers] Please, look forward!
[Interaction Module][Dashboard] Please, look forward!
[HiL Module][Constraint] Constraint violation (5 seconds ellapsed)
[HiL Module][Fallback Plan] Executing Fallback Plan: 'Driving to a minimal risk condition'
```

Figure 16: Console of the Supervised Autonomous Driving for `TC04`

For the execution of the *𝒯1 Supervised Autonomous Driving* - `TC04` test case, the scenario began like the one described above. However in this case, the driver was asked to not react to the requests of the system. Therefore, the task precondition was unsatisfied because the human did not have her hands on the wheel and was not paying attention to the car. Consequently, since the task precondition was initially unsatisfied, the system asked the driver to take the wheel (*Request the human to take the wheel*) and to pay attention (*Request the human to pay attention*) by means of voice feedback on the speakers and by showing a message with an icon on the dashboard. Nonetheless, the driver remained inattentive (see Fig. 15). Consequently, the system notified the driver the current autonomous driving mode (*Notify the current autonomous driving mode*) by means of a text message on the dashboard. After 5 seconds (task constraint), the car executed the fallback plan. The event log that resulted from this test is shown in Fig. 16. Note that this result matches the one expected (see row four of Table 5).

### *8.4. Analyze the results*

The analysis of the results lies in checking whether the three design challenges are effectively pursued in the autonomous car tasks. Specifically, we checked whether the results for each identified question (Q1, Q2, and Q3) matched the expected results:

- **Q1**. The human was involved in performing the task by co-working with the system. If the collaboration was not achieved, a fallback plan was executed. Therefore, we can state that the *Complement 𝒜CPS functionality* design principle is fulfilled.
- **Q2**. The system performed feedback actions when they were specified. This means that the system copes with the *Achieve understandability* design principle based on what was designed. Therefore, we can state that the system implements mechanisms to help the user understand the system.
- **Q3**. Each interaction mechanism satisfied the initiative and attention level of the defined action. This implies that the system copes with the *Avoid obtrusiveness* design principle as it has been designed. Therefore, we can confirm that the system implements mechanisms to avoid annoying the user.

The aim of the evaluation performed is to find evidence of whether the solution obtained by applying our approach takes into account the design challenges as they were specified. The tests performed allowed us to validate that our proposal can be used to implement a functional *HiL-𝒜CPS* according to its collaborative work specification and that it behaves as specified in order to *complement the task functionality*, *achieve understandability*, and *avoid obtrusiveness.* As our methodological proposal suggests (see Fig. 8), rapid prototyping can be used to obtain feedback from users in order to refine the specification of the collaborative tasks and to achieve understandable and unobtrusive *HiL-𝒜CPS* systems even though they may be limited by the technology available in the solution domain.

This evaluation also provided us with knowledge that was used to improve both the design of the autonomous car and our approach. This knowledge includes:

- During the specification of the collaborative tasks, we detected several patterns in the specification of human-system interactions. Most of the collaborative task interactions were nearly the same, which implied getting very similar task and action models. Therefore, providing mechanisms that reuse the specification of common interactions may be of great interest.
- Checking the collaborative tasks allowed us to realize that $\mathcal{T}6$ *Takeover* required a feedback action once the system had transferred control to the human. This action was included in the final version of the action diagram.
- Implementing the collaborative tasks helped us to be aware that all of the implementations adopted the same execution model strategy. Therefore, we plan to develop an implementation framework in order to help developers build collaborative tasks based on this conceptual framework and strategy.

## 9. Conclusions

This work paves the way for Human Integration design in $\mathcal{A}$*CPS*s. We have studied how human participation within $\mathcal{A}$*CPS*s should be, and we have defined a conceptual framework that identifies the aspects that must be taken into consideration in order to design this human participation from an abstract and engineering point of view. This constitutes foundations of the design. Designing *HiL-*$\mathcal{A}$*CPS*s poses tremendous additional challenges. Experts from many disciplines must successfully solve these challenges.

As we have discussed in this paper, $\mathcal{A}$*CPS*s are powerful enough to take over control of what used to be controlled by people; however, they are not powerful enough to manage all of the misbehaviors that could negatively affect the fulfillment of system requirements and user expectations. Our conceptual framework is an attempt to address this issue by exploring the range of ways in which humans can participate and the roles they can play. This work provides a conceptual framework and a set of tools for introducing the user into the control loop of any $\mathcal{A}$*CPS* in order to integrate humans into the adaptation process. By designing the framework according to our design principles, the framework manages the trade-off between autonomy and human control and always takes into account understandability and user attention as key aspects. The application to the case of the autonomous car shows the feasibility of applying our solution in an existing $\mathcal{A}$*CPS*.

Our ongoing work is devoted to validate the usefulness of the framework with system designers and to provide methodological guidance and tools to apply the framework in existing methodologies and systems. Moreover, we plan to extend our approach to allow different human profiles to collaborate within the same task attending to different responsibilities. Future work will also provide more specific guidelines and design patterns for each type of user participation.

## Funding

This work was supported by MINECO under the project PROTEUS TIN2017-84094-R; and co-financed by the *Generalitat Valenciana* under the postdoctoral fellowship APOSTD/2016/042.

## Declarations of interest

None

## References

[1] D. S. Nunes, P. Zhang, J. S. Silva, A survey on human-in-the-loop applications towards an internet of all, IEEE Communications Surveys and Tutorials 17 (2) (2015) 944–965.

[2] P. Lalanda, J. A. McCann, A. Diaconescu, Autonomic Computing: Principles, Design and Implementation, Springer Publishing Company, Incorporated, 2013.

[3] Software Engineering for Self-Adaptive Systems: A second Research Roadmap, in: Software Engineering for Self-Adaptive Systems II, Vol. 7475 of Lecture Notes in Computer Science (LNCS), Springer, 2013, pp. 1–32.

[4] A. Moore, T. O'Reilly, P. D. Nielsen, K. Fall, Four thought leaders on where the industry is headed, IEEE Software 33 (1) (2016) 36–39. doi:10.1109/MS.2016.1.

[5] N. Jaynes, Timeline: The future of driverless cars, from audi to volvo (Aug 2016).
[6] Digital transformation of industries. automotive industry (Jan 2016).
[7] U. Farooq, J. Grudin, Human computer integration, ACM Interactions 23 (6) (2016) 26–32.
[8] J. Cámara, G. Moreno, D. Garlan, Reasoning about human participation in self-adaptive systems, in: 2015 IEEE/ACM 10th International Symposium on Software Engineering for Adaptive and Self-Managing Systems, 2015, pp. 146–156.
[9] L. Cranor, A framework for reasoning about the human in the loop., in: UPSEC'08 Conference on Usability, Psychology, and Security, 2008.
[10] W. Ju, L. Leifer, The design of implicit interactions: Making interactive systems less obnoxious, Design Issues 24 (3) (2008) 72–84.
[11] R. Parasuraman, T. B. Sheridan, C. D. Wickens, A model for types and levels of human interaction with automation, Trans. Sys. Man Cyber. Part A 30 (3) (2000) 286–297.
[12] D. Miller, W. Ju, Joint cognition in automated driving: Combining human and machine intelligence to address novel problems (2015).
[13] P. M. Fitts, Human engineering for an effective air-navigation and traffic-control system, National Research Council.
[14] J. C. Winter, D. Dodou, Why the fitts list has persisted throughout the history of function allocation, Cogn. Technol. Work 16 (1) (2014) 1–11.
[15] J. de Winter, P. Hancock, Reflections on the 1951 fitts list: Do humans believe now that machines surpass them?, Procedia Manufacturing 3 (2015) 5334 – 5341, 6th International Conference on Applied Human Factors and Ergonomics (AHFE 2015) and the Affiliated Conferences, AHFE 2015.
[16] SAE International, Surface Vehicle Recommended Practice. Taxonomy and Definitions for Terms Related to Driving Automation Systems for On-Road Motor Vehicles, Tech. rep., SAE International (2016).
[17] T. B. Sheridan, On how often the supervisor should sample, IEEE Transactions on Systems Science and Cybernetics 6 (2) (1970) 140–145.
[18] C. R. Frost, Challenges and opportunities for autonomous systems in space, in: National Academy of Engineering's U.S. Frontiers of Engineering Symposium, Armonk, New York, 2011.
[19] T. B. Sheridan, W. L. Verplank, Human and computer control of undersea teleoperators (Man-Machine Systems Laboratory Report).
[20] T. E. Trimble, R. Bishop, J. F. Morgan, M. Blanco, Human factors evaluation of level 2 and level 3 automated driving concepts: Past research, state of automation technology, and emerging system concepts, Tech. Rep. DOT HS 812 043, United States. National Highway Traffic Safety Administration (2014).
[21] ERTRAC, Automated driving roadmap, Tech. rep., ERTRAC Task Force "Connectivity and Automated Driving" (2015).
[22] M. Salehie, L. Tahvildari, Self-adaptive software: Landscape and research challenges, ACM Trans. Auton. Adapt. Syst. 4 (2) (2009) 14:1–14:42.
[23] B. Cheng, R. De Lemos, H. Giese, P. Inverardi, Software engineering for self-adaptive systems.
[24] X. Zhang, A user's perspective of design for context-awareness, in: Proceedings of the 13th International Conference on Ubiquitous Computing, UbiComp '11, ACM, New York, NY, USA, 2011, pp. 531–534.
[25] S. Munir, J. A. Stankovic, C.-J. M. Liang, S. Lin, Cyber physical system challenges for human-in-the-loop control, in: Presented as part of the 8th International Workshop on Feedback Computing, USENIX, San Jose, CA, 2013.
[26] C. Shin, A. K. Dey, W. Woo, Mixed-initiative conflict resolution for context-aware applications, in: Proceedings of the 10th International Conference on Ubiquitous Computing, UbiComp '08, ACM, New York, NY, USA, 2008, pp. 262–271.
[27] W. D. Nothwang, M. J. McCourt, R. M. Robinson, S. A. Burden, J. W. Curtis, The human should be part of the control loop?, in: Resilience Week (RWS), 2016, 2016.
[28] C. Dorn, R. N. Taylor, Coupling software architecture and human architecture for collaboration-aware system adaptation, in: Proceedings of the 2013 International Conference on Software Engineering, ICSE '13, IEEE Press, Piscataway, NJ, USA, 2013, pp. 53–62.
[29] C. Evers, R. Kniewel, K. Geihs, L. Schmidt, The user in the loop: Enabling user participation for self-adaptive applications, Future Generation Computer Systems 34 (0) (2014) 110–123.
[30] A. G. Mirnig, M. Gärtner, A. Laminger, A. Meschtscherjakov, S. Trösterer, M. Tscheligi, R. McCall, F. McGee, Control transition interfaces in semiautonomous vehicles: A categorization framework and literature analysis, in: International Conference on Automotive User Interfaces and Interactive Vehicular Applications (AutomotiveUI '17), 2016.
[31] B. E. Noah, T. M. Gable, S.-Y. Chen, S. Singh, B. N. Walker, Development and preliminary evaluation of reliability displays for automated lane keeping, in: International Conference on Automotive User Interfaces and Interactive Vehicular Applications (AutomotiveUI), 2017.
[32] A. Kunze, S. J. Summerskill, Enhancing driving safety and user experience through unobtrusive and function-specific feedback, in: International Conference on Automotive User Interfaces and Interactive Vehicular Applications, 2017.
[33] G. Schirner, D. Erdogmus, K. Chowdhury, T. Padir, The future of human-in-the-loop cyber-physical systems, Computer 46 (1) (2013) 36–45.
[34] E. Lloyd, S. Huang, E. Tognoli, Improving human-in-the-loop adaptive systems using brain-computer interaction, in: 2017 IEEE/ACM 12th International Symposium on Software Engineering for Adaptive and Self-Managing Systems (SEAMS), 2017, pp. 163–174.
[35] S. Huang, P. Miranda, Incorporating human intention into self-adaptive systems, in: 2015 IEEE/ACM 37th IEEE International Conference on Software Engineering, Vol. 2, 2015, pp. 571–574.
[36] J. D. Lee, K. A. See, Trust in automation: Designing for appropriate reliance, Human Factors 46 (50-80).
[37] N. A. Stanton, M. S. Young, Vehicle automation and driving performance, Ergonomics 41 (7) (2010) 1014–1028.
[38] S. Nahavandi, Trusted autonomy between humans and robots:toward human-on-the-loop in robotics and autonomous systems, IEEE Systems, Man, and Cybernetics Magazine 3 (1) (2017) 10–17.
[39] B. Y. Lim, A. K. Dey, Evaluating intelligibility usage and usefulness in a context-aware application, in: HCI 2013: Human-Computer Interaction. Towards Intelligent and Implicit Interaction, 2013, pp. 92–101.
[40] S. Bouzit, G. Calvary, J. Coutaz, D. Chêne, E. Petit, J. Vanderdonckt, Design space exploration of adaptive user interfaces, in: IEEE 11th Int. Conference on Research Challenges in Information Science RCIS, 2017.
[41] A. Jameson, User modeling meets usability goals, in: User Modeling 2005, 2005.
[42] D. M. Russell, P. Maglio, R. Dordick, C. Neti, Dealing with ghosts: Managing the user experience of autonomic computing, IBM Sys. Journal 42 (1) (2003) 177–188.
[43] S. Stumpf, W.-K. Wong, M. Burnett, V. Pipek, End-user interactions with intelligent and autonomous systems, in: CHI'12, 2012.
[44] T. Litman, Autonomous Vehicle Implementation Predictions: Implications for Transport Planning (2013).

[45] J. Muller, No hands, no feet: My unnerving ride in google's driverless car, Forbes.
[46] S. Row, The future of transportation: Connected vehicles to driverless vehicles...what does it mean to me?, ITE Journal 83 (10) (2013) 24–25.
[47] National Transportation Safety Board, Collision Between a Car Operating With Automated Vehicle Control Systems and a Tractor-Semitrailer Truck Near Williston (HAR1702), Tech. rep., NTSB (May 2016).
[48] National Transportation Safety Board, Preliminary Report Released for Crash Involving Pedestrian, Uber Technologies, Inc., Test Vehicle (HWY18MH010), Tech. rep., NTSB (March 2018).
[49] A. Larsson, Issues in Reclaiming Control from Advanced Driver Assistance Systems, in: European Conference on Human Centred Design for Intelligent Transport Systems, Vol. 2, HUMANIST publications, 2010, pp. 557–564.
[50] N. Gkikas, Automotive Ergonomics, Driver-Vehicle Interaction, CRC Press, 2012.
[51] H. Dubberly, P. Pangaro, U. Haque, On modeling: What is interaction?: Are there different types?, interactions 16 (1) (2009) 69–75.
[52] D. Eskins, W. H. Sanders, The Multiple-Asymmetric-Utility System Model: A Framework for Modeling Cyber-Human Systems, in: Proceedings of the 2011 Eighth International Conference on Quantitative Evaluation of SysTems, IEEE Computer Society, Washington, DC, USA, 2011, pp. 233–242.
[53] A. K. Dey, Understanding and using context, Personal Ubiquitous Comput. 5 (1) (2001) 4–7.
[54] E. Horvitz, C. Kadie, T. Paek, D. Hovel, Models of attention in computing and communication: from principles to applications, Commun. ACM 46 (3) (2003) 52–59.
[55] B. Buxton, Integrating the periphery and context: A new model of telematics, in: Proceedings of Graphics Interface, 1995, pp. 239–246.
[56] M. Gil, V. Pelechano, Self-adaptive unobtrusive interactions of mobile computing systems, JAISE 9 (6) (2017) 659–688.
[57] M. E. Khan, Different approaches to black box testing technique for finding errors, International Journal of Software Engineering & Applications (IJSEA) 2 (4).