



**HoliDes**  
**Holistic Human Factors Design of**  
**Adaptive Cooperative Human-Machine**  
**Systems**

**HoliDes**

**D 3.7a - Techniques and Tools for Adaptation Vs2.0 incl. Handbooks and Requirements Analysis Update**

<b>Project Number:</b>	332933
<b>Classification:</b>	Public
<b>Work Package(s):</b>	WP 3, T3.1, T3.3, T3.4, T3.5, T3.6
<b>Milestone:</b>	M2
<b>Document Version:</b>	V 1.0
<b>Issue Date:</b>	31.09.2016
<b>Document Timescale:</b>	Project Start Date: October 1, 2013
Start of the Document:	Month 25
Final version due:	Month 36
<b>Deliverable Overview:</b>	<b>Confidential Document:</b> D 3.7b - Techniques and Tools for Adaptation Vs2.0 incl. Handbooks and Requirements Analysis Update <b>Public document:</b> D 3.7a - Techniques and Tools for Adaptation Vs2.0 incl. Handbooks and Requirements Analysis Update, Public part excluding confidential information + <b>Annex I:</b> Communication Guidelines (Public) <b>Annex II:</b> Handbooks (Confidential) <b>Annex III:</b> HF Guidelines (Public) <b>Annex IV:</b> Requirements analysis update (Confidential)
<b>Compiled by:</b>	Richard Leblond (EAD-FR)
<b>Authors:</b>	Sara Sillaurren (TEC), Elisa Landini (REL), Roberta Presta (SVN), Zdenek Moravek (HON), Mark Eilers (OFF), Denis Javaux (SYM), Richard Leblond, Nicolas Schneider, Helge Ludwig (TWT), Marco Botta (UTO), Mark Eilers (OFF), Stefan Griesche (DLR), T. Bellet, J.C. Bornard, B. Richard, D. Gruyer (IFS), Oliver Klaproth (EAD-DE), Andreas Riegger (EAD-FR),
<b>Reviewers:</b>	Frank Jonat (EAD-CAS-DE), Ignacio Gonzalez Fernandez(ATO)
<b>Technical Approval:</b>	Jens Gärtner, Airbus Group Innovations
<b>Issue Authorisation:</b>	Sebastian Feuerstack, OFF

© All rights reserved by HoliDes consortium

This document is supplied by the specific HoliDes work package quoted above on the express condition that it is treated as confidential to those specifically mentioned on the distribution list. No use may be made thereof other than expressly authorised by the HoliDes Project Board.



**HoliDes**  
**H**olistic Human Factors **D**esign of  
Adaptive Cooperative Human-  
Machine Systems



**DISTRIBUTION LIST**

Copy type <sup>1</sup>	Company and Location	Recipient
T	HoliDes Consortium	all HoliDes Partners

<sup>1</sup> Copy types: E=Email, C=Controlled copy (paper), D=electronic copy on Disk or other medium, T=Team site (AjaXplorer)



## HoliDes

**H**olistic Human Factors **D**esign of  
Adaptive Cooperative Human-  
Machine Systems

### RECORD OF REVISION

Date	Status Description	Author
13.06.2016	Initial Version	Richard Leblond (EAD-FR)
07.07.2016	Contribution to chapter 3.1	Sara Sillaurren (TEC), Elisa Landini (REL), Roberta Presta (SVN), Zdenek Moravek (HON),
07.07.2016	Annex I	Simona Collina – SVN, Roberta Presta –SVN, Zdenek Moravek HON, Martina Boecker & Frank Jonat - EAD-GE, Flavia De Simone – SNV, Elisa Landini – REL, Gert Weller - TAK, Sara Sillaurren - TEC
15.07.2016	Contribution to chapter 3.3	Mark Eilers (OFF),
25.07.2016	Contribution to chapter 2	Denis Javaux, (SYM-EAD-FR)
18.08.2016	Contribution to chapter 3.3.1	Helge Ludwig (TWT)
21.08.2016	Contribution to chapter 3.3.1	Stefan Griesche (DLR)
22.08.2016	Annex II	Stefan Griesche (DLR)
22.08.2016	Contribution to chapter 3.3.3	T. Bellet, J.C. Bornard, B. Richard, D. Gruyer (IFS),
28.08.2016	Chapter 4 + Annex III	Oliver Klaproth (EAD-DE)
29.08.2016	Contribution to chapter 3.2	Andreas Riegger (EAD-FR)
09.09.2016	Introduction conclusion and Contribution to Chapter 2	Nicolas Schneider, Richard Leblond (EAD-FR)
26.09.2016	Final version of public document	Richard Leblond (EAD_FR)



**Table of Contents**

- 1 Introduction .....10**
- 2 From Adaptation Framework to Derivation of HF Requirements for AdCoS Builder .....13**
  - 2.1 AdCoS modelling ..... 13
    - 2.1.1 Modelling AdCoS with control loops..... 14
    - 2.1.2 Modelling agents allocation to loops ..... 16
    - 2.1.3 Primitives for AdCoS modelling..... 19
    - 2.1.4 Building an AdCoS model..... 22
    - 2.1.5 How to practically build AdCoS models ..... 27
  - 2.2 Automatic derivation of HF requirements from an AdCoS model ..... 38
    - 2.2.1 Families of HF requirements..... 38
    - 2.2.2 Associating HF requirements with individual AdCoS primitives... 39
    - 2.2.3 Derivation of additional HF requirements when the allocation of the agents to the steps is known ..... 42
    - 2.2.4 Implementation of AP in the Platform Builder ..... 45
- 3 Resolution process .....46**
  - 3.1 UC2, Diversion Assistant..... 46
    - 3.1.1 Data flow enabled by HF-RTP tool chain ..... 47
    - 3.1.2 Inputs & outputs ..... 48
    - 3.1.3 HF-RTP tools applied to realize adaptation ..... 49
    - 3.1.4 Integration ..... 50
    - 3.1.5 Results of Proofs of concept ..... 51
    - 3.1.6 Discussion & Perspectives..... 51
  - 3.2 UC3, Command and Control Room..... 52
    - 3.2.1 Data flow full treatment chain ..... 52
    - 3.2.2 Inputs & outputs ..... 54
    - 3.2.3 Tools used ..... 54
    - 3.2.4 Integration ..... 56
    - 3.2.5 Results of Proof of concept ..... 56
    - 3.2.6 Discussion & Perspectives..... 60
  - 3.3 UC4, Overtaking including lane change assistant ..... 61
    - 3.3.1 AdCoS based on Cognitive Distraction Classifier and CONFORM . 61
    - 3.3.2 AdCoS based on MOVIDA ..... 73
    - 3.3.3 AdCoS based on Adapted Assistance ..... 92



**4 Holistic Human Factors Design Guidelines ..... 109**

- 4.1 Introduction ..... 109
  - 4.1.1 Objective..... 109
  - 4.1.2 Holistic Design ..... 109
- 4.2 Requirements analysis ..... 110
  - 4.2.1 Guideline Design ..... 110
  - 4.2.2 Target Group Interviews..... 110
  - 4.2.3 Guideline Requirements..... 111
- 4.3 Adaptive Components in AdCoS..... 111
  - 4.3.1 Adaptability, Adaptivity and Adaptation ..... 111
  - 4.3.2 Adaptive systems ..... 112
  - 4.3.3 Level of control ..... 112
  - 4.3.4 Automation..... 113
  - 4.3.5 Classification of adaptive components ..... 114
- 4.4 ‘Five Ws and one H’ ..... 116
  - 4.4.1 Method..... 116
  - 4.4.2 Five Ws and one H for Adaptive Components ..... 117
- 4.5 Guidelines ..... 121

**5 Requirements update ..... 121**

**6 Conclusion ..... 126**

**7 References..... 126**



**HoliDes**  
**H**olistic Human Factors **D**esign of  
Adaptive Cooperative Human-  
Machine Systems



## Table of Tables

Table 1: Integration status of tools associated with adaptation in Diversion Assistant use-case ..... 50  
Table 2: Validation status of tools associated with adaptation in Diversion Assistant use-case ..... 51  
Table 3: Log data structure ..... 54  
Table 4: KNIME Framework execution input parameter ..... 54  
Table 5: Extract from hidden patterns ..... 57  
Table 6: CONFORM RTMaps inputs ..... 63  
Table 7: Comparison of normed “Best-Worst-Score” for the baseline, the AdCoS and the actual participant rating  
in different situation [D9.9] ..... 71  
Table 8: Input data for the inference-engine component of the DIR module..... 97  
Table 9: Additional input for the DIR model, required for data pre-processing when utilized on the CRF  
demonstrator vehicle.....100



# Table of Figures

Figure 1: Control loop..... 14

Figure 2: Adaptive loop on an executive loop..... 14

Figure 3: Adaptive and executive loops in an AdCoS ..... 15

Figure 4: Tree-like structure of adaptive and executive loops ..... 16

Figure 5: Manual (H) and Automatic (M) loop execution ..... 17

Figure 6: Mixed execution of the loop ..... 17

Figure 7: Allocation of the agents to the execution of the steps ..... 18

Figure 8: A control loop that control a process ..... 20

Figure 9: A control loop used by an agent to control a process. .... 20

Figure 10: A control loop used by an agent to control a process through an user interface ..... 20

Figure 11: An example of an AdCoS model with primitive control loops ..... 21

Figure 12: Finite set of primitive control loops (M: machine, H: human, T: task, TD, task distribution, P: process, UI: user interface) ..... 21

Figure 13: Border control room AdCoS modelled with the primitives. On left the primitive graph structure and on right the same structure with variable instantiation. .... 22

Figure 14: A single machine agent M performing a loop..... 24

Figure 15: The agent M is in fact performing all the steps ..... 24

Figure 16: A human agent H and machine agent M perform the loop ..... 24

Figure 17: The human agent H and machine agent M are involved in different steps ..... 25

Figure 18: Steps may involve more than one agent..... 25

Figure 19: More than two agents can be involved in a loop ..... 26

Figure 20: Two fully detailed executive and adaptive loops ..... 27

Figure 21: Detailed description of an AdCoS (a) and ..... 28

Figure 22: Executive loops in example AdCoS model ..... 29

Figure 23: Extract of Excel file to support the elicitation of the content to integrate in loop models..... 31

Figure 24: AdCoS model with allocation of the agents to the executive loops (loop-based & step-based descriptions) ..... 32

Figure 25: AdCoS model with adapted parameters and adaptive loops for the two executive loops (loop-based & step-based descriptions) ..... 35

Figure 26: AdCos model with agents determined for both executive and adaptive loops (loop-based and step-based descriptions) ..... 36

Figure 27: framework for adaptation in the in the platform builder application ..... 46

Figure 28: Data flow supporting the adaption strategies for Diversion Assistant. Both strategies assume the same steps, though algorithms and tools applied to accomplish the steps differ, see colour highlight. .... 47

Figure 29: Tools enabling the transfer of raw input data, e.g. camera or EEG, to means for triggering adaption.. 48

Figure 30: Border Control Room ..... 52

Figure 31: KNIME Framework..... 53

Figure 32: Part of a KNIME workflow ..... 53

Figure 33: KNIME Analytics Platform ..... 55

Figure 34: Overview of different KNIME nodes ..... 55

Figure 35: User interface to set execution parameters ..... 56

Figure 36: User interface to display result ..... 56



## HoliDes

Holistic Human Factors Design of Adaptive Cooperative Human-Machine Systems



Figure 37: Number of absence on Fridays.....	57
Figure 38: Number of absence on Tuesdays .....	58
Figure 39: Result of KNIME Framework correlation pattern .....	59
Figure 40: Mean number of absence per day .....	60
Figure 41: Mean number of absence per hour.....	60
Figure 42: RTMaps CONFORM inputs .....	63
Figure 43: CONFORM GUI and choice of input source .....	64
Figure 44: CONFORM input choices in the offline mode. The available inputs depend on the considered csv file.	65
Figure 45: Structure of the CONFORM Model.....	67
Figure 46: Two screenshots of the CONFORM GUI .....	68
Figure 47: Integration of the MTTs CDC and CONFORM in the Adapted Automation AdCoS [D9.9] .....	69
Figure 48: Integration of CONFORM in the RTMaps framework for the Adapted Automation AdCoS .....	70
Figure 49: Functional architecture of the AdCoS based on MOVIDA.....	73
Figure 50: Driving Scenarios and Use cases for the MOVIDA-AdCoS.....	74
Figure 51: pictogram delivered by MOVIDA (on the in-vehicle display) to inform a non-distracted driver that the Lane Change Manoeuvre is possible.....	76
Figure 52: pictogram delivered by MOVIDA to warn a distracted driver that a Lane Change Manoeuvre is required and possible.....	76
Figure 53: pictogram delivered by MOVIDA to warn the driver that that a Lane Change Manoeuvre is not possible .....	78
Figure 54: Pictogram used by the Collision Warning System of MOVIDA.....	78
Figure 55: pictogram delivered by MOVIDA to inform the driver about the automatic “Emergency Braking”, when implemented by the AdCoS.....	79
Figure 56: pictogram delivered by MOVIDA to inform the driver about the “Lane Keeping” automatic manoeuvre, when implemented by the AdCoS .....	79
Figure 57: pictogram use by MOVIDA to inform the driver about the “Full Automation” status of the AdCoS (i.e. automatic Lane Keeping and Braking) .....	79
Figure 58: Overview of the V-HCD platform, as an example of a tailored HF-RTP based on RTMaps for automotive application .....	80
Figure 59: RTMaps diagram for MOVIDA-AdCoS tests with COSMODRIVE .....	81
Figure 60: simulation of visual distraction effects with COSMODRIVE.....	82
Figure 61: Visualization of MOVIDA outputs for a well-managed situation by a non-distracted driver (as simulated with COSMODRIVE).....	84
Figure 62: Visualization of MOVIDA outputs delivered to support the Lane Change manoeuvre of a visually distracted driver.....	85
Figure 63: Visualization of MOVIDA outputs to warn the driver of a dangerous Lane Change manoeuvre .....	86
Figure 64: Visualization of MOVIDA outputs when the Automatic Lane Keeping function is activated .....	87
Figure 65: MOVIDA outputs to warn the driver of frontal collision risk .....	88
Figure 66: Visualization of MOVIDA outputs when the Automatic Braking is implemented by the AdCoS .....	89
Figure 67: Visualization of MOVIDA outputs when the Automatic Lane Keeping and Braking functions are jointly implemented .....	90
Figure 68: Virtual design process of MOVIDA-AdCoS with the V-HCD platform .....	91
Figure 69: Architecture of the Adapted Assistance AdCoS .....	93
Figure 70 : Schematic overview of the DIR module. ....	96
Figure 71: Classification of potential alter-vehicles (dark) in the vicinity of the ego-vehicle (light) in relation of to the position of the ego-vehicle when driving on the slow lane. ....	98



Figure 72: Classification of potential alter-vehicles (dark) in the vicinity of the ego-vehicle (light) in relation of to the position of the ego-vehicle when driving on the fast lane..... 99

Figure 73: Overview of the tool-chain used for the DIR module. ....101

Figure 74: (Simplified) overview of the DIR module integrated in RTMaps, arranged to highlights RTMaps components for the Data Pre-Processing and Inference Engine of the DIR module.....102

Figure 75: Example of driver intention recognition module in RTMaps. In this example, the intention to perform a lane change to the left (bottom, orange line) is recognized approx. 1 sec prior to the activation of the indicator (bottom, red line).....103

Figure 76: Learned graph-structure representing  $pIt, Bt, Pt|It - 1, Bt - 1Lt = \text{slow\_lane}$ . The additional parent  $Lt = \text{slow\_lane}$  and variables not conditioned by  $It$  or  $Bt$  are omitted to improve visibility.....107

Figure 77: Learned graph-structure factorizing  $pIt, Bt, Pt|It - 1, Bt - 1Lt = \text{fast\_lane}$ . The additional parent  $Lt = \text{fast\_lane}$  and variables not conditioned by  $It$  or  $Bt$  are omitted to improve visibility. ....108

Figure 78: Framework for adaptive human-machine-systems [3].....112

Figure 79: Framework for Automation Design [6] .....114

Figure 80: Classification of Adaptive Components.....115

## 1 Introduction

The objective of this document “D 3.7b - Techniques and Tools for Adaptation Vs 2.0 incl. Handbooks and Requirements Analysis Update”, confidential, is to give a final update on the use of the Adaptation Framework proposed and improved along the project and the last status about the development of the techniques and tools concerning Adaptation, and their integration in the different AdCoS.

The public document “D3.7a - Techniques and Tools for Adaptation Vs 2.0 incl. Handbooks and Requirements Analysis Update” will be an extract of the confidential one and will be available on HoliDes official site.

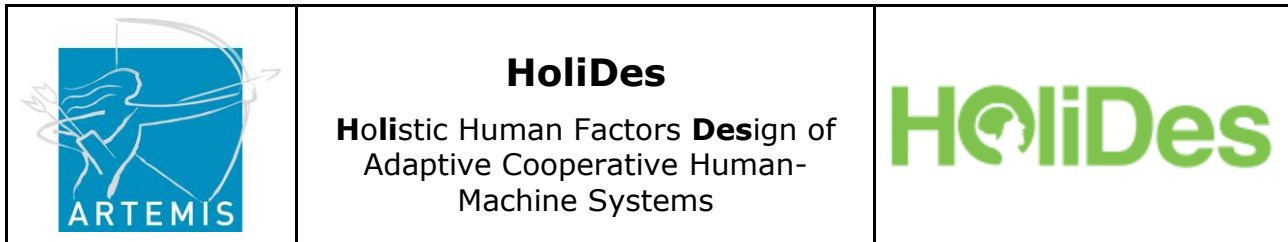
The ANNEX I concerning an update of the Communication Guidelines is added to the public document. The ANNEX II concerning the Handbooks is added to the confidential document. The ANNEX III concerning the Human Factor Guidelines is added to the public document. The ANNEX IV concerning the Requirements analysis and update is added to the confidential document.

In the confidential document, the section 2 presents and improvement of the technique to derive Human Factors (HF) requirements for AdCoS on the basis of the Framework for Adaptation proposed in previous versions of this document. The technique can easily be implemented into software, for automating such derivations as soon as an AdCoS model is available, and is as such of interest in particular for AdCoS designers. A limited version of the concept has been implemented as software module of the platform builder and is now accessible in HoliDes HF-RTP platform.

The section 3 presents the evolution during last months and the last status of the different AdCoS in terms of integration of adaptation functions.

In the Aeronautical domain, Pilot Pattern Classifier (PPC), a machine learning tool able to leverage the provided dataset to the aim of the online detection of the workload level has been experienced. The first results cover individual dedicated tests of tools connected to the Diversion assistant. It appears that PPC failed to provide a generic classification tool for any user but performs well as a user-specific classifier. Further step is to verify the performance of PPC on aeronautics tasks.

In the Control Room domain, The AdCoS uses the MTT KNIME framework to observe the behaviour of operators regarding absences in border control room. The implementation in a test environment was quite meaningful and proofed that the MTT provides the expected results. In terms of performance the underlying KNIME execution engine in the non-commercial version seems to be limited. A switch to a commercial product version could mitigate this drawback.



In the Automotive domain, three AdCoS were experienced:

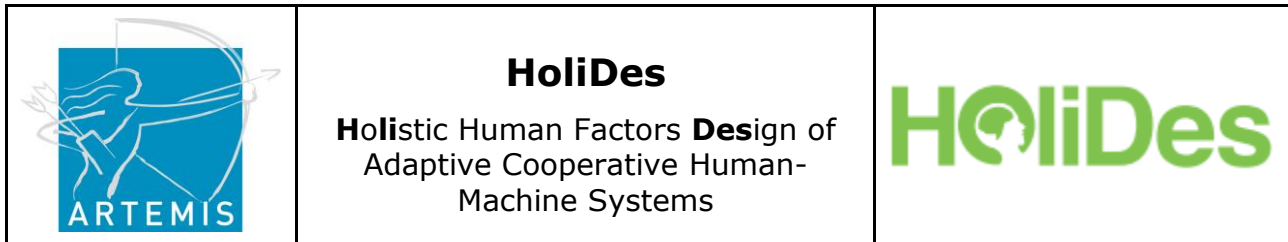
The first is AdCoS based on one hand on the Cognitive Distraction Classifier (CDC) which is a software tool to detect cognitive driver distraction from video images of the driver's face and vehicle kinematic and control data and on the other hand on the tool CONFORM. The use of audio data as well as eye-tracking data has also been investigated and showed that it may contribute to a higher accuracy of cognitive distraction detection. The CDC and CONFORM have been integrated with the IAS autonomous driving system in the IAS Test Vehicle. Additionally the suitability of the CDC in the aviation domain has been investigated. The data analysis is not yet finished.

For CDC, on facial video and behavioural data, low prediction results on online analysis, based on Naives Bayes, were investigated but other classification algorithms show better results. Extending the CDC with eye-tracking and audio data may significantly improve results. On the other hand and highly significantly detection of cognitive distraction offline during a driving task, using facial video data, has been shown. The driver's preferred driving style of the automated vehicle was predicted by CONFORM and consequently adapted, with a noticed quality of prediction. The AdCoS increased the appealing of the automation behaviour compared to the none-adaptive baseline, and some values can be interpreted as a clear benefit.

The second AdCoS based on MOVIDA functions (Monitoring of Visual Distraction and risks Assessment) is an integrative co-piloting system supervising several simulated Advanced Driving Aid Systems (ADAS). All the MTTs required for the MOVIDA-AdCoS and its design process with the Virtual Human Centred Design platform (V-HCD) platform were integrated from RTMaps software. V-HCD platform was particularly used to support MOVIDA AdCoS design processes.

At the earliest stages of the design process, COSMODRIVE-based simulations were used to help identifying the critical driving scenarios due to visual distraction for which an AdCoS based on MOVIDA could support them. Through these simulations, it has been possible to provide ergonomics specifications of human driver needs. During the virtual design process of the AdCoS, simulations of MOVIDA-based assistance according to situational risk and the drivers' visual distraction status were implemented in order to progressively design, evaluate and thus increase the MOVIDA-AdCoS *efficiency* for the different critical scenarios and use cases of reference previously identified.

The third AdCoS Adapted Assistance is a Lane-Change Assistant (LCA) system, able to adapt to the internal and external scenarios. The "optimal" manoeuvre is



suggested by means of specific warnings, advice and information, according to the visual state and intentions of driver, as well as to the external environment. It involves a Driver Distraction Classifier (DDC) and a Driver Intention Recognition (DIR) that are currently integrated together through RTMaps.

The AdCoS as a whole and the HMI have been under evaluation during the third year of the project. In particular, separate studies in the REL simulation environment have been performed for the Adapted Assistance AdCoS and for the HMI. Also, other evaluation studies have been performed about the communication strategies proposed in D 3.7a Annex I. Experimental analysis has been applied to evaluate the benefit of having the communication of why performed by means of the haptic channel, even in this case from both a subjective and an objective point of views. The results of the AdCoS evaluation compared with the baseline showed that the adaptation had a great benefit on the performance indicators, both for technical assessment and user related assessment.

For the HMI evaluation, results showed that the solutions with or without haptic channel do not have significant differences, indicating that, even if the why haptic warning represents a cooperation mode the subjects are not used to, it is judged acceptable as other more familiar warning alarms.

For the DIR module, CRF performed a free-driving study with the CRF demonstrator vehicle. Based on previous versions of the DIR module developed for simulation environments, a generative modelling approach has been used. As the quality of the training data was not sufficient to reliably learn models for predicting the control-behaviour for lateral and longitudinal control, and as such output was not planned to be used within the AdCoS "Adapted Assistance", the focus was made on the intention and behaviour recognition aspects and the control inputs of the driver, as additional input features was provided.

In Section 4, the human factors needs have been replaced in an holistic design process and proposed main HF guidelines that can be found in Annex III.

**The section 5 presents the status of requirement update process.**

## 2 From Adaptation Framework to Derivation of HF Requirements for AdCoS Builder

In this section we will describe a technique to derive Human Factors (HF) requirements for AdCoS.

The technique first implies to model the target AdCoS in a formalism we describe in section 2.1.

The model is then used in section 2.2 to derive the HF requirements associated with the AdCoS.

We then demonstrate in section 2.2.4 that the technique can easily be implemented into software, for automating such derivations as soon as an AdCoS model is available.

The technique and the associated software are of interest for HoliDes, and in particular AdCoS designers. By allowing the automatic derivation of HF requirements it opens the door to easy AdCoS prototyping and modification (e.g., an AdCoS can be tested with the technique at an early design stage and if it leads to HF requirements that are known to be difficult to satisfy in the current project, the AdCoS design can be modified accordingly).

The technique should be integrated into the HoliDes HF-RTP.

### 2.1 AdCoS modelling

The HF requirements derivation technique relies on a first step that consists in modelling the target AdCoS in a peculiar formalism, based on a series of elementary primitives that are then combined into a complete AdCoS model.

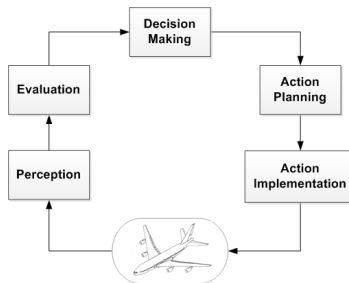
The formalism is based on the notion of control loops. In D3.3 - Framework for Adaptation and in D3.4 - Techniques and Tools for Adaptation, we have already shown how AdCoS could be modelled in terms of such loops. The notations used in these deliverables were mostly graphical and could not be used for formally modelling AdCoS. We are now here proposing a simplification and formalization of these notations which will ultimately be usable for automatic derivation of HF requirements from AdCoS models.



### 2.1.1 Modelling AdCoS with control loops

#### 2.1.1.1 Control loop

A control loop involves a series of operations that are performed in sequence to control some object.



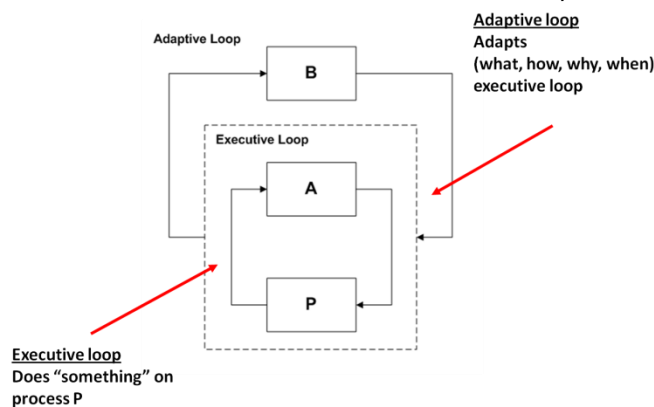
**Figure 1: Control loop**

In the control loop of Figure 1, an aircraft (process) is controlled through the steps of perception, evaluation, decision-making, action planning and action implementation.

The steps can be performed by a human or a machine agent or by combination of one or more of them (cooperative system).

#### 2.1.1.2 Control loops and AdCoS

In D3.4 - Techniques and Tools for Adaptation, we have shown how control loops can be used to capture the essential feature of an AdCoS, which is adaptiveness.

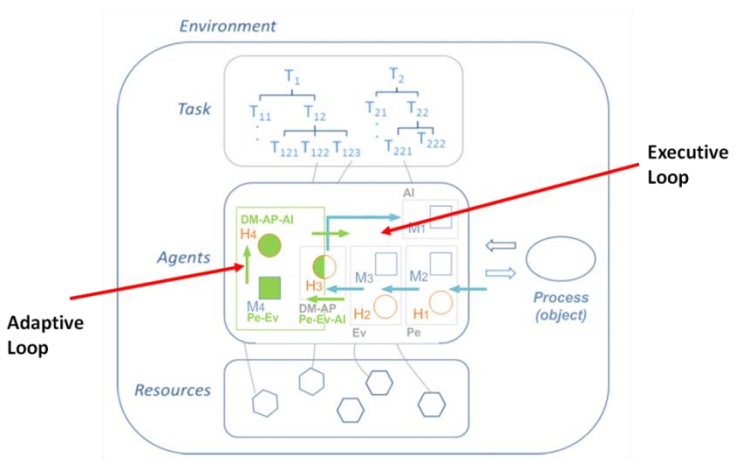


**Figure 2: Adaptive loop on an executive loop**



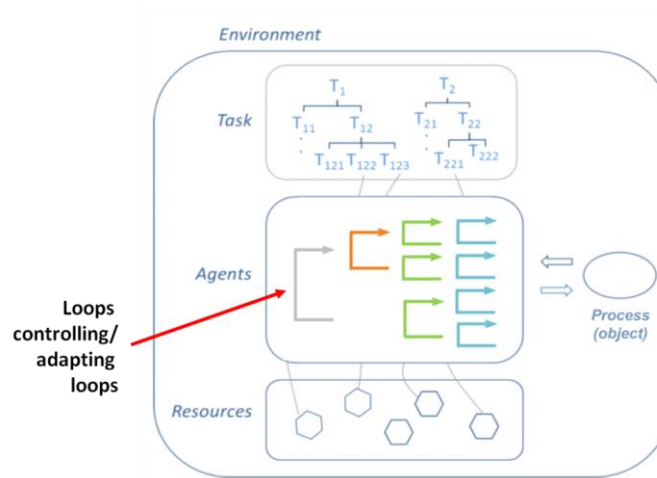
In Figure 2, a basic loop is closed by an agent A (e.g., autopilot) on a process P (e.g. aircraft). That loop, called the executive loop because it is executed on the final target process, can be described through a series of parameters (e.g., the target the loop is trying to achieve). A second control loop, called the adaptive loop, is then closed by an agent B on one or more of these parameters. These executive loop parameters are therefore under control of the adaptive loop. This allows "adapting" the value of these parameters to various circumstances. For example, in an aircraft, the Flight Management System (FMS) (agent B) may adapt the target speed the autopilot (agent A) has to achieve, based on the progress over the successive segments of the flight plan.

The Figure 3 shows that such adaptive and executive control loops are common in an AdCoS and are a convenient way of modelling them.



**Figure 3: Adaptive and executive loops in an AdCoS**

The Figure 3 shows a single adaptive loop and a single executive loop in the AdCoS. This corresponds to a rather simple AdCoS. In general though, there are more than two control loops in an AdCoS. There may be several executive loops, acting on different systems or parameters. And there may be several adaptive loops acting on them. There may even be adaptive loops acting on other adaptive loops. The Figure 4 shows for example a series of such loops acting on others, in a tree-like structure.



**Figure 4: Tree-like structure of adaptive and executive loops**

That tree-like structure is simple and rather organized. In practice, the control relations between loops take the general shape of a graph, whose nodes depict control loops and edges depict control relations between them.

### 2.1.2 Modelling agents allocation to loops

In the sections above, we have shown how AdCoS can be modelled via executive and adaptive loops, typically through hierarchic structures (see Figure 4) that provide several incremental layers of adaptation to the executive loops a CoS applies on some external process (e.g., human driver and assistant systems drive the car. How the assistant systems behave and what they do may be dynamically adapted to the state of the driver or the state of the weather). CoS (or COoperative human-machine System), initially presented in Figure 4 of D3.4a has been described more in detail in section 2.1.2.3 of D3.5a.

Little was said though on how human and machine agents act together to complete executive and adaptive loops.

One or more agents, human or machine, are always needed to execute a control loop. As shown in Figure 1, control loops can be modelled in term of steps (perception, evaluation, decision making, acting planning, action implementation) and something obviously has to perform the tasks or actions associated with the steps. The agents involved in the loop specifically do that.

In this deliverable, we will consider two types of agent allocation to loops:

- agent allocation at the loop level

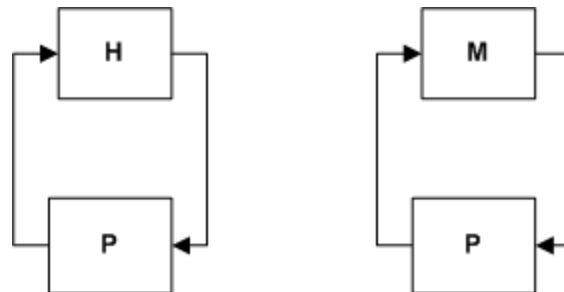




- agent allocation at the step level

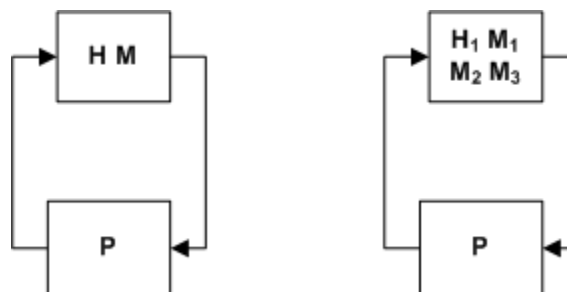
### 2.1.2.1 Agent allocation modelled at the loop level

There can be loops that are completely achieved by a single agent, the agent being in charge of all the steps. When the agent is a human, the loop is said to be manually executed. When the agent is a machine or a system, the loop is automatically executed. The Figure 5, using the type of formalism of Figure 2, shows the two cases



**Figure 5: Manual (H) and Automatic (M) loop execution**

Beyond these simple cases, we meet situations where human and machine agents contribute together to the execution of a loop. These are mixed - or cooperative - execution of the loop. The Figure 6 shows a loop executed in cooperation by a human agent H and a machine agent M. It also shows a more complex case where a single human agent H<sub>1</sub> and three machine agents, M<sub>1</sub>, M<sub>2</sub> and M<sub>3</sub>, execute the loop. This would for example be the case in the automotive UC, when the car is driven by a cooperative system composed of the driver and several assistant systems.



**Figure 6: Mixed execution of the loop**

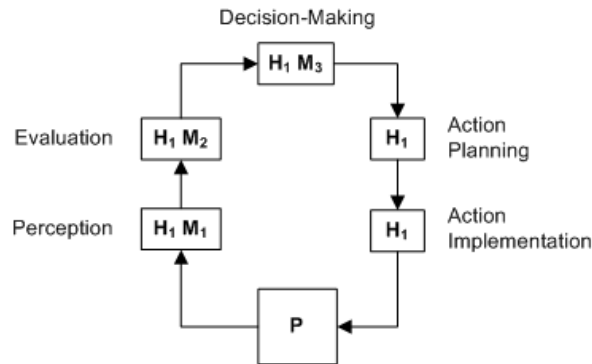
In this kind of description of the allocation of agents to the loop, we only know and care about which agents are involved in a loop, but not specifically about what they do (e.g. to which step of the loop they contribute and what they do).



### 2.1.2.2 Agent allocation modelled at the step level

Modelling of agent allocation in loops can go further when the loop model describes in which steps the agents, human or machine, work, possibly including describing what they specifically do in the step.

Typical assistant systems for example are very specialized and only work in specific loop steps to assist a human agent in charge of the overall loop. The human agent can then be assisted by one or more agents, on one or more of the five steps typical of the control loops.



**Figure 7: Allocation of the agents to the execution of the steps**

The Figure 7 shows how the more complex case in Figure 6 can be further refined by allocating the human agent H1 and three machine agents M1, M2 and M3 to specific steps. The figure indicates that the Perception, Evaluation and Decision-Making steps are performed jointly by the human and the three machine agents, typically in an assistance paradigm: the human is in charge and executes all steps but is assisted by dedicated machine agents for the three first steps. This could again correspond to an automotive UC with enhanced perception of the environment (e.g., blind spot monitor), assistance to evaluating the current situation (e.g., lane departure warning system, forward collision warning) and to decision-making (e.g., lane change assistance).

Thus a more general and powerful modelling of agent allocation to loops will describe which agents are allocated to the steps and what they do. The Figure 3 above is a good example. It clearly shows that the AdCoS is made of an executive and an adaptive loop and which agents contribute to which loop and to which step. Human agent H1 and machine agent M1 for example implement the perception step in the executive loop: together they perceive the state of the process under control, as well as of the environment. This would for example be the case when a human military agent uses a Night Vision Device (NVD). The device assists in "seeing better".

### 2.1.3 Primitives for AdCoS modelling

The formalism we are going to define to model AdCoS is based on the ideas above. It is based on a finite set of primitive control loops typically found in AdCoS, organized in terms of a control graph. The graph and its primitive loops characterize the AdCoS and are then used to derive the HF requirements associated with that AdCoS.

Given agent allocation to loops in AdCoS can be modelled either at the loop level or at the step level we will consider both.

We will start in 2.1.2.1 with modelling agent allocation at the loop level. We will continue later in 2.1.2.2 with agent allocation at the step level, which should be seen as a refinement of modelling at the loop level, with incremental knowledge of what the agents exactly do being available (in particular to which step they contribute).

#### 2.1.3.1 Primitives with agent allocation at the loop level and how they will be used

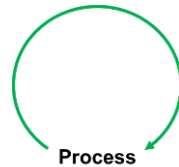
In this scheme thus, we will identify a series of primitives that characterize how agents are allocated at the loop level and how to organize these primitives to model a complete AdCoS. This will be complemented, incrementally, in 2.1.2.2 with additional knowledge of the steps to which the agents contribute and what they do.

A primitive control loop is a sequence of operations aims to control a process (see Figure 1).

A control loop execute every time, the following five operations:

- Perception
- Evaluation
- Decision Making
- Action Planning
- Action Implementation

Then a control loop performs such operations to control a Process. Processes could be a task (T), a task distribution (TD) or any kind of other process (P). The control loop is schematized as follows:



**Figure 8: A control loop that control a process**

As seen above, to enhance the concept of control loop, we added the notion of allocation. Each control loop is managed by an agent. An agent could be human (H), machine (M) or the combination of several human machine (HM). The control loop is then schematized like that:



**Figure 9: A control loop used by an agent to control a process.**

Each control loop act on the environment, we should then specify on **what** the loop is performing. This is the outcomes of the action implementation operation. This outcome may have several aspects. That could be:

- display information on a screen
- generate a sound
- generate a vibration
- or human physical actions
- ...

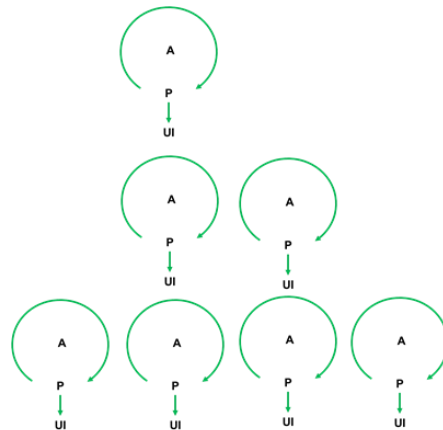
We define these outcomes as User Interface. The control loop is then schematized as:



**Figure 10: A control loop used by an agent to control a process through an user interface**

Then, the framework for adaptation models any kind of AdCoS as a combination of primitive control loops.

An AdCoS is defining as a tree like structure of primitive cognitive loops:



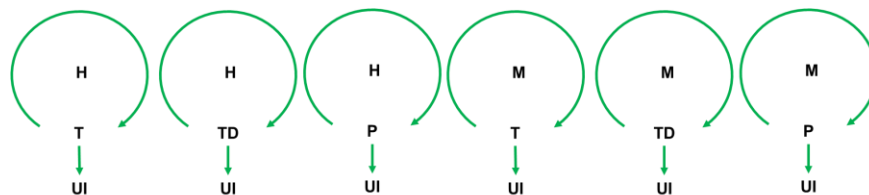
**Figure 11: An example of an AdCoS model with primitive control loops**

The primitives will be used by combining them into control graphs such as those mentioned above (see Figure 4). The Figure 11 shows graph with only seven primitives.

### 2.1.3.2 Primitives

Such control graphs can be built through many different types of primitive loops. We propose the ones below. That list though should not be considered exhaustive. The proposed primitives cover most of the common cases found in AdCoS modelling.

- A **human** that close a loop on a **task** through a **user interface**
- A **human** that close a loop on a **task distribution** through a **user interface**
- A **human** that close a loop on a **process** through a **user interface**
- A **machine** that close a loop on a **task** through a **user interface**
- A **machine** that close a loop on a **task distribution** through a **user interface**
- A **machine** that close a loop on a **process** through a **user interface**



**Figure 12: Finite set of primitive control loops (M: machine, H: human, T: task, TD, task distribution, P: process, UI: user interface)**

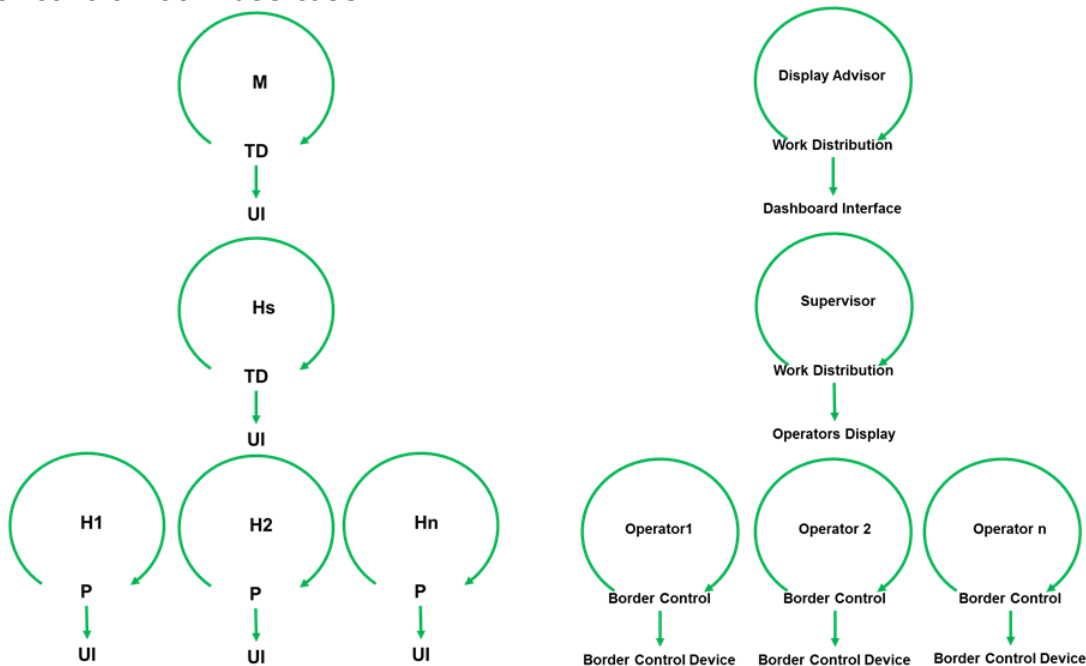


To summarize: an AdCoS is a combination of primitive control loops. Each control loop could be seen as a sub part of the AdCoS system and aims to manage a sub part of the global process. Formally, a primitive control loop is then defined by:

- The **agent (H, M or P)**: that will control the loop
- The **process (T, TD or P)**: on which the loop will be perform
- The **global process** of the AdCoS (not display in the picture as every loop refers to the same global process).
- The **User Interface (UI)**: on which the agent will act to manage the loop.

### 2.1.4 Building an AdCoS model

To build an AdCoS model, one simply combines the different primitives into a control graph that corresponds to the AdCoS. The example below corresponds to the border control room use case.



**Figure 13: Border control room AdCoS modelled with the primitives. On left the primitive graph structure and on right the same structure with variable instantiation.**

Let's describe it in details, starting from the bottom.

- a series of human operators ( $H_1, H_2, \dots, H_n$ ) control specific border portions (P). Their role is to close a control loop on the border and detect anomalies. For



## HoliDes

Holistic Human Factors Design of  
Adaptive Cooperative Human-  
Machine Systems

HoliDes

- that, they interact with the Border control devices (that are computer on which they can perform border control)
- a human supervisor ( $H_s$ ) supervises the operators ( $H_1, H_2, \dots, H_n$ ) by closing a control loop on them. The supervisor controls the task distribution (TD) between the operators. Therefore he or she assesses the presence, condition or workload of each individual and decides of the most optimal task distribution (e.g., portion of the border they have to monitor) between them. Finally, the supervisor act on operators display to update operators' tasks.
  - to perform task distribution between the human operators, the supervisor ( $H_s$ ) receives assistance from a machine assistant system (M). The assistance system closes a control loop on a work distribution (TD) to help the supervisor. The assistance system provide workload information and suggestion of task distribution to the supervisor by mean of a dashboard interface (UI)
  - The model above therefore describes how the supervisor is assisted to perform a work distribution, through a dedicated adaptive UI, in distributing tasks between a series of human operators who monitor a border.

### 2.1.4.1 Primitives with agent allocation at the step level and how they will be used

As seen in 2.1.2, allocation of agents to control loops can be seen and modelled at two levels:

- in the coarse version, modelling allocation at the loop level, we only state that some agents are involved in the execution of a control loop, but nothing is said about their respective roles, and in particular to which step of the control loop they contribute
- in the more detailed and advanced version, modelling allocation at the step level, we specify to which of the five steps of the control loop each individual agent contributes. This therefore requires additional information. On the other hand this will allow understanding in better details how the control loop is achieved by the agents and derive more detailed Human Factors requirements for this execution (see 2.2 below).

The graphical modelling formalism is simply a variation and sophistication of the type of modelling used for the primitives for the simpler case (see 2.1.3) where we only model allocation of the agents to the loop, without detailed knowledge of the loop's steps to which they contribute.

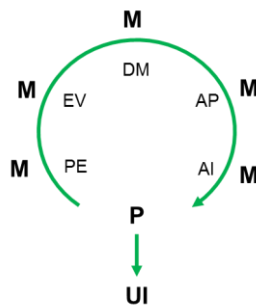


Obviously the formalism is only of interest when there is more than one agent involved in the performance of the loop. In the loop of Figure 14, a single machine agent M is performing the loop.



**Figure 14: A single machine agent M performing a loop**

In that case the machine agent is clearly performing each of the five steps of the loop and there is no need to specify explicitly to which step the agent contributes. This is equivalent to the depiction in Figure 15 where we see that the same agent M performs the Perception (PE), Evaluation (EV), Decision-Making (DM), Action Planning (AP) and Action Implementation (AI) steps...



**Figure 15: The agent M is in fact performing all the steps**

The objective of this section is describe how to go beyond the simple case where a given agent performs the loop alone and address cases where more than one agent are involved (human and/or machine) and the agents in question are allocated to different steps.

In the case of Figure 16, there is a single human agent H and a single machine agent M closing a loop together. The figure shows how this case would be depicted with the primitives of 2.1.3, where we do not specify exactly to which step the two agents contribute.

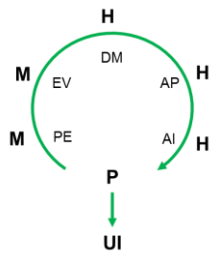


**Figure 16: A human agent H and machine agent M perform the loop**



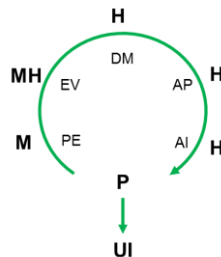


But if we know more of the exact role of each agent, and for example know that all perceptions and evaluations are performed by the machine agent M and all remaining steps, decision-making, action planning and action implementation are achieved by the human agent H, we are in position in Figure 17 to provide a more detailed model of the loop where the allocation of both agents to the steps is specified.



**Figure 17: The human agent H and machine agent M are involved in different steps**

We may also meet more interesting cases where both the human and machine agents contribute to the same step. In the Figure 18 for example, the machine agent is still solely in charge of Perception, but in the Evaluation step, both the human and machine agents are involved, for example with the machine agent contributing some information to the evaluation of the perception provided by the previous step.



**Figure 18: Steps may involve more than one agent**

This could for example be the case in the border control UC when a system provides (Perception) a video feed of some portion of the border to monitor and superimposes additional information on the image to indicate who is foe or enemy (Evaluation). This is a typical case of augmentation where the system provides additional, evaluative information to the human in charge of evaluating the image, for making subsequent decisions and acting accordingly.

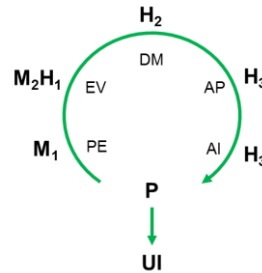
In the more general and final case, there are more than two agents involved in the loop and the allocation of the agents to the steps is more complex. For example in Figure 19, there are 5 agents: 2 machines agents and 3 human agents:



## HoliDes

Holistic Human Factors Design of Adaptive Cooperative Human-Machine Systems

# HoliDes



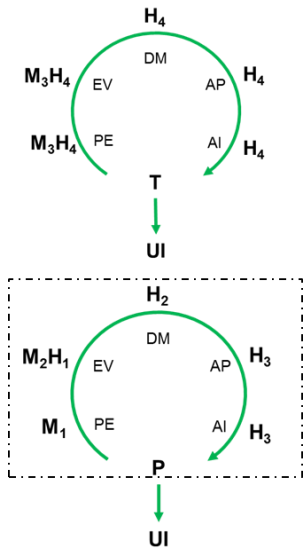
**Figure 19: More than two agents can be involved in a loop**

According to the figure, the agents have the following functions:

- $M_1$  is purely in charge of Perception
- $M_2$  assists (augmentation) a human agent  $H_1$  in charge of Evaluation
- $H_2$ , for example a supervisor, is in charge of Decision-Making, based on the information received from Perception and Evaluation.
- $H_3$ , a human planner and executer is in charge of Action Planning and Implementation, based on the decisions made by  $H_2$ .

So far we only have explained how to depict the allocation of agents to the steps in a control loop. We now have to combine this with the notion of control loop primitives (see 2.1.3 above) and show how they can be used to produce very detailed control graphs, where more than one loop are interacting.

The principle is very simple: we will just replace the loops described in 2.1.3 above by the corresponding, more detailed versions developed in this section. In the Figure 20, we combine two loops: an executive loop (bottom of the figure) made of the 5 agents of Figure 19 acting on an process  $P$  using an interface  $UI$ . The allocation of the agents to the steps is provided in details. That loop is controlled by an adaptive loop that assigns the tasks  $T$  the executive loop has to execute. The adaptive loop is achieved by a human agent  $H_4$ , assisted by a machine agent  $M_3$ . The machine agent, for example a tool for evaluating the workload of the human agents  $H_1$ ,  $H_2$  and  $H_3$  in the executive loop informs the human agent  $H_4$ , who is then in position to make decisions about the tasks to be allocated to the executive loop and communicate them to the agents in question.



**Figure 20: Two fully detailed executive and adaptive loops**

The same primitives can therefore be used, exactly in the same way we were using them when nothing was known of the allocation of the agents but now with that additional information provided. This yields the same type of diagram but enriched with the allocation of the agents to the steps.

As will be seen later in section 2.2, this will allow to use exactly the same Human Factors derivation mechanisms (based on the relations between loops), with additional, incremental ones provided for the agents involved in the steps (based on what is known of their allocation to the steps).

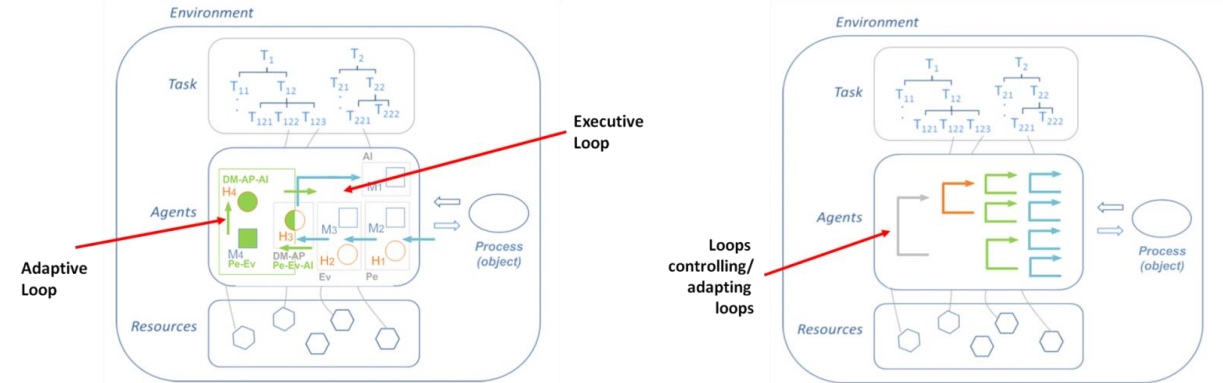
### 2.1.5 How to practically build AdCoS models

The sections above explain how to model AdCoS in terms of imbricated control loops based on a catalogue of control loop primitives.

Building such models is not always easy though, because they are abstract functional versions of something the AdCoS designer perceives as very practical and concrete. The Figure 21a shows a detailed description of an AdCoS (from Figure 3) with an executive and adaptive loop and the assignation of human and machine agents to the different steps. The Figure 21b shows how in more complex AdCoS the different underlying loops are interrelated through a hierarchical control structure, or control graph. It is thus not always easy for the AdCoS designer to produce such descriptions. Though the idea would be to resort to such diagrams at the earliest stages of the AdCoS design process, precisely to better structure, build



and understand the AdCoS, particularly in terms of what it entails for human agents (HF requirements). See section 2.2.



**Figure 21: Detailed description of an AdCoS (a) and a complex control graph (b), based on Figure 3 and Figure 4**

The goal of this section is thus to try to guide the AdCoS designer for the construction of such models, in particular with the modelling formalism described in the section 2.1.3 above.

The recommended approach proceeds in 5 steps:

- determination of executive loops
- determination of agents in executive loops
- determination of adaptive loops
- determination of agents in adaptive loops
- determination of final loops structure (control graph)

### 2.1.5.1 Determination of executive loops

The AdCoS is a Cooperative Human-Machine System (CoS) that presents adaptive features (Ad). So the right question to ask when trying to model an AdCoS is "*What is the CoS doing?*"

In this first step of the approach, we neglect the adaptive part of the AdCoS (in fact the adaptive loops) and only look at what the CoS as a whole is doing. We look at the executive part of the AdCoS.

As shown in Figure 21 above, the CoS is acting on an external process (in fact there can be more than one process, but we will stick to the simple case for the sake of explanation). For example, in the automotive UC, the CoS is made of the driver and several assistant systems and together they control a vehicle and its trajectory (see for example Figure 7). Control is achieved by the CoS via a control loop and as

explained earlier (2.1.1) we call this type of loop an executive loop: this is a loop the CoS *executes* on the process under control.

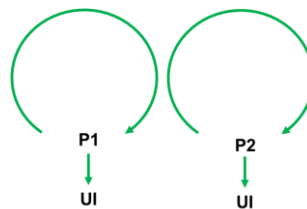
Thus the first step in building AdCoS model is to determine the process on which the (Ad)CoS is acting and what is the executive control loop through which control is achieved. If there are more than one process on which control is performed, additional executive loops will need to be determined. In all cases, executive control is achieved by these loops by taking information on the process (and the environment the process is in, see Figure 21, and processing that information through evaluation, decision-making, action planning and action implementation on the target process. See control loops in 2.1.1. The Figure 21a above shows the only executive loop in the (Ad)CoS, depicted in light blue. And in the Figure 21b, we see that another (Ad)Cos is closing four separate executive loop on the process (or more plausibly four separate sub-processes). These loops are also in light blue in this figure.

The first step of the approach is therefore to determine these executive control loops, mostly based on identifying the processes the (Ad)CoS is acting upon.

The AdCoS executive loops are thus the interface between the core of the AdCoS and the process and its environment (the "external world").

During this first stage of the approach, the modeller should also explicitly identify the process(es) on which the executive loop(s) operate(s). Attention can also be paid to determining the environment in which the AdCoS is immersed, especially if that environment impacts the behaviour of the AdCoS (e.g., weather for an automotive vehicle or an aircraft).

At the end of this stage, we come up with models as in Figure 22.



**Figure 22: Executive loops in example AdCoS model**

At this stage, in that example, we know that the AdCoS is acting on two processes,  $P_1$  and  $P_2$ , but we still do not know how exactly.

### **2.1.5.2 Determination of the agents in the executive loops**

Once one or more executive loops have been determined, one has to determine the agents involved in the performance of the loop. As shown in the many examples above there are many possibilities:

- simple cases where a single agent, human or machine, closes the whole loop (e.g., Figure 14)
- a slightly more complex case where a single human agent and a single machine agent (typically for assistance) perform the loop (e.g., Figure 16)
- more sophisticated versions where there are more than two agents (e.g., Figure 19)
- finally, when there are two agents, one can resort to more detailed descriptions of the agents' allocation to the different loop's steps (e.g., Figure 20)

To determine the agents involved, the best strategy is to consider the executive loop in its entirety, see if it is performed by an obvious single or principal agent (e.g., the driver, the pilot, a supervisor, the practitioner), and if there are systems (i.e. machine agents) involved that contribute to performing the loop or assisting the main agent, identify them. One can also consider the different loop's steps and determine if there are any machine agents that contribute to them. The same step-based strategy should be used when faced with executive loops where there is no clear "main agent" and the relation of machine agents with human agents is more cooperative than assistive).

To assist in the modelling and understanding of the different loops steps, and consequent identification of the agents associated with the steps, the modeller can resort to textual modelling in an Excel file, presented in Figure 23 below.



# HoliDes

## Holistic Human Factors Design of Adaptive Cooperative Human-Machine Systems



Definition	Concept	Data Input	Process Assessment		Data	Process		Data	Action	Data
			Perceptions	Execution		Decision (data)	Action Plan			
WP8 UCS.1 Patient positioning	M1 (display advisor, check list decision, patient position) on priority display H1 (practitioner, patient positioning, patient positioning) MTTs	MR1 sensor data MR1 procedure	Context assessment Evaluate the status of the actions done by the practitioner	Decision making Determine the actions to perform, if any	Decision (data) Decide on how to behave based on the evaluation of the situation	Action Plan Organize the actions, if any	Action Implementation Execute the actions, if any	Action (data) Behaviors that achieve the choices made to be done		
WP8 UCS.4.3 Baracostation	M1 (system position, New Software, 3D) on MCS MTTs	System geometry data Selected ROI	Context assessment Evaluate ROI can be established with current system constraints	Decision making Decide on movement path to ROI without collision decision that movement can be made	Decision (data) Path to move to ROI. Or not made	Action Plan None	Action Implementation None	Action (data) None		
WP8 UCS.2 Operator Guidance	M1 (Work flow engine, work distribution, operator guidance) on global interface, mobile, interface H1 (operator, operator guidance, operator guidance) MTTs	Task status information, operator location workflow Mobile device	Context assessment Evaluate workload of operators available to perform task	Decision making Decide on task	Decision (data) Task	Action Plan None	Action Implementation None	Action (data) None		
WP7 UCS.1.5 IVA	M1 (Display Advisor, Data, Data) on Display interface H1 (operator, Data, Data) MTTs	Aircraft sensor Human monitoring ECG, eye tracking, EEG Display interface Wired Event Director (WED), Cognitive Director Display (CDD)	Context assessment Evaluate workload of operators available to perform task	Decision making Decide on task	Decision (data) Task	Action Plan None	Action Implementation None	Action (data) None		
WP7 UCS.2.1.1 AT	M1 (TTC Train Manager, SM range prediction, timing) on display interface M1 (TTC Train Manager, vehicle, train) on display interface M1 (TTC Train Manager, vehicle, train) on display interface MTTs	Standard operating procedures High speed train High speed train High speed train High speed train	Context assessment Evaluate workload of operators available to perform task	Decision making Decide on task	Decision (data) Task	Action Plan None	Action Implementation None	Action (data) None		
WP8 UCS.1.5 workload warning	M1 (Display Advisor, work distribution, border control) on dashboard interface H1 (supervisor, work distribution, border control) on operators display H1 (operator, border control, border control) MTTs	Line data System, advisory warning Operator (border control, work) on display interface Human sensor Human knowledge	Context assessment Evaluate workload of operators available to perform task	Decision making Decide on task	Decision (data) Task	Action Plan None	Action Implementation None	Action (data) None		
WP9 UCS.1 Lane Change Assistant	M1 (Warning Advisor, warning decision, lane change) on warning interface H1 (driver, lane change, lane change) MTTs	Vehicle sensor data Human sensor data CAM display	Context assessment Evaluate workload of operators available to perform task	Decision making Decide on task	Decision (data) Task	Action Plan None	Action Implementation None	Action (data) None		
WP9 UCS.2 Adaptive Automation Overlaying	M1 (Trajectory control system, Adaptive Overlaying, Adaptive Overlaying) on steering wheel, acceleration pedal MTTs	Self localization Free space estimation Driver distraction estimation	Context assessment Evaluate workload of operators available to perform task	Decision making Decide on task	Decision (data) Task	Action Plan None	Action Implementation None	Action (data) None		
WP9 UCS.3.1 front collision warning	M1 (Warning Advisor, front collision, front collision) on warning interface, brake H1 (Driver, front collision, front collision) MTTs	Data coming from simulated car sensors (radar & camera) Driver's visual scanning (simulate with eye tracking system) COSMO/INT mode or collected from an eye tracking system COSMO/INT mode or collected from an eye tracking system	Context assessment Evaluate workload of operators available to perform task	Decision making Decide on task	Decision (data) Task	Action Plan None	Action Implementation None	Action (data) None		
WP9 UCS.4.1 Lane Change Assistant	M1 (Warning Advisor, front collision, front collision) on warning interface, brake H1 (Driver, front collision, front collision) MTTs	Data coming from simulated car sensors (radar & camera) Driver's visual scanning (simulate with eye tracking system) COSMO/INT mode or collected from an eye tracking system	Context assessment Evaluate workload of operators available to perform task	Decision making Decide on task	Decision (data) Task	Action Plan None	Action Implementation None	Action (data) None		

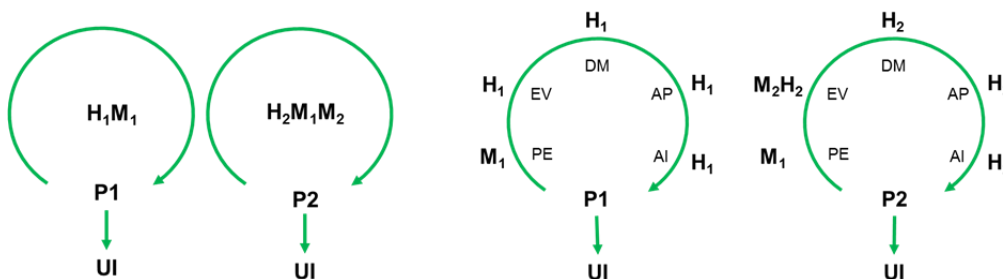
Figure 23: Extract of Excel file to support the elicitation of the content to integrate in loop models

The Excel file of Figure 23 shows all five steps in a loop, with the corresponding sub-tasks, as well as the intermediate data produced and processed by the steps. This helps the modeller better understand how a loop works, in concrete terms, and closer to his or her experience as a (AdCoS) modeller or designer.

The full models of every AdCoS filled in this excel file are in the Annex V. It illustrates for each AdCoS the framework for Adaptations concepts and their corresponding MTTs. It reveals that same concepts (ie: interpretation,) could be cover by several tools. The information stored in the excel file are:



- Input: Main input data
- Context assessment
  - o Perception: Collect and elaborate information from the environment
  - o Evaluation: Evaluate the information according to a referencial
- Context (data): All elements used to evaluate a situation to make a decision
- Decision making
  - o Decision making: Determine actions to perform, if any
- Decision (data): Choices on how to behave based on the evaluation of the situation
- Execution
  - o Action Plan: Organize the actions, if any
  - o Action Implementation: Execute the actions, if any
- Action (data): behaviours that achieve the choices made before
- MTTs

The models identified at the previous step are thus now evolved, with additional information about the agents, and potentially, in the more sophisticated version their allocation to the different steps.



**Figure 24: AdCoS model with allocation of the agents to the executive loops (loop-based & step-based descriptions)**



	<p><b>HoliDes</b></p> <p><b>H</b>olistic Human Factors <b>D</b>esign of Adaptive Cooperative Human- Machine Systems</p>	
---	---	---

One must note that a single physical agent, human or machine, can superpose involvement in more than one step. In the Figure 24 the human agent  $H_1$  and  $H_2$  are involved in four of the steps of their respective loop.

### 2.1.5.3 Determination of adaptive loops

The next step is absolutely crucial. It consists in determining if some of the executive loops in the AdCoS are *adapted* during AdCoS operations. This will allow determining if adaptive loops operate on the AdCoS executive loops.

Let's again consider the simple case where this is a single executive loop in the CoS (like in Figure 21a), though the more complex case where there are several executive loops (Figure 21b) can be treated in the same way.

An executive loop can be characterized by a series of parameters that determine how the executive loop behaves, for example:

- *tasks*: the tasks the executive loop has to perform
- *resources*: the resources available for executing the loop
- *agents*: agents, human and machine, involved in executing the loop
- *task distribution*: how the tasks are allocated to the agents
- *resource allocation*: how the resource are allocated or made available to the agents
- *interfaces*: H2M (human to machine interfaces), H2H (human to human interfaces), M2M (machine to machine interfaces), which allow interaction and communication between the AdCoS agents.

The question to ask is the following one: *is any of these parameters dynamically modified during execution of the executive loop?* F

For example:

- *tasks*: are the tasks the CoS has to achieve changing from time to time? If the executive loop is attempting to control the speed of a vehicle (automotive UC) or follow a specific trajectory (aviation UC), do these target speeds or trajectories change over time? If yes, something is adapting those speeds or trajectories and they are therefore the process of some adaptive loop (hence the imbrication of loops as in Figure 21b).
- *resources*: are the resources available to the CoS for achieving its tasks on the process changing dynamically over time as an effect of intentional entities (i.e. agents) that modify those resources based on dynamic circumstances. For example, in the border control UC, if the weather conditions degrade drastically new sensors, radars, remote sensing tools may be made available to compensate for the additional difficulty of getting information on the



## HoliDes

Holistic Human Factors Design of  
Adaptive Cooperative Human-  
Machine Systems

HoliDes

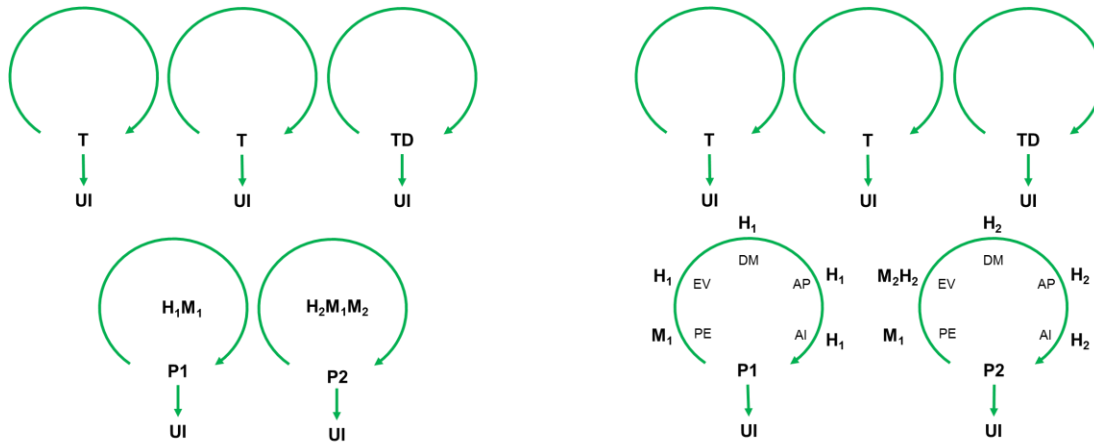
- process and environment. The resources available are adapted to the new circumstances
- *agents*: is the set of agents involved in the loop completely fixed and stable. Or can that set change based on circumstances? For example if the CoS has to deal with new types of tasks the current set of agents cannot correctly deal with alone, additional "specialist" agents are integrated to the CoS, and correspondingly non necessary agents may leave the CoS. Additional agents may also be added when the workload needed to achieve the tasks cannot be met by the current set of agents and additional agents are needed (e.g., staffing).
  - *task distribution*: task distribution specifies how the tasks assigned to the CoS are distributed amongst the agents in the CoS. Task distribution may be fully static (i.e., each agent always performs the same task or type of task) or on the contrary, dynamic, when the task distribution is changed or adapted to new circumstances. For example if some of the human agents start to fatigue, some of their tasks could be distributed to human agents with better capabilities at the time or even to automatic systems (machine agents) capable of performing the same tasks. In this case, task distribution changes or is adapted, here to some state of the agents (fatigue, internal context).
  - *resource allocation*: resource allocation defines the resources each agent has access to. This can be fully static, which agents having permanently access to specific resources, or dynamic, with that allocation dynamically changing over time. Mutual exclusion for resources that can only be used by a single agent at a time for example lead to such dynamic allocations. On most airports, aircraft willing to land have to be coordinated by an adaptive agent (air traffic controller) to land because an airport with a single runway can only accept a single aircraft at a time. The runway as a resource is dynamically allocated to the aircraft willing to land.
  - *interfaces*: interfaces between agents, in particular human-machine interfaces can be the process of adaptation, with adaptation of physical settings (e.g., illumination level), content or format of information,... based on circumstances.

Thus, the goal here is to determine if the parameters that characterize a given executive loop are sometimes changed by an **intentional** agent that adapts them to some circumstances. If yes, that agent closes an adaptive loop on these parameters. The executive loop is *adapted*. A corresponding adaptive loop will need to be included in the (Ad)CoS model.

The same investigation procedure must be conducted independently for each executive loop in the CoS, if there is more than one.



One therefore comes up with further evolutions of the executive loop models, with additional identification of the parameter(s) or the executive loop(s) that are adapted by some external adaptive loop(s).



**Figure 25: AdCoS model with adapted parameters and adaptive loops for the two executive loops (loop-based & step-based descriptions)**

In Figure 25: AdCoS model with adapted parameters and adaptive loops for the two executive loops (loop-based & step-based descriptions) we see that the adaptation on the first executive loop on  $P_1$  is on the Tasks performed by the loop. For the second executive loop on  $P_2$ , the adaptation is on the Tasks but also Task Distribution. In that second loop,  $H_2$  and  $M_2$  contribute together to the Evaluation step, and task distribution will define dynamically, adaptively how the human and machine agent will share the workload (based for example on measurement of the human agent's workload). There is thus a single adaptive loop on the executive loop on  $P_1$  and two on the executive loop on  $P_2$ .

If there are at least one adapted executive loop and a corresponding adaptive loop in a CoS, the CoS is an AdCoS. Otherwise the CoS is a pure CoS, with no adaptive capabilities.

One must note that a given executive loop can be controlled by more than one adaptive loop. This happens when more than one parameter of the executive loop (e.g., Tasks and Agents) are adapted (e.g., one adaptive loop adapts the Tasks to perform, based on variations in the process and the environment, and a second one adapts the number and speciality of the agents according to the new task load).

Nothing yet though is known of the details of the adaptive loop(s) and in particular of the agent(s) they involve.

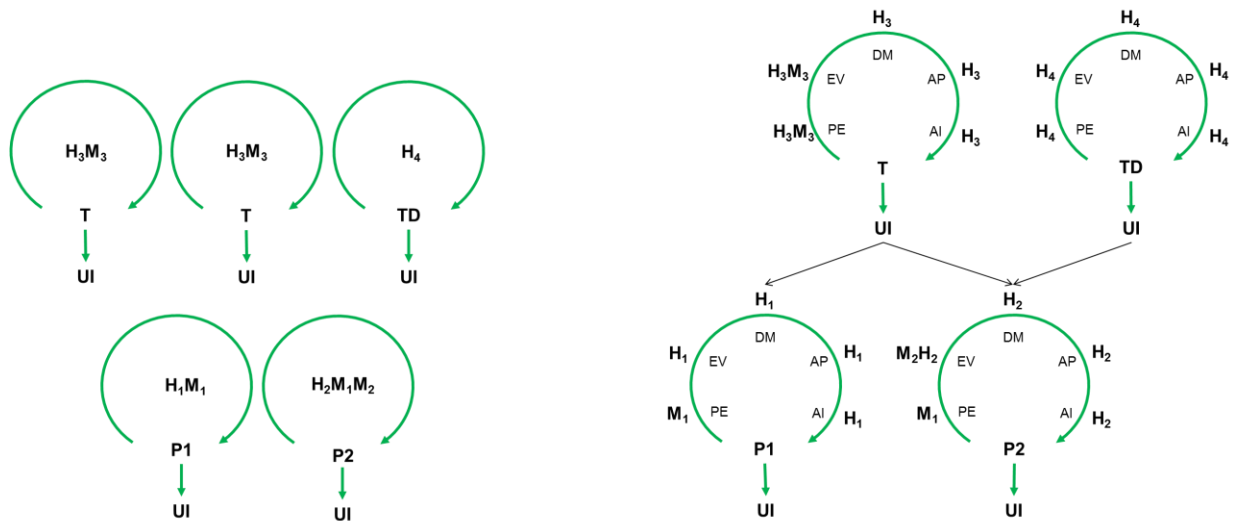


**2.1.5.4 Determination of the agents in the adaptive loops**

This obviously consists, as for executive loops, in determining which agents are involved in the adaptive loop(s), possibly with some knowledge of the steps to which the agents contribute.

The strategy for determining the agents contributing to an adaptive loop is the same than for an executive loop. See above.



This yields further refinements to the AdCoS model, now sporting executive loops interfacing the AdCoS with the external world (process and environment) and one or more adaptive loops acting on these executive loops.



**Figure 26: AdCos model with agents determined for both executive and adaptive loops (loop-based and step-based descriptions)**

Three adaptive loops control the three parameters for the two executive loops. The two first adaptive loops are identical (same agents, same task distribution, same target parameter) and in the step-based description on the right they have been aggregated on a single adaptive loop that controls Tasks for both executive loops. Task distribution for the second executive loop is controlled by a dedicated adaptive loop. The first adaptive loop may be adapting Tasks for both loops based on changes on the target process P<sub>1</sub> or P<sub>2</sub> or in the environment. The second one may be adapting task distribution between human and machine agents in the second executive loop based on these dynamic task definitions.

**2.1.5.5 Determination of potential additional adaptive loops**

	<p><b>HoliDes</b></p> <p><b>H</b>olistic Human Factors <b>D</b>esign of Adaptive Cooperative Human- Machine Systems</p>	
--	---	---

As shown in Figure 21b, adaptive loops can be controlled by other adaptive loops.

Thus when a new adaptive loop has been found, one has to wonder if it is not itself adapted in some way. The adaptation of loops, be they executive or adaptive, is always through the parameters that characterize the loop. Thus, the list of parameters identified for executive loops in 2.1.5.3 can be reused for adaptive loop adaptation.

The same approach then for executive loops can be used to determine if they are adapted or not, and their associated adaptive loop(s). The same type of modelling can be used to derive the complete control graph of imbricate executive and adaptive loops for the AdCoS.

#### **2.1.5.6 Determination of executive and adaptive loop(s) structure (control graph)**

The procedure of trying to determine all executive and adaptive loops in an AdCoS should thus be applied, including iteratively on adaptive loops until no more significant adaptive loop can be found.

This leads to the question of deciding what to include in the AdCoS model or not. Adaptive loops can usually be related to loops of little interest or impact on the AdCoS (e.g., in some cases, organizational adaptive loops in the context of operation that are not useful to include). It is up to the modeller to decide "when to stop". The goal is to understand how the AdCoS is structured and works, to determine requirements, including HF, that should be satisfied for the AdCoS performance to be optimal. Thus, the level of detail (e.g., loop-based vs step-based allocation of agents, number of layers of adaptive loops,...) should be correlated to the desired level of understanding of the AdCoS performance and the level of optimization required.

In fine, all loops obtained at this stage, executive and adaptive, including their interrelations should be aggregated into a single control structure that takes the shape of a control graph, specifying which loops act on which loop, through which parameter(s) and with which agents involved (at a loop or step level). See Figure 21 for an example (abstracted to the structure only).

This will constitute the complete AdCoS model. Such models can then be used to automatically derive requirements for the agents, human and machine, in the AdCoS, which a peculiar focus on HF requirements for human agents.

## 2.2 Automatic derivation of HF requirements from an AdCoS model

The border control room example makes it obvious that specific relations exist between the different graph elements: human supervisor ( $H_s$ ), user interface (UI), assistant machine agent (M), the human operators ( $H_1, H_2, \dots, H_n$ ) and their task distribution (TD), and finally the border itself (P).

Specific Human Factors (HF) requirements can be derived from these requirements. For example (non exhaustively):

- the human supervisor ( $H_s$ ) must be able to perceive the user interface (UI)
- the assistant machine agent (M) must be able to perform the state (e.g., workload) of the human operators ( $H_1, H_2, \dots, H_n$ )
- the human supervisor ( $H_s$ ) must be able to control the operators task distribution (TD), typically by communicating with the operators, naturally or electronically.

The structure of the graph, the relations between the primitives, the process types they control, the nature of the human or machine agents involved, ... determines the HF requirements the AdCoS has to satisfy.

The goal of the method we propose here is to produce these requirements automatically, from the AdCoS model.

### 2.2.1 Families of HF requirements

The method addresses HF requirements on the following topics, derived from the work in WP1 and the HF-RTP:

- **CL**: cognitive capacity limits
- **CM**: communication
- **CP**: cooperation
- **DM**: decision-making
- **FA**: fatigue
- **SA**: situation awareness
- **ST**: (user) satisfaction
- **TA**: technology acceptance
- **TR**: trust in automation
- **US**: usability

- **VD:** visual distraction
- **WL:** workload

### 2.2.2 Associating HF requirements with individual AdCoS primitives

The first objective is to produce HF requirements for our different types of primitives. The requirements must cover, whenever applicable, the families of HF requirements above.

The requirements for a given primitive are derived from its components:

- the control agent: H, M, or HM
- the type of controlled process: T, TD, P
- The outcome resource of the primitive: UI

We associate a series of HF requirements to each type of control agent and another series for each type of controlled processes.

#### 2.2.2.1 HF requirements for agents

<b>HF Requirements for Human Agents (H)</b>	
WL	<human agent> workload must stay in acceptable bounds (low & high)
FA	<human agent> fatigue must stay in acceptable bounds (low & high) for the operations performed in this loop
CL	the operations requested from <human agent> must stay below its cognitive capacity limits
VD	<human agent> must not be visually distracted in operations where visual perception and evaluation are involved
ST	the operations in which <human agent> are involved must provide satisfaction
SA	<human agent> may need to inspect its states (introspection)

<b>HF Requirements for Human and Machine Agents (HM) closing the loop cooperatively</b>	
SA	An <agent> must be able to detect that changes to the <obj> are made by one or more other <agent(s)>
CP	An <agent> must be able to collaborate with the other <agent(s)> on the changes to make to the <obj>, through communication
CM	An <agent> may need to be able to communicate about its internal states with the other <agent(s)>
TR	<human agent(s)> must have trust in <machine agent(s)>
TA	the <machine agent(s)> with which <human agent(s)> cooperate must be acceptable and accepted by the <human agent(s)>

#### 2.2.2.2 HF requirements for types of controlled processes

<b>HF Requirements for User Interfaces (UI)</b>	
SA	<agent> must be able to perceive and evaluate <immediate input> state(s)
SA	<agent> may be able to perceive and evaluate <heritage(input)> state(s)



## HoliDes

Holistic Human Factors Design of Adaptive Cooperative Human-Machine Systems



SA	<agent> must be able to perceive and evaluate <immediate input> state(s)
SA	<agent> may be able to perceive and evaluate <non immediate input> state(s)
SA	<agent> must be able to perceive and evaluate <UI> state(s)
SA	<agent> must be able to perceive and evaluate the <input operational environment(s)> (in which the <input> is)
SA	<agent> must be able to perceive and evaluate the <UI operational environment> (in which the <UI> is used)
DM	Information presented on the <UI> and about the <input operational environment> and the <UI operational environment> must allow the <agent> to decide if the <UI> needs to be changed
DM	Information presented on the <UI> and about the and about the <input operational environment(s)> and the <UI operational environment> must allow the <agent> to decide how the <UI> needs to be changed
US	<agent> must be able to access the controls that allow changing the <UI>
SA	<agent> must get a feedback from the changes made on the <UI>

HF Requirements for tasks (T)	
SA	<agent> must be able to perceive and evaluate the current state of the <tasks>
SA	<agent> must be able to perceive and evaluate the <state of the operational environment> in which the <tasks> are executed
SA	<agent> must be able to perceive and evaluate the <state of process of control> on which the <tasks> are executed
SA	<agent> must be able to perceive and evaluate the <state of the agents> to which the <tasks> are assigned
SA	<agent> must be able to perceive and evaluate the <state of the resources> involved in the achievement of the <tasks>
DM	Information on the <tasks> and the <state of the operational environment> and the <tasks> <agents> must allow the <agent> to decide if the <tasks> need to be changed
DM	Information on the <tasks> and the <state of the operational environment> and the <tasks> agents must allow the <agent> to decide how the <tasks> need to be changed
US	<agent> must be able to change the <tasks>
SA	<agent> must get a feedback on the changes made to the <tasks>

HF Requirements for task distributions (TD)	
SA	<agent> must be able to perceive and evaluate the current state of the task distribution
SA	<agent> must be able to perceive and evaluate the operational environment in which the task distribution operates
SA	<agent> must be able to perceive and evaluate the <process of control> on which the task distribution operates
SA	<agent> must be able to perceive and evaluate the state of the agents to which the tasks are assigned
SA	<agent> must be able to perceive and evaluate the state of the resources involved in the achievement of the tasks
DM	Information on the tasks and the operational environment and the tasks agents must allow the <agent> to decide if the tasks need to be changed
DM	Information on the tasks and the operational environment and the tasks agents must allow the <agent> to decide how the tasks need to be changed
US	<agent> must be able to change the task distribution
SA	<agent> must get a feedback on the changes made to the task distribution

HF Requirements for generic process type (P)	
SA	<agent> must be able to perceive and evaluate the state of <obj>
SA	<agent> must be able to perceive and evaluate the <operational environment> in which <obj> is



	controlled
DM	Information on <obj> about the <operational environment> must allow the <agent> to decide if <obj> needs to be changed
DM	Information on <obj> and about the <operational environment> must allow the <agent> to decide how <obj> needs to be changed
US	<agent> must be able to change <obj>
SA	<agent> must get a feedback from the changes made on <obj>

### 2.2.2.3 Instantiation of primitives and derivation of requirements

As explained in 2.1.3.2, a primitive has a few free variables. In the primitive models above, there are four free variables:

- the control agent involved
- the type of controlled process
- the name of the global process
- the user interface used to control the process.

In a peculiar AdCoS, these variables will have specific values.

e.g. control agent: "ADAS"

e.g. type of controlled process: "adaptive user interface for ADAS"

When the values of the free variables have been assigned, the primitive is said to be instantiated. To be usable for derivation of HF requirements, an AdCoS model must be first completely instantiated. All free variables must be assigned a value. In the border control room use case, we could for example have the following variable assignments:

- M -> "Display advisor"
- UI -> "Dashboard interface"
- H<sub>s</sub> -> "Supervisor"
- TD -> "Work distribution"
- H<sub>1</sub>, H<sub>2</sub>,... H<sub>n</sub> -> "Human operator 1", "Human operator 2",... "Human operator n"
- P<sub>1</sub>, P<sub>2</sub>,..., P<sub>n</sub> -> "Border portion 1", "Border portion 2",... "Border portion n"

### 2.2.2.4 Example of instantiation: HF requirements for H<sub>s</sub> ⇔ TD in border control room AdCoS

<b>Workload</b>	"Supervisor" workload must stay in acceptable bounds
<b>Fatigue</b>	"Supervisor" fatigue must stay in acceptable bounds for the operation perform in the loop
<b>Cognitive</b>	The operations requested from "Supervisor" must stay below its cognitive

<b>Capacity Limit</b>	capacity limits
<b>Visual Distraction</b>	"Supervisor" must not be visually distracted in operations where visual perception and evaluation are involved
<b>Satisfaction</b>	The operations in which "Supervisor" are involved must provide satisfaction
<b>Situation Awareness</b>	"Supervisor" may need to inspect its states (introspection)
<b>Situation Awareness</b>	"Supervisor" must be able to perceive and evaluate the current state of the "Work Distribution"
<b>Situation Awareness</b>	"Supervisor" must be able to perceive and evaluate the "Border under Control" environment in which the "Work Distribution" operates
<b>Situation Awareness</b>	"Supervisor" must be able to perceive and evaluate the "Border under Control" on which the "Work Distribution" operates
<b>Situation Awareness</b>	"Supervisor" must be able to perceive and evaluate the state of the "Operators" to which the "Border Control" are assigned
<b>Situation Awareness</b>	"Supervisor" must be able to perceive and evaluate the state of the resources involved in the achievement of the "Border under Control"
<b>Situation Awareness</b>	"Supervisor" must get a feedback on the changes made to the "Work Distribution"
<b>Decision Making</b>	Information on the "Border under Control" must allow the "Supervisor" to decide if the "Border under Control" need to be changed
<b>Decision Making</b>	Information on the "Border under Control" must allow the "Supervisor" to decide how the "Border under Control" need to be changed
<b>Usability</b>	"Supervisor" must be able to change the "Work Distribution"

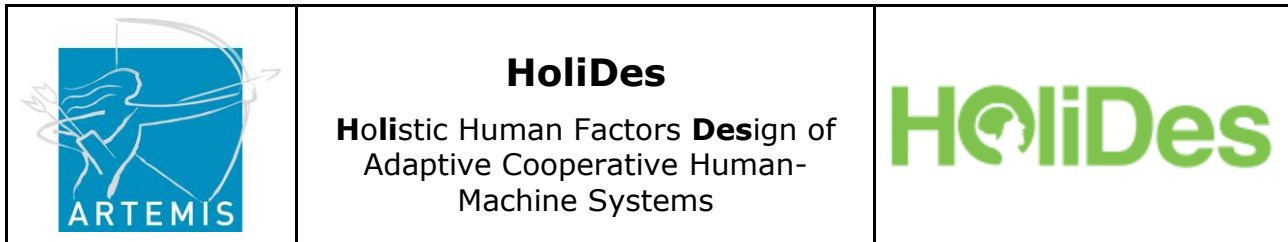
### 2.2.3 Derivation of additional HF requirements when the allocation of the agents to the steps is known

The derivation of HF requirements above was based on the models of primitives presented in 2.1.3.

We have seen though that when knowledge of how the agents are allocated to the different steps in a loop, more detailed models can be produced (e.g., Figure 21). Thanks to that knowledge, additional HF requirements can be derived that complement those already obtained by the processing of the primitives in the control graph that depicts the AdCoS.

These new requirements are derived from three key ideas:

- (a) the agents are now allocated into specific steps, therefore more is known about the tasks the agents have to perform (e.g., Evaluation) and therefore additional requirements for those agents can be derived from knowledge of these tasks
- (b) the agent(s) in a given step have to pass information to the next step in the loop, and therefore interact with the agent(s) involved in that step (this does not apply to Action Implementation, given it is the last step in a loop)
- (c) the agents in a given step, if more than one, have to work together to perform the task assigned to the step (e.g., decision-making)



The requirements are produced incrementally to those already obtained with the primitive-based method described in 2.2.2.

Each (executive and adaptive) loop in the AdCoS must be processed separately. The requirements obtained by processing a loop will be added to the pool of requirements already obtained with the primitive-based method.

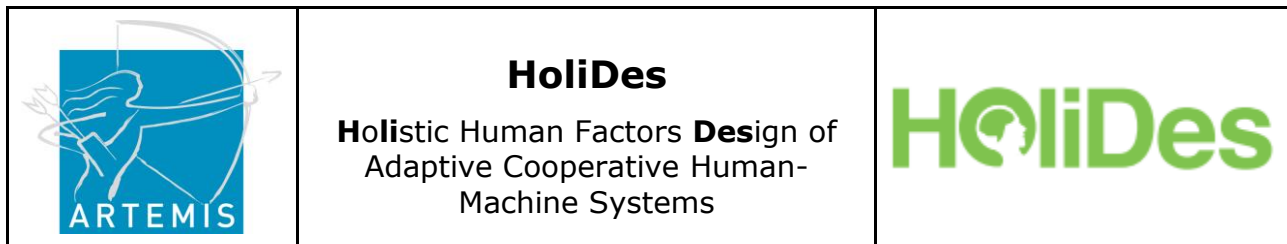
When a loop is processed, each step must be processed separately to generate requirements specific to this step and those agents, human and machine, involved in the step. The requirements obtained by processing a step will be added to the pool of requirements already obtained for the loop.

In fine, the new requirements are therefore obtained by processing all individual steps and adding the corresponding set of requirements to those already available.

To derive the requirements associated with a step, we must distinguish three cases:

- (1) the step is exclusively performed by one or more machine agents: in this case, a human agent is not involved, and there are therefore no additional HF requirement associated with the step.
- (2) the step is performed by a mix of human and machine agents, with at least one human agent. The human and machine agents form a small human-machine cooperative system (CoS) local to the step. These CoS can take various shapes, depending on their organisation and how the tasks relative to the step are distributed between the human and machine agents. In terms of derivation of HF requirements this is the most complex case given these requirements will strongly depend on the tasks or roles assigned to the human agents in the local CoS (local to the step). A possible approach to deal with that complexity is to apply the HF requirements derivation methodology recursively to the CoS under concern.
- (3) the step is exclusively performed by a single human agent: in this case, there are indeed new HF requirements to derive and they are exclusively related to the specific task performed by the step (i.e., perception, evaluation, decision-making, action planning and action implementation) and to the relations between that step and the preceding and following ones (if any).

Given no HF requirements need to be derived from case (1), or could be derived recursively from the CoS local to the step in case (2), we will only address the third case (3), when the step is performed by a single human agent. As hinted above, the HF requirements fall in two cases: (a) HF requirements related to the task performed by the step and (b) HF requirements related to the interaction with the



previous and next step (if any). The third case (c) is of no concern here given only a single (human) agent is involved in the step.



- (a) HF requirements associated with the task performed by the step

<b>HF Requirements for Human Agents (H) involved in Perception</b>	
PEP	<human agent> must have access to all perceptive information (sensors or direct perception) needed to Perceive the object or process of the loop (executive or adaptive) the step is in
PEF	The perceptive information made available to <human agent> must be in a form that is perceptible by humans, either directly or through some transformative equipment that makes the information directly perceptible.
PEW	The workload associated with the performance of the step by <human agent> must not beyond human capabilities (e.g., bandwidth of data flow) and must be compatible with the workload expected from the involvement of <human agent> in other tasks.

<b>HF Requirements for Human Agents (H) involved in Evaluation</b>	
EVF	<human agent> must have access to the results of the Perception step in a form that is adequate for Evaluation.
EVK	<human agent> must have access or knowledge of, implicit or explicit, an evaluation function that determines how to evaluate the Perceptive information. This typically involves some knowledge of the tasks assigned to the loop. <human agent> must then have some knowledge, implicit or explicit, of these tasks.
EVW	The workload associated with the performance of the step by <human agent> must not beyond human capabilities (e.g., bandwidth of Perceptive information to evaluate) and must be compatible with the workload expected from the involvement of <human agent> in other tasks.

<b>HF Requirements for Human Agents (H) involved in Decision-Making</b>	
DMF	<human agent> must have access to the results of the Evaluation step in a form that is adequate for Decision-Making.
DMK	<human agent> must have access to or knowledge of, implicit or explicit, a decision-making procedure or algorithm that determines how decisions shall be made.
DMG	<human agent> must have access to or knowledge of, implicit or explicit, the tasks the loop has to perform.
DMC	<human agent> must have knowledge, implicit or explicit, of the agents involved in the Action Planning and Action Implementation steps and of the actions or class of actions accessible to the loop, in order to determine the acceptability and achievability of tentative decisions.
DMW	The workload associated with the performance of the step by <human agent> must not beyond human capabilities (e.g., number of decisions to make in a given period of time) and must be compatible with the workload expected from the involvement of <human agent> in other tasks.

<b>HF Requirements for Human Agents (H) involved in Action Planning</b>	
APF	<human agent> must have access to the results of the Decision-Making step in a form that is adequate for Action Planning.
APK	<human agent> must have access to or knowledge of, explicit or implicit, an action planning procedure or algorithm that allows producing action plans appropriate and achievable by the loop.
APR	<human agent> must have access to or knowledge of, explicit or implicit, the action repertoire available to the loop and the costs and benefits (in terms of task completion or goal achievement) associated with the execution of these actions.
APC	<human agent> must have access to or knowledge of, explicit or implicit, of the constraints that lay on the actions and their combinations into action plans that make tentative action plans non

	<h2>HoliDes</h2> <p><b>H</b>olistic Human Factors <b>D</b>esign of Adaptive Cooperative Human- Machine Systems</p>	
--	--	---

	executable or non-achievable.
APW	The workload associated with the performance of the step by <human agent> must not beyond human capabilities (e.g., number of action plans to produce in a given period of time) and must be compatible with the workload expected from the involvement of <human agent> in other tasks.

HF Requirements for Human Agents (H) involved in Action Implementation	
AIF	<human agent> must have access to the results of the Action Planning step in a form that is adequate for Action Implementation.
AIA	<human agent> must have access to the set of actions involved in actions plans and the capability to execute or trigger them.
AIW	The workload associated with the performance of the step by <human agent> must not beyond human capabilities (e.g., amount of actions to implement in a given period of time) and must be compatible with the workload expected from the involvement of <human agent> in other tasks.

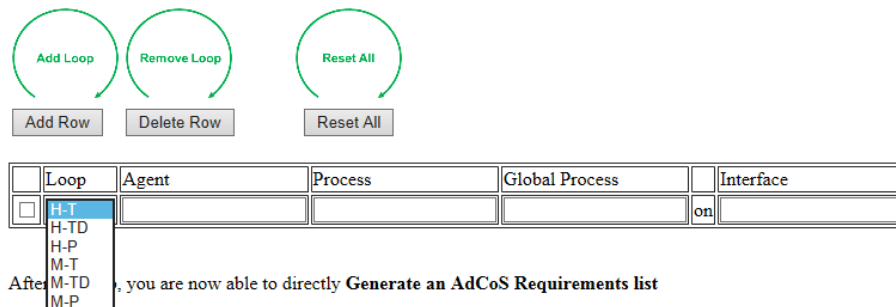
- (b) HF requirement associated with the interaction with the previous and next steps (if any).

HF Requirements for Human Agents (H) involved in a step	
INT	<human agent> must be able to interact with the agents involved in the previous and following step in the loop, if any. If <human agent> is one of the agents involved in any of those steps, interaction is obviously satisfied (e.g., in working memory).

### 2.2.4 Implementation of AP in the Platform Builder

An implementation of the framework for adaptation and the generation of HF requirements have been integrated in the platform builder. A description of how to use it is described in the D1.10 deliverable focus on the platform builder. Below a screen shot of the HTML user interface:

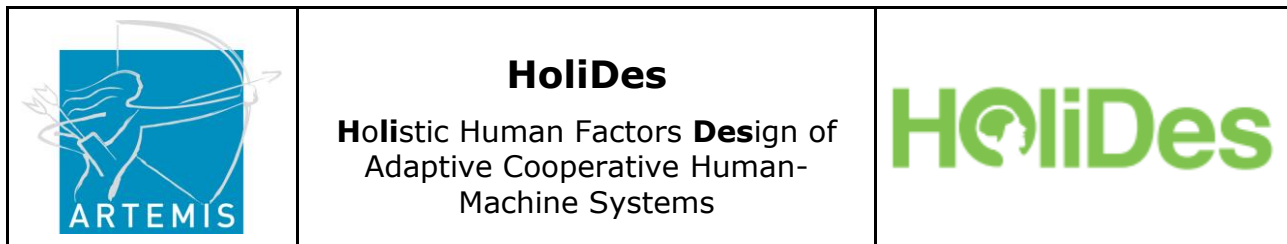
**Step 1:** Please add, remove loops... or clear tab, by clicking on the following icons. And fill in the tab below with the relevant items,



Loop	Agent	Process	Global Process	Interface
<input type="checkbox"/> H-T				<input type="checkbox"/> on
<input type="checkbox"/> H-TD				
<input type="checkbox"/> H-P				
<input type="checkbox"/> M-T				
<input type="checkbox"/> M-TD				
<input type="checkbox"/> M-P				

After M-TD, you are now able to directly **Generate an AdCoS Requirements list**

**Step 2 :** Please click on the button below to verify the loops involved in your AdCoS and to obtain your AdCoS requirements list,



**Figure 27: framework for adaptation in the in the platform builder application**

### **3 Resolution process**

#### **3.1 UC2, Diversion Assistant**

The Diversion Assistant provides support in situation when crew needs to change the destination because of an unpredicted event during a flight. The Diversion Assistant integrates various sources of information spread across the cockpit and paperwork and uses the information to prioritize airports in reach with respect to suitability to current situation of the aircraft, the environment and the crew.

The adaptation in Diversion Assistant use-case is based on two complementary strategies:

1. Manipulating of the electronic flight bag (EFB) device assumes head-down time and the system must assure that crew while using the device does not loose contact with the situation or even break operating procedures.
2. A state of the operator with respect to workload and attention/distraction is used to adjust information presented to the operator. Having determined a high workload situation, the level of information is adjusted and also the prioritization process reflects the reduced ability of pilot to comprehend all relevant information.

The rational for the selected strategies reflects the fact that EFB device provides only supportive information that has always lower priority than information presented on avionics displays and so the system adaption provides means to enforce the operating procedures, e.g. monitoring avionics displays at defined times, and avoid missing important changes of the situation (strategy 1 enabled by Missed Event Detector tool-MED).

Airport prioritization is a multidimensional optimization problem that takes into account various aspects of flying to and airport, landing at an airport, staying at an airport and connecting to the original destination. State of the pilot is one of aspects to be considered, e.g. a fatigued pilot should rather land at an airport with lower traffic or easier approach procedures (strategy 2 enabled by Pilot Pattern Classifier-PPC or Cognitive Distraction Classifier-CDC tools).

The concepts of the adaption in Diversion assistant have been discussed in previous deliverable, D3.5 – Techniques and Tools for Adaptation Vs1.5. The current deliverable reflects the process of implementation and first results of AdCoS validations. The first results cover individual dedicated tests of tools connected to

the Diversion Assistant; the validation of integrated AdCoS is described in a dedicated deliverable (D7.8).

### 3.1.1 Data flow enabled by HF-RTP tool chain

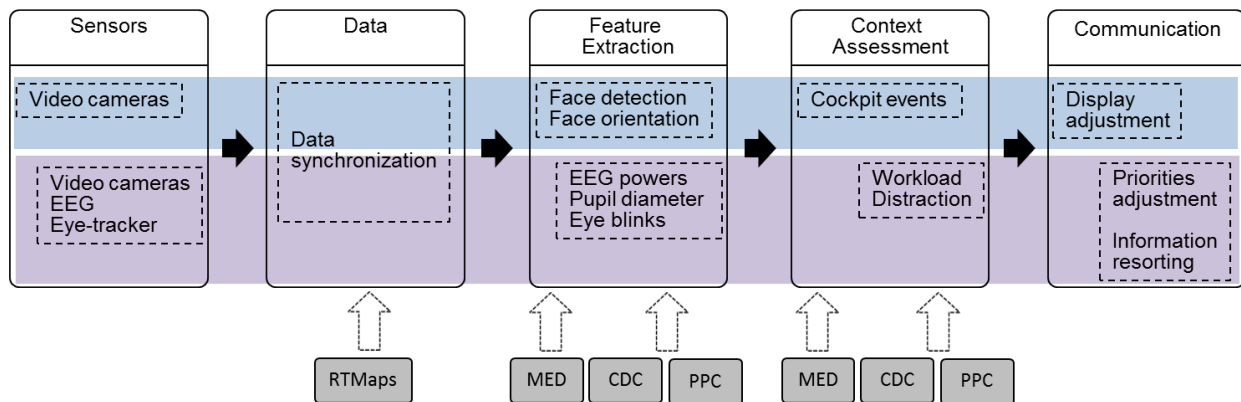
The general data flow consists of data acquisition, data processing, feature extraction, context assessment and communication, see Figure 28. The data flow is common to both adaptation strategies, but execution of particular steps may differ.

Data acquisition for MED and CDC relies on unobtrusive video recordings, while PPC uses head mounted EEG cap and eye-tracker.

Feature extraction for MED determines the relative orientation of head with respect to the cockpit geometry, while workload/distraction strategy uses more detailed features: EEG waves power and eye-related metrics such as pupil diameter, eye closure or eye blinks.

Context assessment for MED tries to interpret what happens in the aircraft systems and how the pilot activities are aligned with aircraft status. CDC and PPC transform physiological data to cognitive state of pilot.

Communication for MED leads to alerting and orienting the pilot to where his attention is needed. Information from CDC and PPC is used rather silently to adjust background calculations in Diversion Assistant.



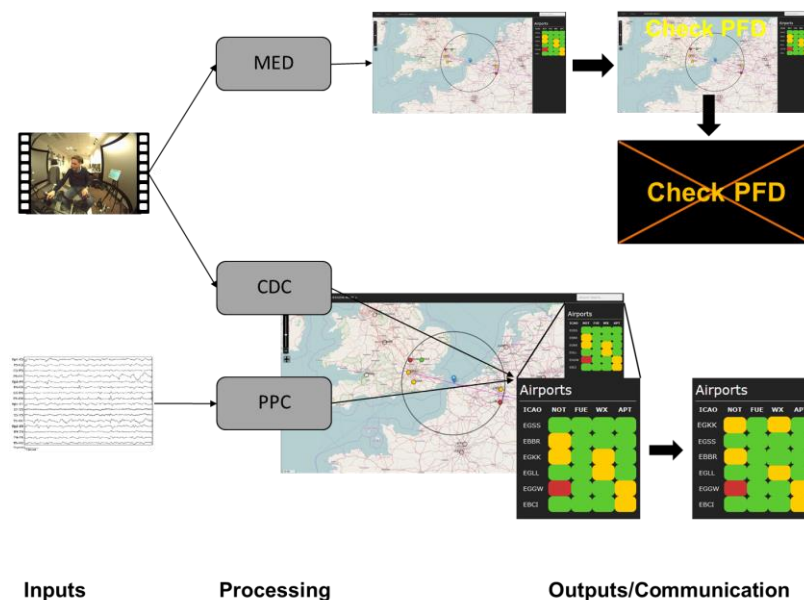
**Figure 28: Data flow supporting the adaptation strategies for Diversion Assistant. Both strategies assume the same steps, though algorithms and tools applied to accomplish the steps differ, see colour highlight.**

### 3.1.2 Inputs & outputs

The raw data from sensors are pre-processed using generic algorithms. The pre-processing is currently part of individual tools, though it may be modularized in future to allow for more efficient tailoring, e.g. face detection in video stream may be out-sourced to a dedicated tool and re-used by MED and CDC instead of having the face detection algorithm in both tools duplicated.

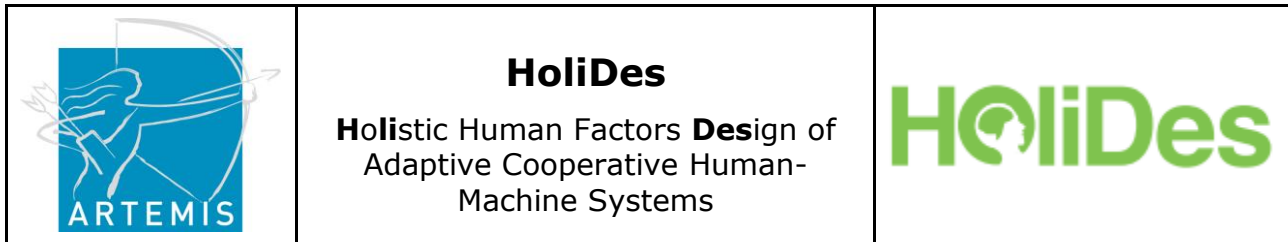
Pre-processed data enter respective analytics loop to provide pilot state data, see Figure 29. In missed event detection the pilot state is defined as being or not being able to perceive information from specific high priority sources. The adaption is triggered only if a high priority source requires attention and at the same time pilot is not monitoring that source. The adaption is communicated via a message shown on a display where pilot looks at. More details are given in previous deliverable D7.8 – Tailored HF-RTP.

In workload/distraction detection, the level of workload or distraction is used to modify weight for airport parameters that are used in prioritization algorithm. Details on prioritization are given in previous deliverable D7.7 – Implementation of aeronautics AdCoS. As output, the prioritization of airports changes with respect to the pilot state.



**Figure 29: Tools enabling the transfer of raw input data, e.g. camera or EEG, to means for triggering adaption.**





Explanation of the above figure: in first strategy, pilot is first warned of important message on higher priority display. If pilot ignores the warning, the low priority display such as EFB may be switched off to enforce the operating procedures. In the second strategy, the airports' priorities change and so the list communicated to the pilot is reorganized as a result of high workload indication.

### 3.1.3 HF-RTP tools applied to realize adaptation

#### 3.1.3.1 Missed Event Detector

MED (developed by BUT) is used as a stand-alone tool that takes video camera data and retrieves real-time information of operator head direction within the working environment, i.e. aircraft cockpit. MED communicates the information in terms of region of interested (ROI) currently monitored by the operator. Diversion Assistant compares the observed ROI with current aircraft status in order to prevent use of the tablet when inappropriate. Details on MED are in other deliverables:

- MED architecture and status in D5.5 – Techniques and Tools for Empirical Analysis
- MED validation in D7.9 – Empirical Evaluation of Aeronautics AdCoS

#### 3.1.3.2 Pilot Pattern Classifier

The model for the Pilot Pattern Classifier (developed by TEC) has been developed in **Python**, using some machine learning libraries such as **sklearn**, and some other scientific libraries such as **numpy**, **pandas**, **scipy**.

The model was based on the **Nearest Neighbours (k-NN)** technique (with these parameters:  $k=3$ ), and used 75% of the data sets for training and 25% for testing, using **stratified cross-validation**.

ANN (Artificial Neural Networks) and SVM (Support Vectors Machine) are well known techniques applied to EEG data obtaining good results in binary problems. In our case, however, we dealt with a multiclass problem. We used the RFC and k-NN. Both of them are usually applied to multiclass problems. We performed different tests by combining different weights for each class (due to the imbalance problem of the data sets), and by trying with several parameters of the RFC (such as the number of trees, the depth, etc.). We achieved the best results with k-NN when  $k=3$ .

We have used the stratified cross-validation procedure to test the model in order to avoid over-fitting.

When dealing with binary classification, the accuracy of the classification results should be evaluated in terms of standard classification metrics such as precision,

recall, and F-score. On the other hand, when dealing with a multiclass classification problem, like the one we were considering, other metrics should be taken into account, like micro and macro averaged scores. Accuracy is sometimes quite misleading, as you may have a model with relatively 'high' accuracy predicting the 'not so important' class labels fairly accurately but not the classes that are actually critical to the application. In our case, we treat all classes equally, thus our metrics were (i) macro averaged precision, (ii) macro averaged recall, and (iii) macro averaged F-score.

### 3.1.3.3 Cognitive Distraction Classifier

CDC (developed by TWT) is another machine learning based tool that evaluates video stream in real time in order to derive facial and eye based metrics for classification of distraction in drivers. Details on CDC are in previous deliverable D5.5 – Techniques and Tools for Empirical Analysis.

The tool was designed for use in automotive and it relies to certain aspect on typical driver behaviour – i.e. driver is supposed to monitor space in front of him. While in aviation, pilot behaviour is much more complex, it was necessary to conduct a study on identifying similarities and differences between drivers and pilots. As result, it was concluded that CDC could be well applicable for manoeuvres such as approach and landing.

### 3.1.3.4 RTMaps

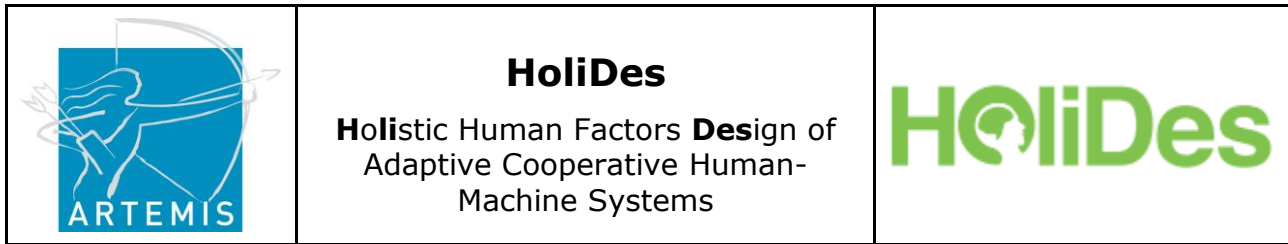
RTMaps (developed by INT) is applied as data integration platform that assures multiple data providers (sensors) are synchronized before being analysed. RTMaps assure that related patterns are interpreted together, which is critical for PPC, i.e. EEG data and eye-tracker data, and CDD, i.e. video streams from multiple cameras.

## 3.1.4 Integration

Details of integration are described in previous deliverable D7.7 – Tailored HF-RTP for Aeronautics. The status is summarized in Table 1.

**Table 1: Integration status of tools associated with adaptation in Diversion Assistant use-case**

	<b>AdCoS Integration</b>	<b>HF-RTP Integration</b>	<b>Plans</b>
<b>MED</b>	Fully integrated using a dedicated protocol and tested	Stand-alone	HF-RTP integration using OSLC protocol
<b>PPC</b>	Ex-post data processing, no real-time connection	Stand-alone, integrated via files in "csv" format (off-line)	Real-time integration with AdCoS
<b>CDC</b>	Ex-post data processing on specific	Stand-alone	Validation of applicability



scenarios, no real-time connection for diversion scenarios

### 3.1.5 Results of Proofs of concept

Details of integration will be described in next deliverable D7.9 – Empirical Evaluation of Aeronautics AdCoS. The status is summarized in Table 2.

**Table 2: Validation status of tools associated with adaptation in Diversion Assistant use-case**

Status	Plans	
<b>MED</b>	Validated in experiment with pilots	
<b>PPC</b>	Validated in non-aeronautics tasks. PPC can be used only as subject-specific classifier with acceptable performance (accuracy of more than 85%).	Subject-specific classification accuracy will be determined in aeronautics tasks.
<b>CDC</b>	Validated in approach scenarios with mediocre accuracy due to insufficiently designed experiment trying to mimic as much as possible automotive experimental design.	Better design of experiment will re-assess accuracy for approach scenarios and for diversion scenarios.

### 3.1.6 Discussion & Perspectives

The adaptation strategy with respect to missed event detection has been implemented to its final state for Diversion Assistant. The integrated AdCoS was validated with pilots in cockpit and its performance was acceptable. As a possible future step, it was proposed to integrate MED in HF-RTP by implementing OSLC protocol.

PPC failed to provide a generic classification tool for any user. Instead, PPC was shown to perform well as a user-specific classifier. Therefore, its integration in Diversion Assistant remains unfinished and the next step will be to verify the performance of PPC on aeronautics tasks.

The background for application of automotive CDC in aeronautics was investigated and CDC was applied in simplified tests in aircraft simulator. The tests indicate at least partial applicability of the tool for aeronautics and the tool will be further investigated in upcoming validations. Due to constraints in applicability to diversion scenarios it is not expected to integrate CDC tool with Diversion Assistant within the timeframe of HoliDes, however its future applicability will be fully understood.



## HoliDes

Holistic Human Factors Design of  
Adaptive Cooperative Human-  
Machine Systems

# HoliDes

### 3.2 UC3, Command and Control Room

The MTT KNIME framework supports the AdCoS Command and Control Room in terms to give the supervisor a more efficient visibility of the operators' absent times and in addition to indicate potential unusual behaviour. Such unusual behaviour could mean a potential risk regarding the border surveillance.



**Figure 30: Border Control Room**

Concerning context assessment IR sensors which recognize the operators' attendance, will be used. An operators' absence at its workplace will be detected by IR sensors which will then start a time measurement until the operator returns to its working place. The measurement will be stored as dataset into the data storage. The KNIME framework reads out the data storage and applies different design patterns to detect unusual absences. The design patterns could distinguish in time related patterns and correlation pattern. The time patterns investigate the data for regular occurring events and could split into different time intervals like month, week, day, hour, etc. The correlation pattern detects unusual behaviour for two or more operators at the same time. The communication aspect is implemented in form of tables and charts to support the control room management to analyse the absences. Based on the results the management draws the consequences for instance to increase the operator's awareness.

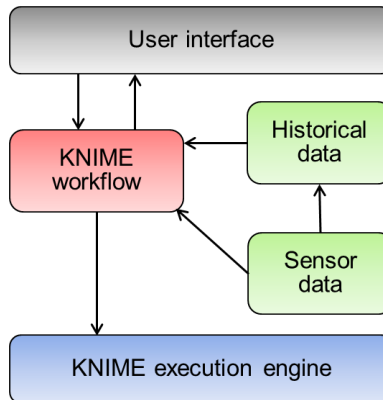
#### 3.2.1 Data flow full treatment chain

Figure 31 gives a general overview of the core elements and data flow regarding the MTT in combination with the AdCoS Border Control Room.



# HoliDes

Holistic Human Factors Design of Adaptive Cooperative Human-Machine Systems

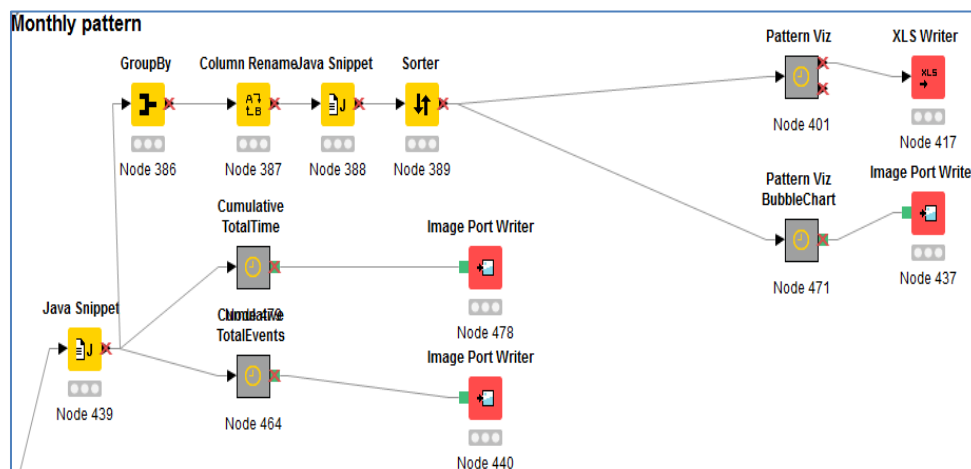


**Figure 31: KNIME Framework**

The user interface allows executing an investigation for unusual behaviour. The user can choose between patterns and adjust further execution parameters. The parameters are listed in Table 4.

KNIME workflow was developed with the KNIME Analytics Platform (see 3.2.3) and includes a set of the different functionalities for data transformation, machine learning and visualization.

The block historical data represents data storage to store sensor data. It could be a text file or a table in a relational database. The block Sensor data represents a current or latest dataset. For the most of the implemented patterns the historical data serves as reference for the current sensor data.



**Figure 32: Part of a KNIME workflow**

Finally the KNIME execution engine is the runtime environment required to execute a KNIME workflow.

### 3.2.2 Inputs & outputs

The incoming data to the KNIME workflow from the data blocks (see Figure 31) contains information about absent times of the different operators. Table 3 lists all relevant data fields, the related datatypes and their meaning.

**Table 3: Log data structure**

Parameter	Datatype	Description
OpId	Number	Unique ID of an operator
AbsenceStartTime	Date	Date and time when the operator was absent
AbsenceEndTime	Date	Date and time when the operator was back
WeekDay	Text	Weekday derived from the parameter AbsenceStartTime
Day	Number	Day derived from the parameter AbsenceStartTime
Month	Text	Month derived from the parameter AbsenceStartTime
Year	Number	Year derived from the parameter AbsenceStartTime
StartHour	Number	Hour derived from the parameter AbsenceStartTime
StartMinutes	Number	Minute derived from the parameter AbsenceStartTime
StartSeconds	Number	Second derived from the parameter AbsenceStartTime
EndHour	Number	Hour derived from the parameter AbsenceEndTime
EndMinutes	Number	Minute derived from the parameter AbsenceEndTime
EndSeconds	Number	Second derived from the parameter AbsenceEndTime
TotalMinutes	Number	Total absence time in minutes

Table 4 shows the practicable input parameter for the KNIME Framework.

**Table 4: KNIME Framework execution input parameter**

Parameter	Datatype	Description
startDate	String	Starting point for the specific time span to investigate. The date pattern must be compliant with dd/MM/yy
startTime	String	Optional parameter to set a time for the start date
endDate	String	End point for the specific time span to investigate. The date pattern must be compliant with dd/MM/yy
endTime	String	Optional parameter to set a time for the end date
timeBasedParam	Number	Is only required if the user select a time based pattern to set the frequency
threshold	Number	Determines the lower bound for recurrences. All values below the threshold will not pass as result to user interface
resultOutputPath	String	Path to store results in form of excel files or image files with the result related graphs
db	String	Contains information about the location of the data source.

### 3.2.3 Tools used

For the development of the Knime framework in WP3 the Knime Analytics Platform (see Figure 33) has been used.

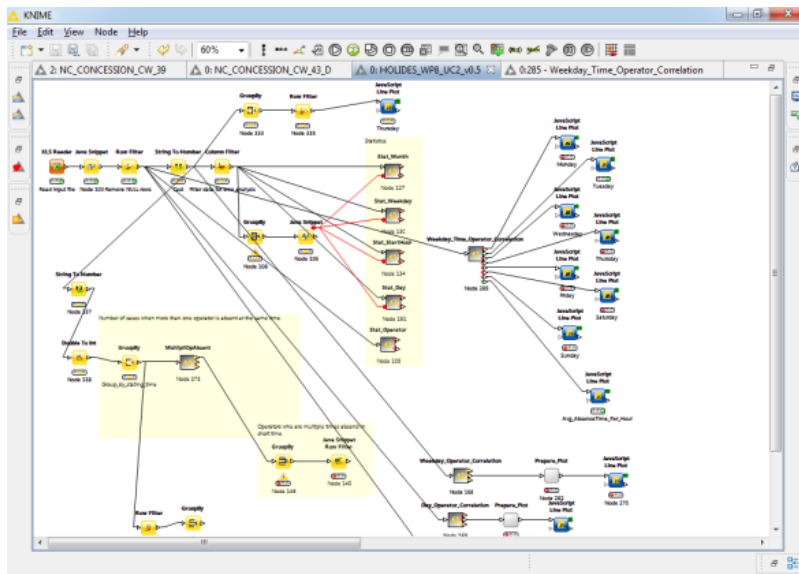


# HoliDes

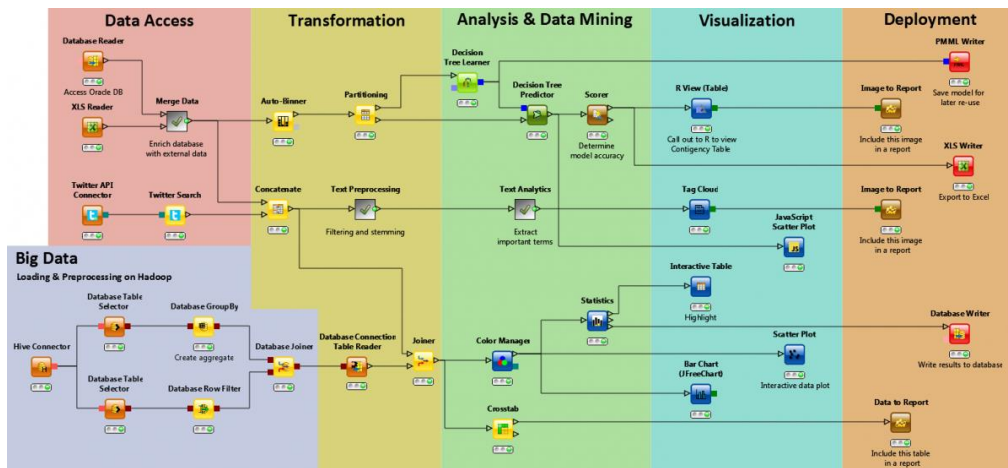
## Holistic Human Factors Design of Adaptive Cooperative Human-Machine Systems



Knime Analytics Platform is a Java based open platform for data analytics and is released under the General Public License (GPL), version 3. Knime Analytics Platform includes several components for data-transformation, data-processing, data-analysing, data-exploring and data-visualization (see Figure 34). It allows using a large set of routines, called nodes, to develop a data-driven workflow to investigate incoming data for identifying potential design patterns and detecting unknown anomalies within the data.



**Figure 33: KNIME Analytics Platform**

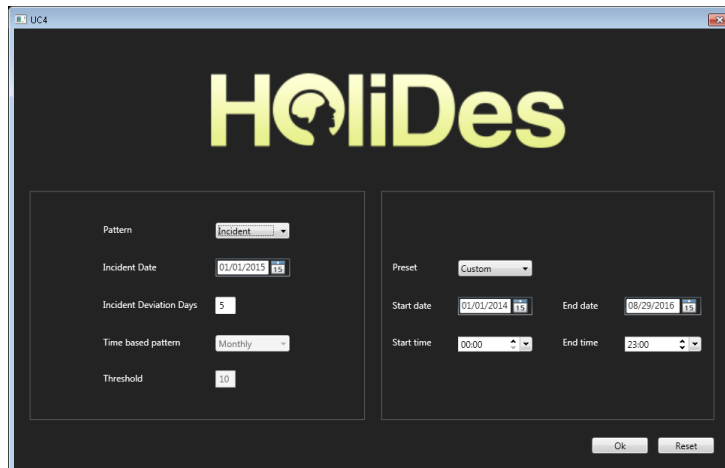


**Figure 34: Overview of different KNIME nodes**

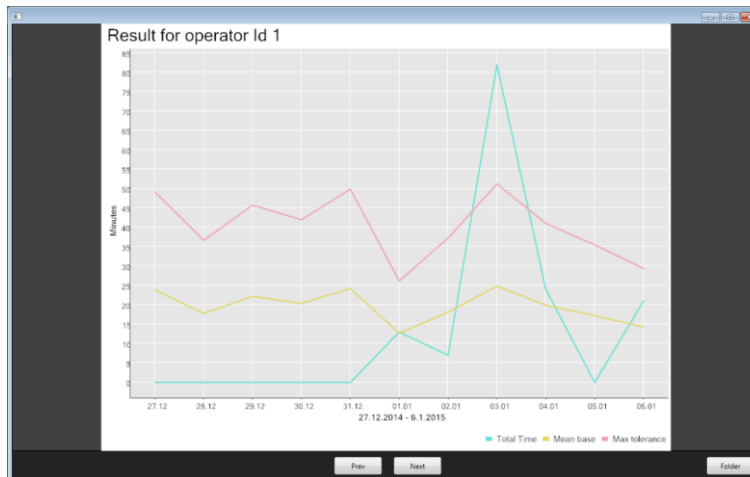


### 3.2.4 Integration

An overview of the general integration is illustrated in Figure 31. The user interface was developed by the AdCoS provider EAD-DE-CAS with Microsoft .NET framework. It calls the KNIME workflows and gets informed when the results from KNIME workflow finished and the results are available. The user interface reads and displays the generated documents as result (see Figure 36) from the executed operation.



**Figure 35: User interface to set execution parameters**



**Figure 36: User interface to display result**

### 3.2.5 Results of Proof of concept

For proofing the correctness outcomes from the KNIME workflow it was used test data with hidden patterns. The objective was to find out if the KNIME Framework



was able to confirm these patterns or disprove them. Table 5 shows an extract of the hidden patterns.

**Table 5: Extract from hidden patterns**

ID	Description of known design pattern	Design pattern classification
1	Operator absents every second Tuesday 3-4AM	Relative Monthly Pattern
2	Operators 17 and 19 absent together 4 times in a month at same time	Correlation Pattern
3	Operator absents 15 <sup>th</sup> day of each month	Absolute Monthly Pattern
4	Operators absent 10 to 10:30, 12:00 to 13:00 and 16:00 to 16:30	Relative Daily Pattern

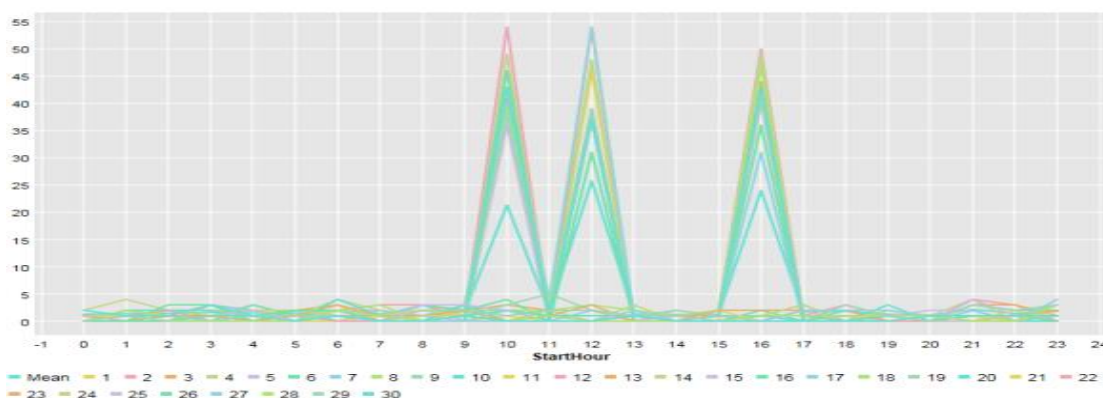
For each of this hidden pattern the KNIME Framework produced an output to support user (in most of the cases it will be the supervisor) to recognize these pattern.

### Hidden pattern to proof: #1

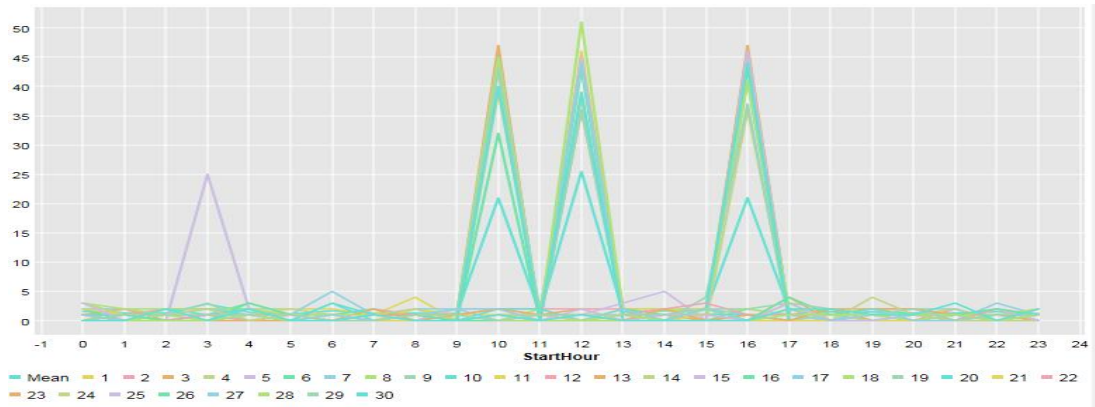
Operator's absence every second Tuesday 3-4AM

The following pictures show the outcome of the KNIME Framework regarding the regular weekday pattern. The horizontal axis represents the hours from 0 to 23. The vertical axis represents the number of absences and the different lines represent the operators. Figure 37 shows the Friday result and serves as reference for a normal behaviour. Figure 38 demonstrates the Tuesday result and represents an unusual behaviour due to the high peak at 3 am caused by operator 5.

**Result:** The outcome from the KNIME Framework confirms the known design pattern namely that an operator is absent every Tuesday between 3 and 4 am.



**Figure 37: Number of absence on Fridays**



**Figure 38: Number of absence on Tuesdays**

**Hidden pattern to proof: #2**

Operator 17 and 19 are both absent together 4 times in a month at same time

Figure 39 shows a KNIME Framework generated table with the three following columns:

- **Frequency** informs about the number of absences
- **Number of operators** who are absent at the same time
- **Operators** contains the IDs of those operators who are absent at the same time

The highlighted row shows that in given time period operator 17 and 19 are both 105 times absent in parallel.

The second largest frequency value is 9 and means a large gap between the two values which leads to the assumption that it could be an unusual behaviour.

**Result:** The following table does not confirm that the operators are absent 4 times in a month but the KNIME Framework result shows that the mentioned operators are quite often absent together. To proof the time constraint the function of the KNIME Framework has to be extended.



# HoliDes

## Holistic Human Factors Design of Adaptive Cooperative Human-Machine Systems



Sorted Table - 0:273:278 - Sorter

File

Table "default" - Rows: 1055 | Spec - Columns: 3 | Properties | Flow Variables

Row ID	Frequency	Number of operator	Operators
Row941	1	3	[7,1,18]
Row942	1	3	[7,10,14]
Row948	1	3	[7,16,8]
Row950	1	3	[7,17,9]
Row951	1	3	[7,18,24]
Row955	1	3	[7,21,18]
Row958	1	3	[7,23,19]
Row959	1	3	[7,23,1]
Row963	1	3	[7,25,25]
Row967	1	3	[7,30,7]
Row972	1	3	[7,6,10]
Row976	1	3	[7,8,30]
Row978	1	3	[7,9,10]
Row983	1	3	[8,10,8]
Row987	1	3	[8,14,4]
Row994	1	3	[8,19,15]
Row997	1	3	[8,21,7]
Row998	1	3	[8,22,14]
Row999	1	3	[8,23,12]
Row1004	1	3	[8,28,12]
Row1008	1	3	[8,30,20]
Row1009	1	3	[8,30,21]
Row1013	1	3	[8,5,5]
Row1019	1	3	[9,10,3]
Row1021	1	3	[9,15,22]
Row1023	1	3	[9,16,5]
Row1032	1	3	[9,23,18]
Row1034	1	3	[9,24,17]
Row1040	1	3	[9,27,3]
Row1042	1	3	[9,28,6]
Row1047	1	3	[9,5,29]
Row1050	1	3	[9,6,30]
Row1052	1	3	[9,8,13]
Row300	105	2	[17,19]
Row420	9	2	[2,2]
Row966	8	2	[7,2]
Row789	7	2	[3,3]
Row1028	7	2	[9,1]
Row127	6	2	[12,18]

**Figure 39: Result of KNIME Framework correlation pattern**

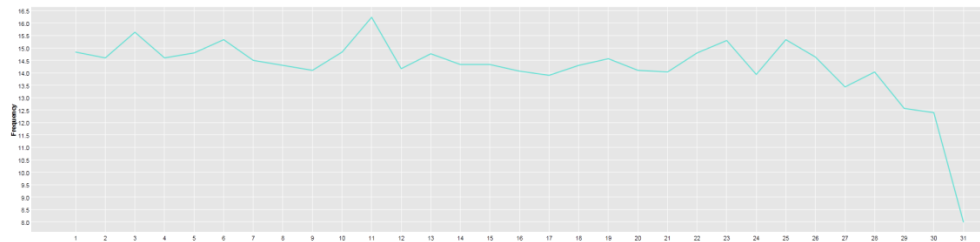
### Hidden pattern to proof: #3

Every 15<sup>th</sup> of each month operators are absent

The following picture shows the mean value for number of absences per day in a given time period of two years. The labels in the horizontal axis represent the days per month from 1 to 31 and the labels in the vertical axis shows the number of absence.

In most cases the mean value is between 14 and 15.5, whereas the max value is on the 11<sup>th</sup> with 16.3. The min value is at the end of the month because only every other month has 31 days.

**Result:** The given design pattern cannot be confirmed by the KNIME Framework. The 15<sup>th</sup> each month doesn't show unusual value. Not even the 11<sup>th</sup> with the max value could be considered as an unusual value.



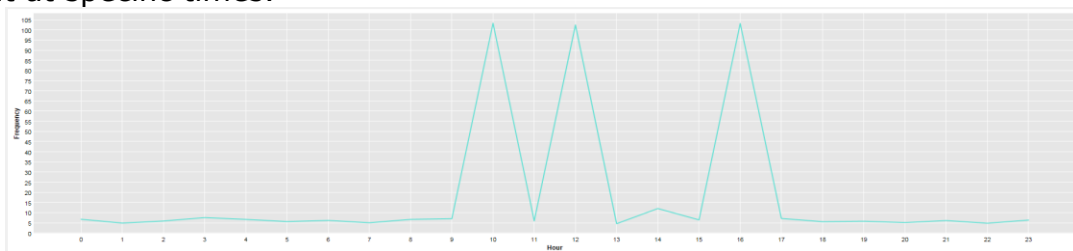
**Figure 40: Mean number of absence per day**

**Hidden pattern to proof: #4**

Operators are absent from 10 to 10:30, from 12:00 to 13:00 and from 16:00 to 16:30

Figure 41 shows the mean value for the number of absences per hour in a given time period of two years. The horizontal axis represents hours from 0 to 23. The vertical axis represents the number of absence. Obviously the peaks at 10:00, 12:00 and 16:00 depict the working breaks.

**Result:** KNIME Framework result approves the pattern that the operators are absent at specific times.



**Figure 41: Mean number of absence per hour**

**3.2.6 Discussion & Perspectives**

In the WP3 it was developed MTT KNIME Framework to detect unusual behaviour within recorded log data. The MTT was implemented in the AdCoS Border Control Room in WP8. Together with the AdCoS responsible person it was tailored the MTT to the specific needs.

In general the implementation in a test environment was quite meaningful and proofed that the MTT provides the expected results. In terms of performance it had been point out that used tool (see 3.2.3) wouldn't be completely sufficient. The KNIME Analytics Platform fulfil the needs for develop the workflows but the underlying KNIME execution engine in the non-commercial version seems to be limited. A switch to a commercial product version could mitigate this drawback.

### 3.3 UC4, Overtaking including lane change assistant

UC4 overtaking including lane change assistant uses several WP3 tools for the adaptive automation/ assistance part.

#### 3.3.1 AdCoS based on Cognitive Distraction Classifier and CONFORM

For the Adapted Automation AdCoS two MTTs were considered: CDC and CONFORM. Both MTTs are used to adapt the driving style of the automated vehicle.

##### 3.3.1.1 Data flow full treatment chain

###### CONFORM

The data flow remained identical compared to the previous deliverables.

###### Cognitive Distraction Classifier

The entire signal processing chain has been integrated in an RTMaps diagram.

Video images of the driver's face are recorded using two web cameras, one positioned behind the steering wheel, another one next to the rear view mirror. The video stream passes several processing stages.

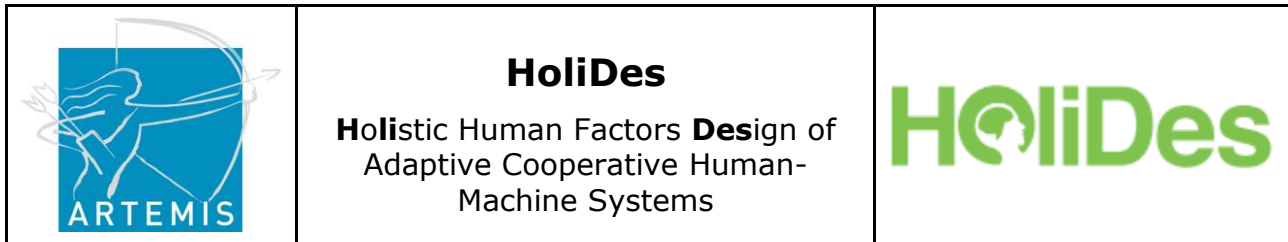
First, facial elements are located and tracked using the Intraface software:

([www.humansensing.cs.cmu.edu/intraface](http://www.humansensing.cs.cmu.edu/intraface); *intraface\_tracker\_1*, *intraface\_tracker\_2*).

Intraface marks these facial elements with dots and yields their coordinates as output.

These coordinates enable us to calculate facial features associated with cognitive distraction. From vehicle kinematics and control data features are evaluated. These calculations and all remaining processing steps described here are carried out in the *distraction detector* component.

Video and vehicle data streams are recorded as a series of video and vehicle data frames, respectively. During acquisition, each frame is labelled with a timestamp, allowing for synchronization of the three data streams. Features calculated from a particular video or vehicle data frame are associated to the timestamp of that frame. If a feature is calculated from a series of past frames, the feature value is



associated with the timestamp of the most recent frame in the series. In this way, features are synchronized into a joint stream of feature frames.

Besides video and vehicle data, we also investigated the use of audio data of the participant's voice as well as eye-tracking data. Advanced signal analysis showed that all these types of data may contribute to a higher accuracy of cognitive distraction detection. For the first version of the CDC, however, the focus was set on facial video and behavioural driving data. Machine learning methods have been developed to classify these data offline. The first online implementation of the CDC, suitable for near-to-real-time use, is developed to use facial video data. The framework allows the other types of data (e.g., eye-tracking) to be included in future developments.

A supervised learning algorithm is used to associate each feature frame with a level of distraction. The term "level of distraction" is explained in the following:

During experiments in a driving simulator, distraction is created by assigning secondary tasks of various levels of difficulty to the driver. There are numerous ways of accomplishing this goal. In an earlier phase of the project, we evaluated the use of mental arithmetic exercises as secondary tasks. We were able to both distract the driver and to recognize distraction during the machine learning classification phase (see D2.6). Recently, we optimized the experimental set-up for focusing on the recording of facial video data. To this end, we employed to the n-back task paradigm (see D5.6) because this paradigm allows for consistent facial movements between conditions for performing the task.

During supervised learning (training phase), the classifier is presented the synchronized features along with the current level of driver distraction that has been prepared in the experiment.

After sufficient training, the classifier is capable of mapping a given synchronized set of feature values to a distraction level with reasonable accuracy.

### **3.3.1.2 Inputs & outputs**

#### **Cognitive Distraction Classifier**

As mentioned above, video images of the driver's face as well as vehicle features serve as tool input.

The output is the distraction level classification value, which is an estimate of the driver distraction level (undistracted, slightly or strongly distracted). It is

accompanied by a quality measure, quantifying the estimation reliability between 0 (unreliable) and 1 (most reliable).

### CONFORM

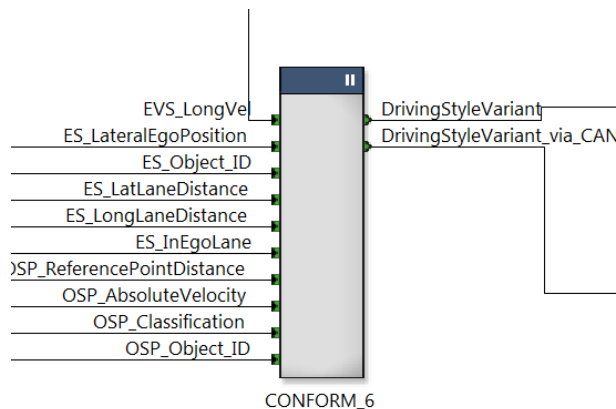
Model parameters are defined throughout the GUI. For the data input and output specification, it has to be distinguished between the online and the offline version.

#### Online version

The online version is currently connected to the automotive use case and the RTMaps framework. Figure 42 illustrates the inputs for the online version.

**Table 6: CONFORM RTMaps inputs**

Parameter	Description
EVS_LongVel	Longitudinal velocity of the ego vehicle
ES_LateralEgoPosition	Lateral deviation from the current lane
ES_LatLaneDistance	Lateral deviation from the current lane of the detected objects
ES_LongLaneDistance	Long distance between the ego vehicle and the detected objects in the current lane
ES_InEgoLane	Information if the object is in the same lane as the ego vehicle
OSP_ReferencePointDistance	Euclidian Distance to detected objects
OSP_AbsoluteVelocity	Absolute velocities of the detected objects
OSP_Classification	Classification of the objects, i.e. truck, car, person
OSP_Object_ID	ID of the object to track the object

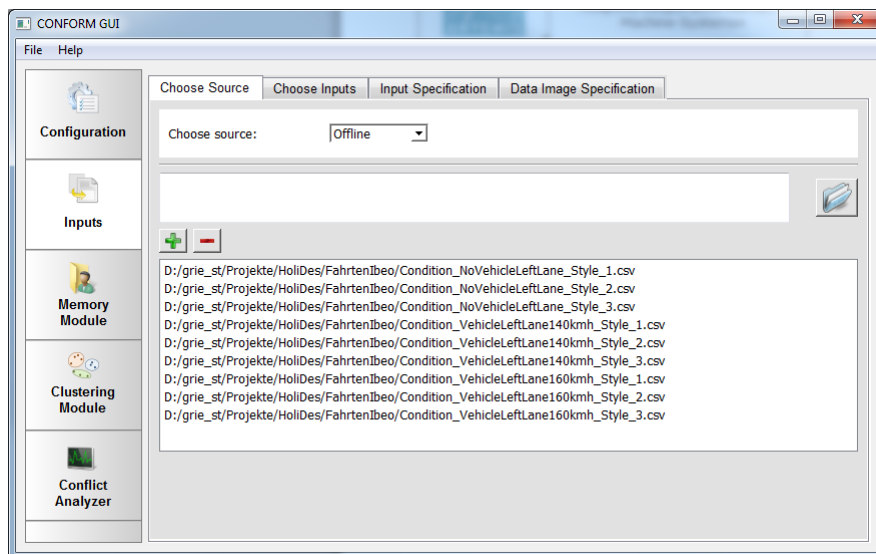


**Figure 42: RTMaps CONFORM inputs**

However it is fairly easy to adjust the RTMaps to other domains and inputs. This flexibility is already used for the offline version. The output is an integer for the predicted driving style of the automated vehicle. This integer is then forwarded via a can signal to the IAS test vehicle.

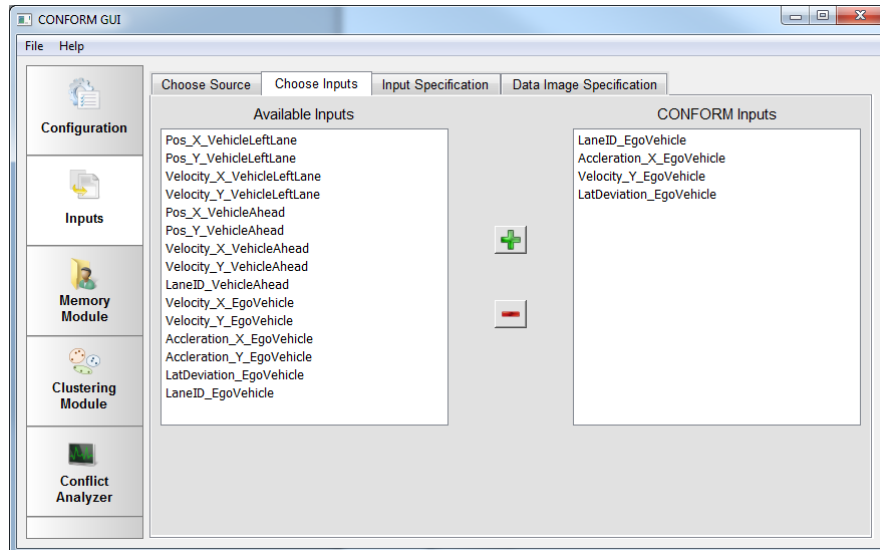
#### Offline version

For the offline version CONFORM considers data stored in csv files. Therefore all parameters named in the header of the csv file are available as possible inputs. The user can select via the CONFORM GUI the parameters of interest, as shown in Figure 44. The output depends on the selected modus. It is either the clustering of different operator behaviours defined through a set of csv files (see Figure 46 as an example), or the similarity between an operator behaviour and predefined operator behaviour clusters. The definition of clusters can be also provided through csv files.



**Figure 43: CONFORM GUI and choice of input source**





**Figure 44: CONFORM input choices in the offline mode. The available inputs depend on the considered csv file.**



### 3.3.1.3 Tools used

#### Cognitive Distraction Classifier

The RTMaps software (Intempora) has been used to integrate all stages in the signal processing chain. We used custom components to embed existing third party software as well as our own code.

Face detection and tracking is achieved by using the Intraface software ([www.humansensing.cs.cmu.edu/intraface](http://www.humansensing.cs.cmu.edu/intraface)), which we embedded in custom RTMaps C++ components (Blocks *intraface\_tracker\_1*, *intraface\_tracker\_2* in RTMaps diagram).

We implemented feature calculation, frame synchronisation, and machine learning in the R programming language for offline learning and classification. This code is mainly used to develop the CDC model (feature definition and machine learning) by studying the effects of algorithmic variations on model performance. During a later project phase, when the model reached sufficient maturity, capability to perform online (real-time) analysis became our next goal. Since the R-code was not embeddable in RTMaps, we re-implemented the algorithms in the Python programming language while satisfying real-time requirements and embedded our code in an RTMaps Python component (Block *distractionDetector* in RTMaps diagram).

	<p><b>HoliDes</b></p> <p><b>H</b>olistic Human Factors <b>D</b>esign of Adaptive Cooperative Human- Machine Systems</p>	
---	---	---

Experiments with test subjects in specific distracted driving scenarios were carried out using the OpenDS driving simulator software ([www.opens.de](http://www.opens.de)).

## **CONFORM**

The MTT CONFORM consists of two parts: The CONFORM Model and the CONFORM GUI. The CONFORM Model as shown in Figure 45 is responsible for all necessary calculations. The CONFORM Model itself is divided in a use case dependent part and a use case independent part. Both parts have been explained in depth in the previous deliverables; see for instance D3.3, D3.5 and D3.6. The CONFORM GUI allows to configure the use case dependent parts of the CONFROM Model and to visualize the output. Both, the CONFORM Model and the GUI form a stand-alone offline tool. This tool can be used to either:

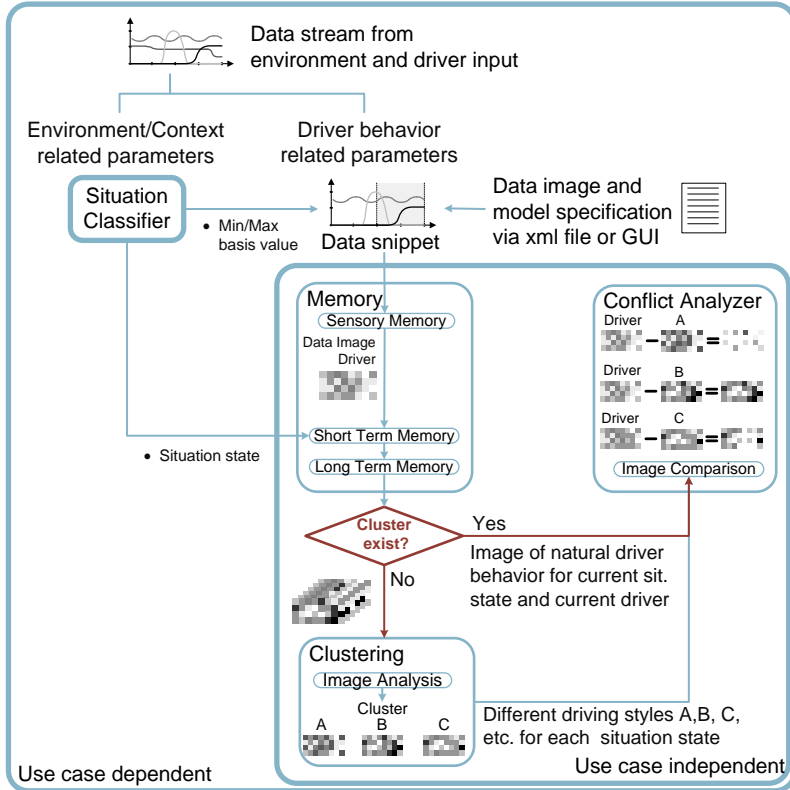
- Compare the similarity between different operator behaviours and to cluster similar operator behaviours
- Compare the similarity between the operator behaviour with predefined cluster and match the operator to one of the cluster
- Compare the similarity between the operator behaviour and an automation behaviour
- Learn the natural operator behaviour for a given context ( requires labelled data)

So far, the development effort relied on the CONFORM Model, for the last project cycle the focus was moved to development and improvement of the CONFORM GUI. The GUI was built using the QT framework. Figure 46 gives some first impressions about the interface. The CONFORM handbook provided in the document Annex II explains in detail the handling of the GUI.

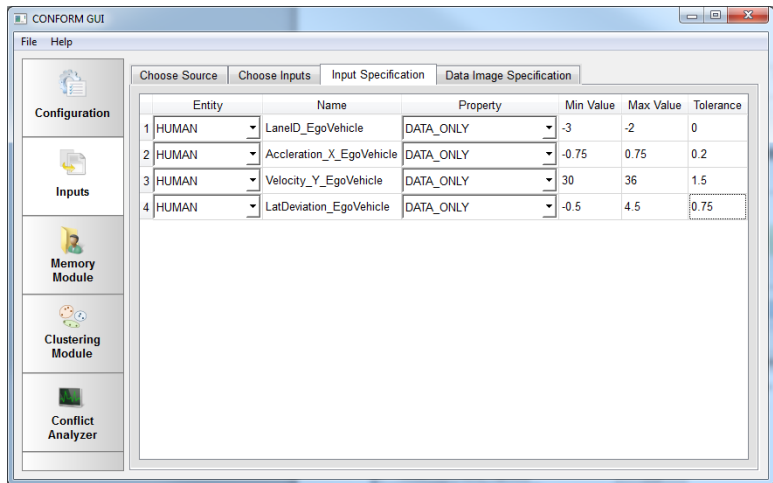


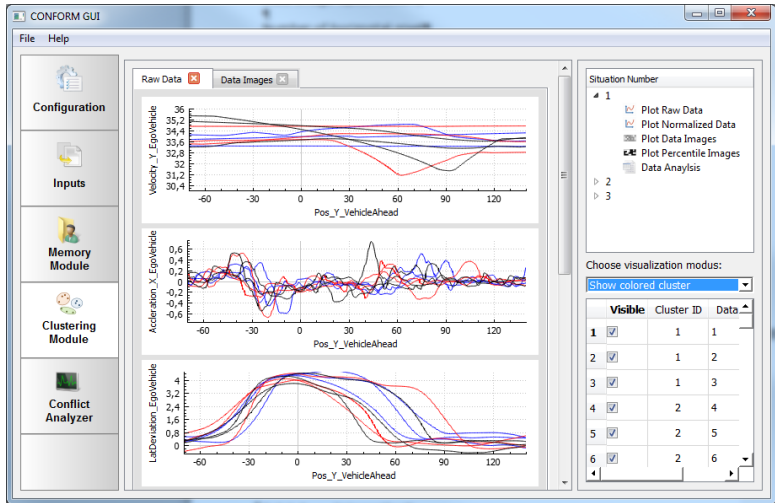
# HoliDes

## Holistic Human Factors Design of Adaptive Cooperative Human-Machine Systems



**Figure 45: Structure of the CONFORM Model**



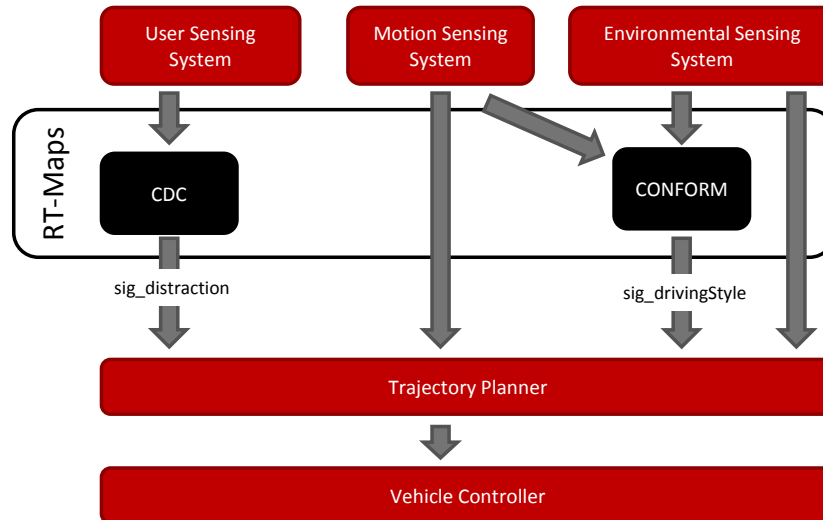


**Figure 46: Two screenshots of the CONFORM GUI**

### 3.3.1.4 Integration

The CDC and CONFORM have been integrated with the IAS autonomous driving system in the IAS Test Vehicle. Additionally, for the CDC cameras were positioned as described above and vehicle data are sent to the CDC via Ethernet.

The CDC output (estimated level of distraction and reliability value) and CONFORM output (predicted driving style) are sent from RTMaps to the vehicle CAN using a USB to CAN adapter and dedicated RTMaps CAN signal processing packages. A specification of the CAN signals can be found in D9.9. Figure 47 summarizes the integration of the MTTs CDC and CONFORM. Figure 48 shows the screenshots of the RTMaps diagrams for the MTT CONFORM. Experiments with a test driver are planned for late August.



**Figure 47: Integration of the MTTs CDC and CONFORM in the Adapted Automation AdCoS [D9.9]**

In a collaborative experiment with TAK, undertaken in the process towards integration with the TAK HMI, video, eye-tracking, and behavioural data were recorded for offline analysis for the CDC.

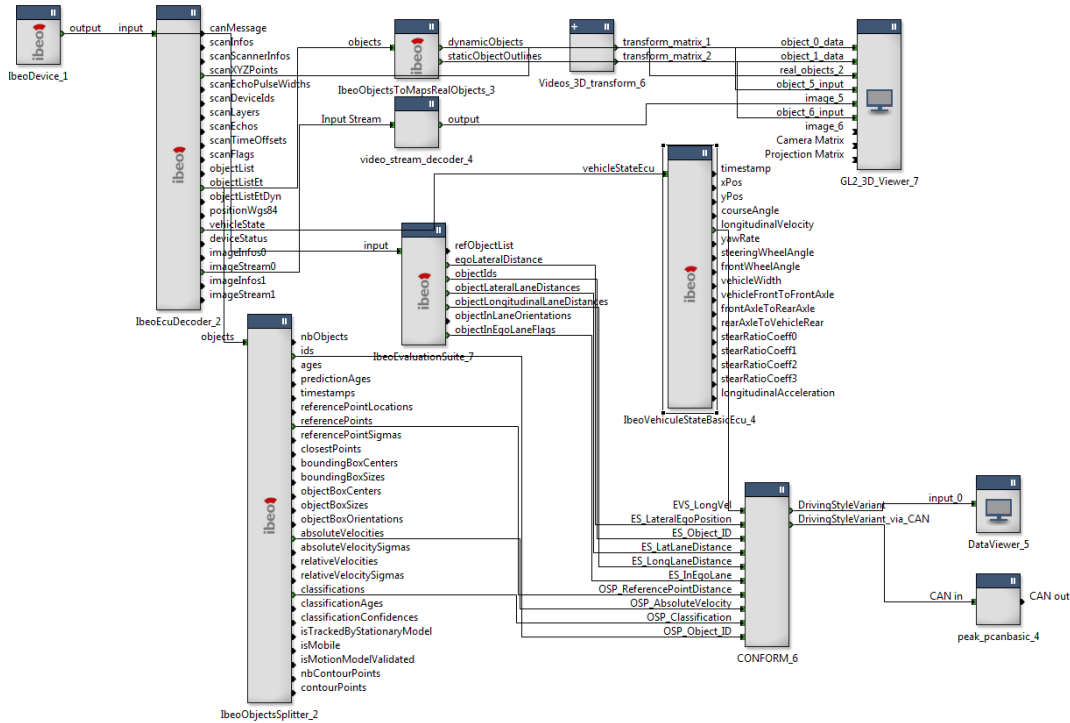
Additionally in WP7 with HON, we investigated the suitability of the CDC in the aviation domain. Offline analysis was performed on video data.

Both MTTs connect via the RTMaps framework to the IAS test vehicle to receive the inputs described about the current user state, vehicle state and environmental state. Both, CDC and CONFORM, send their outputs via CAN signals to the trajectory planner of the IAS test vehicle.



# HoliDes

**Holistic Human Factors Design of Adaptive Cooperative Human-Machine Systems**



**Figure 48: Integration of CONFORM in the RTMaps framework for the Adapted Automation AdCoS**

### 3.3.1.5 Results of Proof of concept

#### Cognitive Distraction Classifier

Cognitive distraction driving experiments with test subjects have been carried out in-house with 6-10 participants so far and at the TAK site with 40 participants, both times using driving simulators. Analysis results from our 2015 in-house data were presented in D3.6. We are currently in the process of analysing our data from 2016 as well as TAK data. Experimental data recorded by HON in an experiment in a cockpit simulator have been analysed recently.

In-house experiments have been performed using the OpenDS driving simulator and a minimal vehicle cockpit consisting of steering wheel, accelerator and brake pedal (Logitech G27). Each of the participants was given the task to drive closely to a pace car, while performing the secondary task (the n-back task Video and vehicle data were recorded during the experiments).

Results are quantified by the probability distribution of classifications (predicted distraction level) given the experimental condition (actual distraction level). An ideal

subject would yield a distribution with a probability of 100% for predicted distraction level = actual distraction level and 0 otherwise.

From our last in-house experiment, both online and offline classification results have been analysed so far.

For the subjects analysed here, true positive rates are above chance level (33%). Most of the true positive rates are well above 60%.

## CONFORM

The evaluation of CONFORM and its proof of concepts are documented in detail D3.6 and D9.9. For the current document we recall the main result from D9.9 in Table 7: The table summarizes the median values for the calculated standard measure (normed "Best-Worst-Score") of the evaluation approach "Best-Worst-Scaling". For the baseline, i.e. none adaptive, the driving style of the automated vehicle was identical for all drivers in the particular situations. For the AdCoS, i.e. driver and context adaptive, the driver's preferred driving style of the automated vehicle was predicted by CONFORM and consequently adapted. The AdCoS increased the appealing of the automation behaviour compared to the none-adaptive baseline. The increase was between 20% and 300% depending on the situation. Values above 50% can be interpreted as a clear benefit.

An interesting aspect of the evaluation of CONFORM is the quality of the prediction. Table 7 also lists the normed median values of the participant rating. An optimal prediction would generate similar values as the output from the participant rating. The prediction of CONFORM is on average between 25-33% below this optimum. On reason is clearly the implemented adaptation approach. Some drivers (24%) prefer an automated driving style different to their own individual driving style. In addition these drivers are distributed arbitrary over the different driving style cluster. This makes it really difficult for any machine learning approach. Thus an optimization of the prediction seems challenging and needs further investigation in the future.

**Table 7: Comparison of normed "Best-Worst-Score" for the baseline, the AdCoS and the actual participant rating in different situation [D9.9]**

	Baseline: Normed median "Best-Worst Score"	AdCoS: Normed median "Best- Worst Score"	Gain	Participant Rating Normed median "Best- Worst Score"	Gain to Baseline	Gain to AdCoS
Situation A	0.417	0.5	20%	0.667	60%	33%
Situation B	0.333	0.5	50%	0.667	100%	33%
Situation C	0.167	0.666	300%	0.833	400%	25%

### 3.3.1.6 Discussion & Perspectives

#### Cognitive Distraction Classifier

We worked on the development of a tool to measure cognitive distraction during driving: the cognitive distraction classifier (CDC). To measure cognitive distraction, we recorded multiple types of data in experiments triggering different levels of cognitive distraction while participants performed a driving task. The recorded data included: audio voice, facial video, behavioural driving, and more recently eye-tracking data. Advanced signal analysis showed that all these types of data may contribute to a higher accuracy of cognitive distraction detection. For the first version of the CDC, however, the focus was set on facial video and behavioural driving data.

During offline (post-experimental) analysis of facial video (and behavioural driving) data, we have been able to predict the level of distraction for all participants analysed so far at rates significantly above chance level .

Both feature pre-processing and machine learning can be tuned by various parameters. We will gain further insight by conducting a detailed study on the variation of relevant parameters and optimizing our classification analysis methods for online processing. Finally, as previous advanced signal analysis showed, extending the CDC with amongst other things, eye-tracking and audio data may significantly improve results.

We have shown that we can highly significantly detect cognitive distraction offline during a driving task, using facial video data only. Further improvements should be made to the online set-up of the CDC, so that higher accuracies may be obtained. We have developed the CDC capable of online analysis, with a framework capable of extending with different types of data. Next steps include continuation of collaboration with partners to integrate the CDC in AdCoS, so that the level of automation can adapt to the cognitive state of the driver, or a safety system can interact with the driver in an appropriate way.

#### CONFORM

The MTT CONFOM was successfully applied in the automotive domain as life and non-life cycle tool. Throughout CONFOM the automated vehicle could adapt its driving style towards the preferred driving style of the driver. The evaluation in WP5 and WP9 highlighted the benefit and the increasing appeal of the driver adaptive automated vehicle.





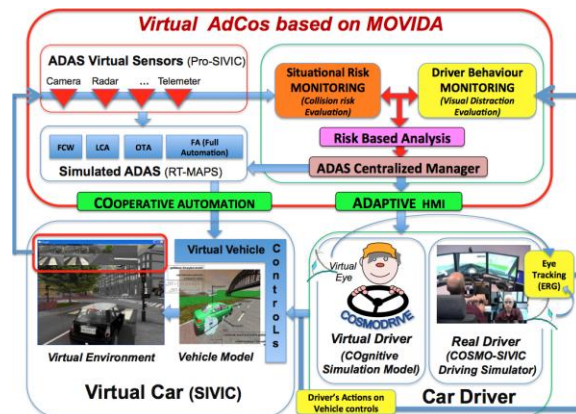
Even though the MTT CONFORM was tailored for automotive domain CONFORM is applicable for other domains as well. The CONFORM GUI is able to handle arbitrary data and to analyse it further. Future research and development on the MTT will concentrate on the cross domain capability.

### 3.3.2 AdCoS based on MOVIDA

In the frame of WP3, IFS was in charge, in partnership with CVT and INT, to virtually design, develop, and prototyping an AdCoS based on a set of monitoring functions named MOVIDA (for Monitoring of Visual Distraction and risks Assessment). Then, virtual evaluations (from WP4 validation methods) of MOVIDA-AdCoS were implemented on the V-HCD simulation platform (as an instance of the HF-RTP in WP9, based on COSMODRIVE virtual driver developed by IFS in WP2) to progressively assess and increase the MOVIDA-AdCoS efficiency and effectiveness, according to the end-users needs.

#### 3.3.2.1 Data flow full treatment chain

The AdCoS based on MOVIDA is an integrative co-piloting system supervising several simulated Advanced Driving Aid Systems (ADAS), according to the drivers' visual distraction status and to the situational risk assessment, to be managed by MOVIDA algorithms in an Adaptive and Cooperative way regarding the car driver's difficulties and needs (Figure 49).

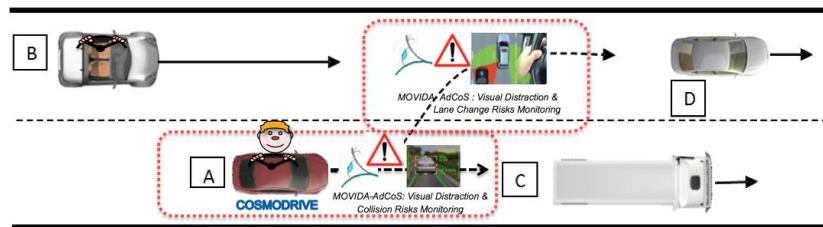


**Figure 49: Functional architecture of the AdCoS based on MOVIDA**

The main ADAS monitored in this MOVIDA-AdCoS, that are simulated with RTMAPS and Pro-SIVIC tools, are a Collision Avoidance Systems (like FCW, for Forward Collision Warning) and a Lane Change Assistant (LCA, including an Over-Taking

Assistant, OTA), and Full Automation devices (FA) taking the control of the car in case of emergency situations and/or inadequate behaviour of the car driver.

MOVIDA is more specifically in charge to support a car driver for the main driving scenario presented in Figure 50 (i.e. that is the common “use cases of reference” shared with other WP9 partners). This scenario may occur when driving on a two-lanes Inter-Urban Highway (limited to 90 km/h).



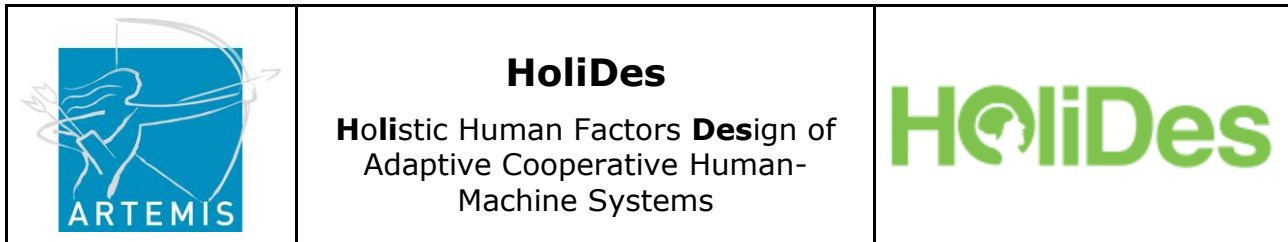
**Figure 50: Driving Scenarios and Use cases for the MOVIDA-AdCoS**

In this driving context, MOVIDA was designed in order to support the driver in Car A. Regarding this driver, the aim is to assist him/her in an adaptive and cooperative way in case of a critical event occurring in the road environment (e.g. emergency braking of the truck C) and/or due to dangerous visual distraction. If this occurs, the aim of MOVIDA-AdCoS is to support the driver in Car A by managing the frontal collision risk with the truck C and/or by helping them in implementing (or not) a safe Lane Change manoeuvre for avoiding any lateral collision risks with the car B.

In this traffic situation, MOVIDA have thus to observe and monitor the car A driver’s behaviours (simulated with COSMODRIVE model) in order to diagnose critical visual distraction and / or potential risky manoeuvres regarding the external events and the situational risk (e.g. intention to implement a lane change at a critical time), and then to adapt the driving aids in an adaptive and cooperative way to support the driver in car A, via information delivery, warning systems to alert the driver, or by activating vehicle automation functions taking the control of the car to avoid the accident.

### 3.3.2.2 Inputs & outputs

MOVIDA-AdCoS Inputs are of two main types. On the one side, they are based on the analysis of the external driving situation as perceived by the car sensors (simulated with Pro-SIVIC software). From the other side, Car A driver’s activity is also monitored by considering their visual scanning or distraction status (simulated with COMSODRIVE or collected among real drivers by eye tracking systems, cf. detailed description in D2.7), and by analysing their driving behaviours (i.e. the



actions currently implemented by the driver/COSMODRIVE on vehicle pedals and steering wheel) collected on the car (simulated on the V-HCD platform by a Pro-SIVIC virtual car).

Then, at the decisional level of MOVIDA, a set of risk-based analysis algorithms are implemented in order to evaluate the distraction risk and to assess the adequacy of the behaviours implemented by the driver according to the external risk of collision with other vehicles (i.e. Truck C regarding frontal collision, and car B regarding Lane Change manoeuvre). Synthetically, these risk-based algorithms consider frontal and lateral Inter-Vehicular Time (IVT) and/or Time To Collision (TTC) values collected from the car sensor of MOVIDA. In case of critical IVT and/or TTC values (like low values or high drop of these values during the last second, for instance), the current fixation point of the COSMODRIVE/driver's eyes is considered. Then, in case of visual distraction or inadequate visual scanning, meaning a potential unawareness of the critical events (e.g. braking of the followed truck or no detection of an approaching car liable to be observed in the left mirror), the car A driving behaviours are assessed as "inadequate" by MOVIDA algorithms, and a diagnosis value of "critical situation" is provided to the Centralized Manager of ADAS. At this level, another set of decision rules are implemented in order to determine which kind of the 2 main ADAS integrated in the MOVIDA-AdCoS, i.e. Frontal Collision Avoidance system (i.e. FCA) and an Lane Change Assistant (i.e. LCA) is able to support the driver in the current context, and how this driving aids have to interact with the Car A driver according to his/her visual distraction status.

Finally, regarding MOVIDA-AdCoS outputs, two core sub-modules are in charge to manage interactions with the human driver (in car A): (a) the "Adaptive HMI manager" has to adapt HMI modalities of information delivery and warning signals (Visual and Auditory) in accordance with the driver visual distraction status, and (b) The "Cooperative Automation" support system has to take the (Partial or Full) control of the car to implement an automatic Braking or Lane Keeping, in case of behavioural errors (e.g. dangerous lane change manoeuvre implemented by the driver), or when the criticality of the situation (i.e. imminent risk of collision with front or lateral vehicles) is assessed as too high for being well-managed by a human driver.

Regarding its Human-Machine Interaction modalities, MOVIDA-AdCoS is liable to interact with the Car A driver from 3 main modalities: Visual Information delivery, Visual and Auditory Warning (both controlled by the "Adaptive HMI manager"), or vehicle control taking abilities (implemented by the "Cooperative Automation" support system) via partial (i.e. lateral or longitudinal control) or Full Automation (i.e. combining both lateral and longitudinal control of the car).

Visual Pictograms used in the MOVIDA-HMI to support the driver while changing of lane or to avoid frontal collision are based on HOLIDES partners' proposals (i.e. REL and CRF demonstrator), as presented and discussed in D9.3 (p. 58), and replicated in the following figures.

The first one (Figure 51) is used to inform a non-distracted driver that the Lane Change Manoeuvre is required (i.e. when the truck C is braking, for instance) and possible in the current traffic situation (i.e. No car is approaching on the left lane). This visual information is delivered on a visual display implanted at the centre of the dashboard of the car, as presented in Figure 51.





**Figure 51: pictogram delivered by MOVIDA (on the in-vehicle display) to inform a non-distracted driver that the Lane Change Manoeuvre is possible**

However, when the driver is initially visually distracted, another pictogram is used (delivered in association with an auditory warning, in order to manage the visual distraction risk) for informing the driver that a Lane Change Manoeuvre is required and may be implemented in the current traffic situation (i.e. No car is overtaking or approaching on the left lane). This pictogram is presented in Figure 52.



**Figure 52: pictogram delivered by MOVIDA to warn a distracted driver that a Lane Change Manoeuvre is required and possible**

By contrast, when the left lane is not free (i.e. the Car B is currently approaching or overtaking the car A), another pictogram is delivered (in association with an

	<p style="text-align: center;"><b>HoliDes</b> <b>H</b>olistic Human Factors <b>D</b>esign of Adaptive Cooperative Human- Machine Systems</p>	
--	--	---

auditory warning) in order to warn the driver about the dangerousness of a Lane Change, and to invite him/her to keep his/her current lane (Figure 53).



**Figure 53: pictogram delivered by MOVIDA to warn the driver that that a Lane Change Manoeuvre is not possible**

Regarding the Frontal Collision risk, or to support the driver in maintaining a safe following distance with the truck C, the pictogram presented in Figure 54 is delivered to the driver in case of a collision risk detected by MOVIDA (when the truck is braking, for instance). To support a distracted driver, this pictogram is also delivered with an auditory warning.



**Figure 54: Pictogram used by the Collision Warning System of MOVIDA**

Moreover, 3 additional pictograms were designed to inform the driver about the different modalities of MOVIDA regarding vehicle control taking and automatic manoeuvres (these pictograms are also adapted from pictograms designed by other HoliDes partners for REL & CRF demonstrator, as presented and discussed in D9.9; from p. 11 to 16).

In case of a high risk of frontal collision detected (from critical values of TTC and/or IVT with the truck collected by car sensors of the FCA ADAS) and assessed by MOVIDA as not manageable by the human driver (due to driver's distraction or to the high emergency of the situation), an automatic braking is implemented by the AdCoS, and the pictogram presented in Figure 55 is presented (in association with an auditory warning) to inform the driver about the "active status" of the MOVIDA (i.e. longitudinal control under the responsibility of the driving Aid).



**Figure 55: pictogram delivered by MOVIDA to inform the driver about the automatic “Emergency Braking”, when implemented by the AdCoS**

Moreover, when the driver started to implement a lane change manoeuvre (by handling the blinkers and by turning the steering wheel of the left, for instance) while the left lane is not free (i.e. Car B is approaching), the vehicle automation functions of MOVIDA inhibits humans’ action and warn them about their errors. To alert the driver about the dangerousness of a Lane Change and to inform him/her of the automatic control taking to keep the car in the current lane, the following pictogram (Figure 56) is activated, in association with an auditory warning.



**Figure 56: pictogram delivered by MOVIDA to inform the driver about the “Lane Keeping” automatic manoeuvre, when implemented by the AdCoS**

Finally, in case of both high risk of Frontal Collision with the truck C and critical risk of Lateral collision with the car B (in case of lane change of car A), MOVIDA takes the full control of the car by both (1) keeping the car A in its lane and (2) by implementing an automatic braking manoeuvre. In this context, the pictogram presented in Figure 57 is delivered to the driver, in association with an auditory warning.



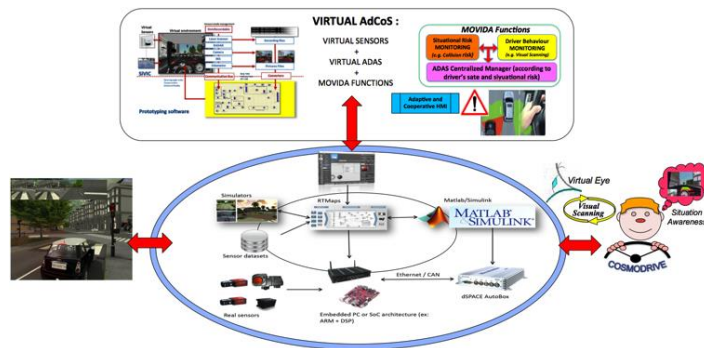
**Figure 57: pictogram use by MOVIDA to inform the driver about the “Full Automation” status of the AdCoS (i.e. automatic Lane Keeping and Braking)**



**3.3.2.3 Tools used**

To support the virtual design, prototyping and then evaluation of this MOVIDA-AdCoS, a “Virtual Human Centred Design platform” (so-called V-HCD; cf. Figure 58) has been jointly developed by IFS, CVT and INT, as an example of a tailored HF-RTP based on RTMaps software specifically dedicated to dynamic simulations of virtual AdCoS (see detailed description in D4.5 and D4.7). In addition, this V-HCD integrative platform will be also one of the WP9 simulation Demonstrators (D9.6). All the ADAS sub-systems managed by the MOVIDA functions have been interfaced through RTMaps, in order to support the virtual prototyping and dynamic simulations of this AdCoS, when using by a human driver, as simulated with COSMODRIVE model.

In its final status, the V-HCD integrates 4 main HoliDes MTTs: (1) a COgnitive Simulation MODEL of the car DRIVER (named COSMODRIVE) able to visually explore the road environment from a “virtual eye” and to drive (2) a virtual car simulated with Pro-SIVIC (3) equipped with the virtual MOVIDA-AdCoS (simulated with RTMaps and Pro-SIVIC), for dynamically progressing in (4) a virtual 3-D road environment (simulated with Pro-SIVIC). According to the HoliDes “HF-RTP” logic, COSMODRIVE plays the role the “Human Factor” (HF) component interacting with a virtual AdCoS, also simulated on the HF-RTP.



**Figure 58: Overview of the V-HCD platform, as an example of a tailored HF-RTP based on RTMaps for automotive application**

From this HF-based virtual design approach supported by the V-HCD integrative platform in WP4, it is expected to better integrate end-users’ needs since the earliest steps of the AdCoS design process. In this human centred design approach, the V-HCD platform was used for generating dynamic simulations of different driving scenarios (more or less critical), when a virtual driver (simulated with COSMODRIVE), distracted or not, was driving a virtual car equipped with MOVIDA-AdCoS. From this simulation, it was possible to virtually generate and the evaluate the functioning of all the components of this AdCoS (i.e. ADAS sub-systems and MOVIDA algorithms), as well as its inputs (from data flows collected by the car

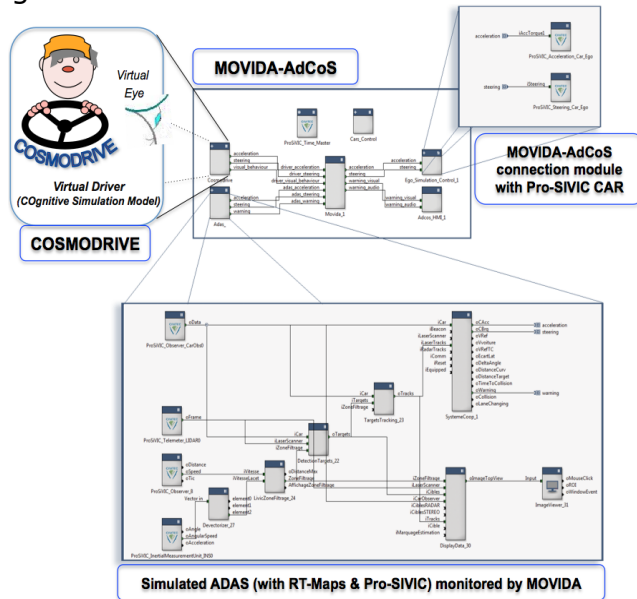




sensors, to visual scanning and distraction status of the driver) and its outputs (from information or warning delivered by the HMI, to vehicle automation functions) when interacting with COSMODRIVE driver model (i.e. the “HF component” of the RTP). Results collected from these simulations were used to progressively increase the MOVIDA-AdCoS efficiency and effectiveness, in accordance with future end-users needs (this virtual design process is in-depth described in D4.7).

**3.3.2.4 Integration**

All the MTTs required for the MOVIDA-AdCoS and its design process with the V-HCD platform were integrated from RTMaps software. The RTMaps diagram presented in Figure 59 provides an overview of the COSMODRIVE and MOVIDA integration/interfacing with this software.

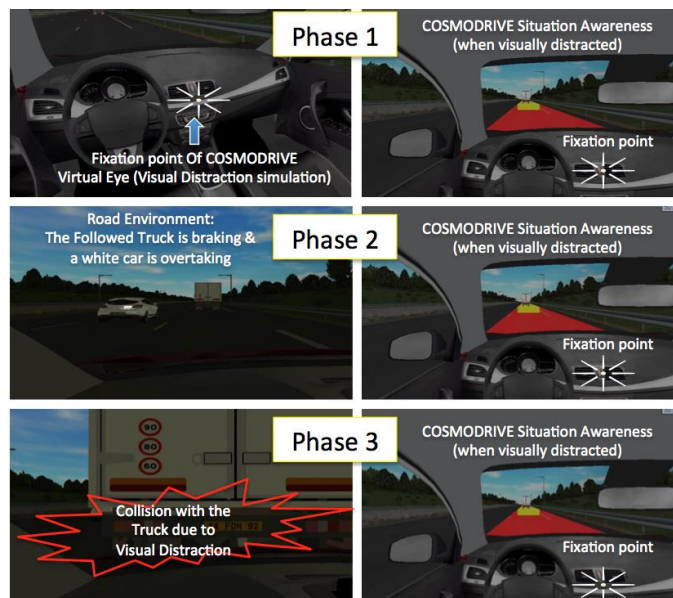


**Figure 59: RTMaps diagram for MOVIDA-AdCoS tests with COSMODRIVE**

On this figure, the MOVIDA-AdCoS receives on the one hand inputs (1) from COSMODRIVE regarding both drivers’ visual behaviour (to assess visual distraction state of the driver) and their actions on vehicle commands (for lateral and longitudinal control of a Pro-SIVIC car) and (2) from the ADAS virtually simulated with Pro-SIVIC and RTMaps. On the other hand, MOVIDA-AdCoS generates outputs towards the Pro-SIVIC virtual car commands to implement MOVIDA-AdCoS driving actions (potentially combined at this level with COSMODRIVE’s actions).

### 3.3.2.5 Results of Proof of concept

In WP4 and WP9, the objective was to use COSMODRIVE-based simulations on the V-HCD platform to support MOVIDA-AdCoS virtual Design and Validation from dynamic simulations, by considering the future use of this AdCoS by real drivers (i.e., end-users, as simulated with COSMODRIVE). The following Figure 60 provides a typical example of the V-HCD use case for identifying critical scenarios due to visual distraction of the driver (as simulated with COSMODRIVE), and then to support the virtual Human Centred Design and Test of the MOVIDA-AdCoS, as implemented during HoliDes.



**Figure 60: simulation of visual distraction effects with COSMODRIVE**

In this generic scenario, the visual distraction begins at phase 1, when the fixation point of the virtual eye of COSMODRIVE focuses on the dashboard (i.e. Off-Road glance). At this moment, the driver's mental model (i.e. his/her Situation Awareness) of the road environment is correct, because the off-road glance is only starting. However, due to this visual distraction, the driver/COSMODRIVE may not detect the braking of the followed truck.

Two seconds later (i.e. phase 2), the truck is critically close and a white car is currently overtaking our driver. However, due to the visual distraction effect, the Situation Awareness of COSMODRIVE is not updated (the lead truck is still far and not any overtaking car is integrated in its mental representation), and the driver is not aware at all of the imminent risk of accident. Then, if the visual distraction is

persisting one second more, a collision with the Truck will occur (phase 3), without any awareness and reaction of the driver / COSMODRIVE.

To adequately support human drivers in this generic “use case of reference”, MOVIDA-AdCoS firstly computes (from its virtual radars and cameras) the Inter-Vehicular Time and the Time to Collision with the followed truck, and detects in parallel all approaching vehicles on the left lane. From the other side, MOVIDA functions are also in charge to assess the visual distraction status of the driver (as simulated on the V-HCD with COSMODRIVE model), in order to interact with him/her in an adaptive and cooperative way.

According to the frontal and lateral collision risks, merged with the drivers’ distraction status as assessed by MOVIDA, this AdCoS may alternatively generate different warnings (visual and auditory) informing the driver on the necessity to (1) look at the road, (2) to keep or to change of lane, and (3) to brake. In case of dangerous behaviours or critical error of the driver, MOVIDA may also take the control of the car (a) for implementing an emergency braking, (b) for avoiding a critical lane change implemented by the driver, or (c) by jointly combining the two preceding actions if required to avoid the accident.

The following figures present a set of different outputs generated by the MOVIDA-AdCoS, as collected from different variations (i.e. replaying) of the initial generic scenario, by alternatively considering (1) a more or less distracted driver when the truck starts to brake, (2) a more or less hard braking of the truck, and (3) the position of other cars on the left lane.

In case of a well-managed traffic situation (regarding both frontal and lateral collision risks) by a non-distracted driver, corresponding to an “ideal case of reference”, MOVIDA-AdCoS may only inform the driver about the possibility to implement the Lane Change Manoeuvre (i.e. No car on the left lane). The following figure (Figure 61) provides a typical example of MOVIDA outputs occurring in this driving context, when simulated with the V-HCD platform.



**Figure 61: Visualization of MOVIDA outputs for a well-managed situation by a non-distracted driver (as simulated with COSMODRIVE)**

By contrast, if the driver is visually distracted while the truck starts to brake, MOVIDA-AdCoS warns the drivers about this event and informs him/her - from the warning presented in Figure 62 - that a lane change is required and currently possible (i.e. Not any car is on the left lane and/or is approaching on the rear left lane).



**Figure 62: Visualization of MOVIDA outputs delivered to support the Lane Change manoeuvre of a visually distracted driver**

In case of an overtaking car occurring on the left lane (more particularly in the blind spot areas) associated with a braking of the followed truck, a warning is sent by MOVIDA to inform the driver about the risk of lateral collisions if a Lane Change manoeuvre is immediately implemented. From this warning (Figure 63), it is expected that the driver will keep his/her lane, until the lane change manoeuvre is possible and safe.



## HoliDes

Holistic Human Factors Design of Adaptive Cooperative Human-Machine Systems

# HoliDes



**Figure 63: Visualization of MOVIDA outputs to warn the driver of a dangerous Lane Change manoeuvre**

If the driver (distracted or not) starts to implement a dangerous Lane Change manoeuvre (by activating the blinkers, turning the steering wheel, and then approaching of the left side of the lane, for instance) while another car is currently overtaking him/her, a warning is sent to the driver and MOVIDA-AdCoS takes the automatic control of the car in order to avoid the lateral collision risk by keeping the car in its current lane (Figure 64).



## HoliDes

Holistic Human Factors Design of Adaptive Cooperative Human-Machine Systems

# HoliDes



**Figure 64: Visualization of MOVIDA outputs when the Automatic Lane Keeping function is activated**

Regarding the frontal collision risk management, there are 2 options in MOVIDA: "Warning" (i.e. Frontal Collision Warning system; FCW) or "Automatic Braking" implemented by the AdCoS (i.e. Frontal Collision Avoidance system; FCA). Figure 65 presents a typical example of FCW outputs when a distracted driver is assisted by MOVIDA. When this warning occurs, the driver should immediately brake to avoid the frontal collision.



## HoliDes

Holistic Human Factors Design of  
Adaptive Cooperative Human-  
Machine Systems

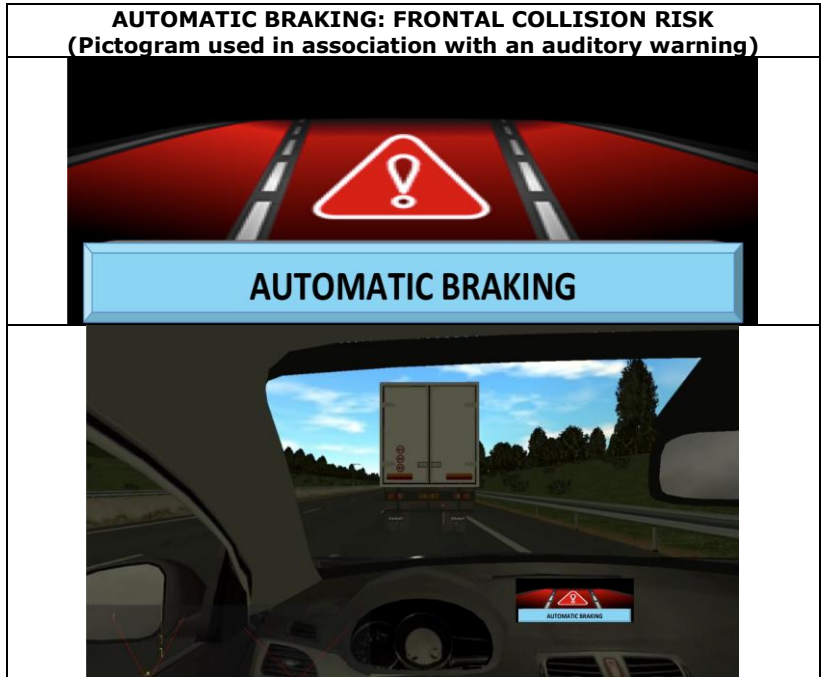
# HoliDes



**Figure 65: MOVIDA outputs to warn the driver of frontal collision risk**

In case of highly critical and very imminent risk of frontal collision (less than 1 second of TTC), or in case of a visually distracted driver assessed by MOVIDA functions, the AdCoS may take the control of the car to implement an emergency braking. In this context, an auditory warning, associated with the pictogram presented in Figure 66, are delivered to the driver to inform him/her about the automatic braking manoeuvre implemented by the AdCoS.





**Figure 66: Visualization of MOVIDA outputs when the Automatic Braking is implemented by the AdCoS**

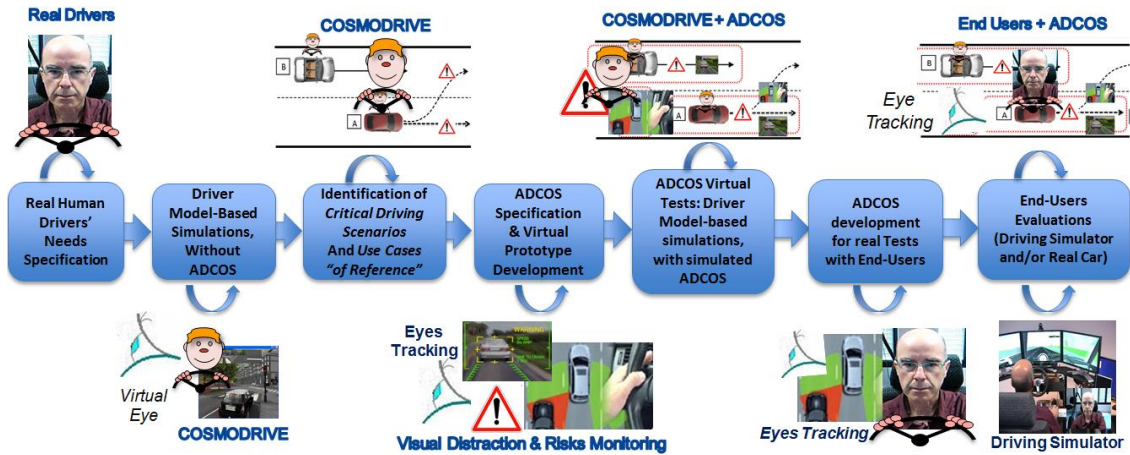
Finally, Automatic Lane Keeping and Braking functions may also be jointly in case of both totally impossible Lane Change and imminent Frontal Collision risk. In this extreme case, the "Full Automation" modality (i.e. Lateral and Longitudinal Control of the car) implemented by MOVIDA has to save the life of the driver, and the different pieces of information presented in Figure 67 are delivered to the driver.





# HoliDes

## Holistic Human Factors Design of Adaptive Cooperative Human-Machine Systems



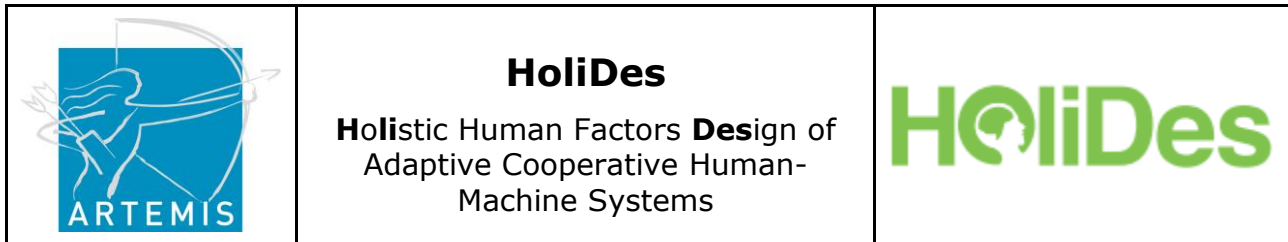
**Figure 68: Virtual design process of MOVIDA-AdCoS with the V-HCD platform**

Synthetically, this V-HCD platform was more particularly used during the project to support MOVIDA AdCoS design processes at 2 main levels.

At the earliest stages of the design process, COSMODRIVE-based simulations were used to simulate human drivers' performances and risks in the frame of an unassisted driving, in order to identify the critical driving scenarios due to visual distraction for which an AdCoS based on MOVIDA could support them. These critical scenarios correspond to the traffic situations when the visual distraction critically impacts the human drivers' reliability, and then increasing the risk of accident. Through these simulations, it has been possible to provide ergonomics specifications of human driver needs, in association with a set of "Critical Instances" of our initial generic scenario (as the core "Use Cases of reference" in WP9), to be at last supported by MOVIDA driving aid.

During the virtual design process of the AdCoS, simulations of MOVIDA-based assistance according to situational risk and the drivers' visual distraction status were implemented in WP4/WP9 in order to progressively design, evaluate and thus increase the MOVIDA-AdCoS *efficiency* for the different critical scenarios and use cases of reference previously identified.

From the use of COSMODRIVE simulation model - as a predictor of real drivers' needs - these simulations allowed the designer to assess the future *effectiveness* of MOVIDA, before developing a real prototype of the AdCoS and then testing its effectiveness among human drivers, through costly full scale tests to be implemented on driving simulators and/or with real cars (final stage of the design process).



In addition with this “Low-Cost” Human Centred Design approach supported by the V-HCD, advantages in using a Human driver model in the design process of and AdCoS are to consider end-users’ needs since the earliest stages (when not any AdCoS prototype is already available), and then to investigate driving scenarios and AdCoS functioning in a systematic way, that is not exhaustively possible and very expensive, when (partially) applied among real human drivers.

### **3.3.3 AdCoS based on Adapted Assistance**

The Adapted Assistance AdCoS has been already introduced in details in D3.6 and in several deliverables of WP9. Here it is briefly recalled to better understand the progress here reported and achieved during the end of the third year of the project.

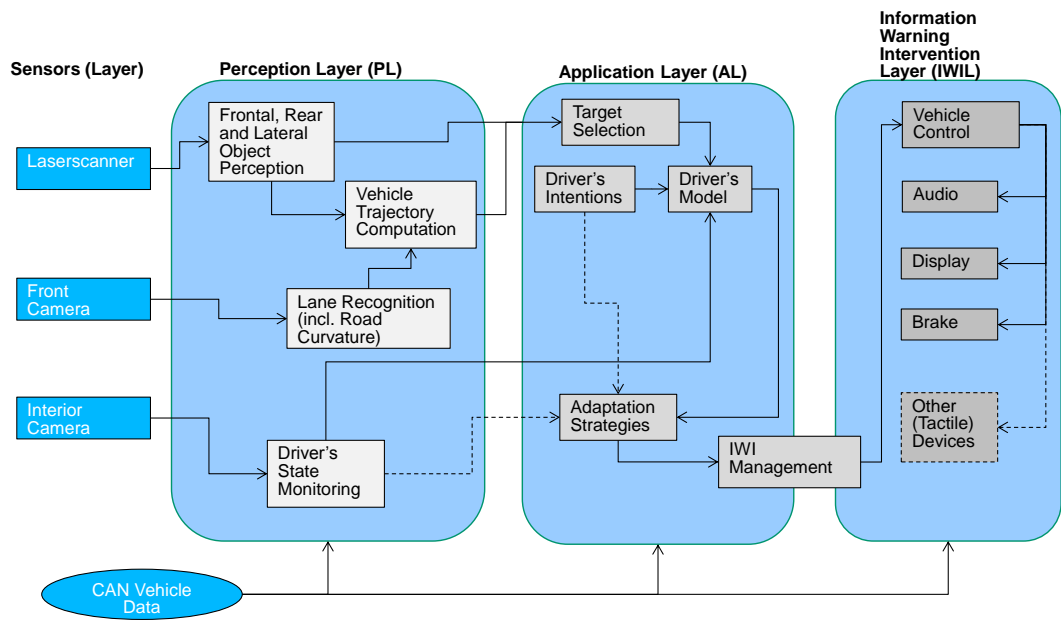
The Adapted Assistance AdCoS is implemented on the CRF (Centro Ricerche Fiat) test-vehicle, which is a Fiat 500L, with the following sensors installed on-board, each one providing raw input to the system (see Figure 69):

- External camera to detect the edges of the lanes on the road and the relative position of the ego-vehicle in the lane.
- Internal camera to detect the head position of the driver (and where he/she is looking at).
- Laser-scanner sensors (four in total: one in the front, one in the rear, one in the left side and another one in the right side of the vehicle) to provide a real-time estimation of the current traffic situation.

The following global functionalities are implemented:

- Lane-Change Assistant (LCA) and Overtaking Assistant (OA).
- Forward Collision Warning (FCW), including assisted braking.

The Adapted Assistance AdCoS is able to adapt to the internal and external scenarios. The “optimal” manoeuvre is suggested from the machine-agent to the human-agent, by means of specific warnings, advice and information, according to the visual state and intentions of driver, as well as to the external environment.



**Figure 69: Architecture of the Adapted Assistance AdCoS**

The elaboration process of the Adapted Assistance AdCoS can be broken down in four stages:

- the perception of the traffic environment around the host vehicle in real-time, as well as of the driver's state
- the assessment and interpretation of the current traffic situation and of the driver's state,
- the planning of appropriate manoeuvres and actions and
- the action to control the vehicle and guide it safely along the planned trajectory and to clearly communicate with the driver

In particular, by looking at Figure 69, the Driver's State Monitoring block classifies if the driver is visually distracted or not (determining distraction of the driver from vehicle dynamic data) exploiting the **Driver Distraction Classifier tool**. On the other hand, the Driver's Intentions block estimates the driving characteristics and intentions (e.g., the wish to change the lane and to overtake) by means of the **Driver Intention Recognition tool**.

***The Driver Distraction Classifier and the Driver Intention Recognition are then the tools used by the Adapted Assistance AdCoS to estimate the user's status.*** Both tools have been thoroughly presented and discussed in past WP3 deliverables in terms of their input-output and architecture (see D3.6, Section 3.4.2).



## HoliDes

Holistic Human Factors Design of  
Adaptive Cooperative Human-  
Machine Systems

HoliDes

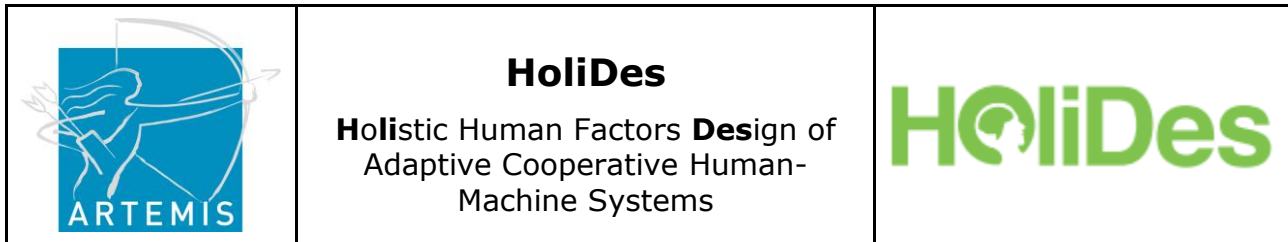
The **Driver Intention Recognition (DIR) module** is a non-lifecycle MTT developed by OFF that provides the AdCoS “Adapted Assistance” with the hidden intentions of the driver in two-lane highway overtaking scenarios, and as such, represents a MTT for context assessment, resp. assessing the user status. Driver Intention Recognition usually deals with the recognition of manoeuvre intentions, which for the target scenario of the AdCoS translates to the recognition of lane change intentions. As such, the DIR shall be able to recognize the intention to perform a lane change to the fast (resp. left) lane, a lane change to the slow (resp. right), or the absence of such an intention, as early as possible.

The **Driver Distraction Classifier**, developed by UTO is in charge of the assessment of the distraction as one of the “trigger” signal for the adaptation. In fact, depending on the cognitive state of the driver (if he/she is distracted or not) and on his/her intentions (the will to change lane), the strategies of the AdCoS adapts accordingly.

The Driver Distraction Classifier uses as input for the classification of the driver’s state only the following vehicle dynamics:

- Speed [m/s]
- Time To Collision [s]
- Time To Lane Crossing [s]
- Steering Angle [deg]
- Street Curvature [deg]
- Lateral Position [m]
- Lane width [m]
- Position of the accelerator pedal [%]
- Position of the brake pedal [%]
- Turn indicator [on/off]
- Yaw rate [deg/s]

The output is represented by the annotation of the driver’s distraction in terms of “distracted” and “not distracted”. As already presented in D3.6, the Driver Distraction Classifier consists of two modules. The first module works offline and learns the classifier from driving data captured by the CAN network of the vehicle and by the car sensors. The second module works online and detects the status of the driver using the knowledge acquired offline, i.e., it provides an online measure of the driver distraction that can be used for adaptation purposes. Theoretical developments and experimental validation and comparison have been carried out during the first year of the project, leading to the Extreme Learning Machine as the selected machine learning approach (see D3.4 for more details). Two RTMaps modules have been developed: one that performs the data pre-processing strategies as required by the classifier to work, and one that implements the neural network computation needed for the classification of the inputs. This second RTMaps



module is inserted in the processing pipeline of the Driver’s Model (see D3.6 for further details).

The output of the Driver Distraction Classifier and of the Driver Intention Recognition, together with the traffic context, are used by the Driver’s Model block – named “**Co-pilot**” – which is in charge of determining the optimal manoeuvre to be suggested to the driver implementing this way the adaptation of the assistance system, according to the *distraction* and *intentions* of the human driver.

The **Co-pilot** which is the driver model developed by University of Torino for the CRF demonstrator in WP9 has a central core which computes an “optimal manoeuvre” that is then suggested to the user through an appropriate, adaptive HMI. The modelling formalism used to realize the Co-pilot is that of Markov Decision Process (MDP), a well-known formalism defined by Bellman in the early sixties for studying optimization problems (see D4.5 for model details). The Co-pilot is implemented as RTMaps module to be integrated in the RTMaps AdCoS Model Adapted Assistance (see D9.6 for more details).

The component is therefore designed to take as input a set of asynchronous data flows from multiple physical sensors and data analysers (i.e., intention and distraction classifiers modules), and produce as output the MDP strategy and the estimated warning level, realizing the adaptation logic.

The last part of the system architecture of Figure 69 illustrates the Human Machine Interface (the **Adapted Assistance HMI**), which implements the communication part of the adaptation loop, and aims at presenting the information to keep the driver informed about the interpretation of the traffic situation, as well as the planned and suggested manoeuvres.

The Adapted Assistance HMI communicates with the driver, helping him in quickly recovering from situations that are judged by the system as “risky”, according to the internal (distraction, intention) and external (road and traffic conditions) context. The HMI has been developed by REL focusing on the overtaking manoeuvre. Preliminary task modelling and task analysis have been carried out for deriving the cognitive tasks involved in the manoeuvre (cognitive, motor, visual or some combination thereof) and the consequent cognitive, motor and visual loads (D2.4, D9.3). The communication strategy has been designed in order to avoid the overloading of the already engaged communication channels. The derived solution envisions a visual and acoustic warning in case of distraction while performing a lane change, reinforced with a haptic warning in case an approaching vehicle hinders the manoeuvre: in this context, the haptic warning indicates the direction of such a vehicle by means of the vibration of the left/rear/right part of the seat or of the steering wheel. Finally, the concept of the HMI has been implemented by

leveraging the communication guidelines (D3.7a, Communication guidelines Annex I).

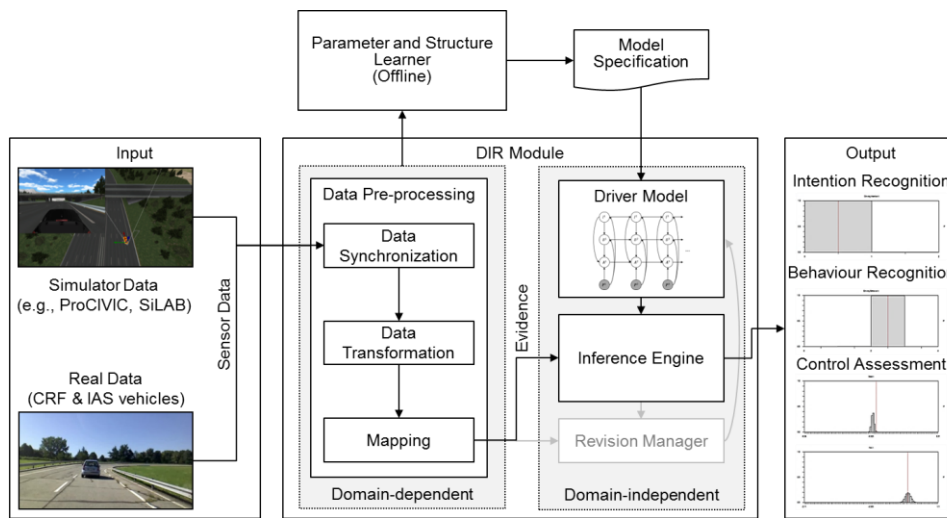
For more information about the final architecture of the Adapted Assistance AdCoS, the reader is re-directed to D9.10.

In the following, only the updates with respect to D3.6 in terms of the input/output, data flow, tools used, integration details and evaluation results of the AdCoS and its component are provided

### 3.3.3.1 Data flow full treatment chain

#### Driver Intention Recognition

As depicted in Figure 70, the DIR module consists of two parts, a domain-dependent part (tailored to the actual system architecture and specification of the AdCoS “Adapted Assistance”) that primarily deals with pre-processing and enhancement of the available (raw) sensor input, which we will call the *data pre-processing component*, and a domain-independent part, which we call the *inference-engine component*, consisting of an inference engine that enables the DIR module to answer probabilistic queries in respect to a probabilistic model of the human driving behaviour. As previously described in Deliverable “D3.5 – Techniques and Tools for Adaptation Vs1.5”, the DIR module conceptually requires input in terms of information about traffic participants in the vicinity of the driver, the future path of the road, the state of the driver’s vehicle (ego-vehicle), the driver’s control behaviour and additional contextual information, like e.g., the current speed-limit.



**Figure 70 : Schematic overview of the DIR module.**



The previous deliverable D3.6 “Techniques and Tools for Adaptation Vs1.8 incl. Handbooks and Requirements Analysis Update” provided an overview of the DIR module developed based on experimental data obtained in simulator experiments. Here, we will update this overview of the DIR module developed based on real-world driving data obtained on the CRF demonstrator vehicle in 2015.

### 3.3.3.2 Inputs & outputs

#### Driver Intention Recognition

Previous versions of the DIR module were based on experimental data obtained in simulator experiments providing an almost ideal amount and quality of contextual information, which reflected in the input required for the DIR module. For the integration of the DIR module in the AdCoS “Adapted Assistance” and the CRF demonstrator vehicle, we tailored the required input to the limited available sensor information provided by the CRF demonstrator vehicle. Table summarizes the input data required for the inference-engine component of the DIR module tailored to the AdCoS “Adapted Assistance”, where inputs 13 to 108 refer to information about twelve potential vehicles in the vicinity of the driver, classified based on the position in relation to the ego-vehicle, as depicted in Figure 71 and Figure 72.

**Table 8: Input data for the inference-engine component of the DIR module.**

Index	Name	Description	Unit
1	TIME	Timestamp	[ms]
2	BRAKE_PEDAL	Indicating whether or not the braking pedal is pressed or not	[#] 0: Not pressed 1: Pressed
3	ACCELERATION_PEDAL	Acceleration-pedal position	[%]
4	STEERING_ANGLE	Steering wheel angle	[deg]
5	EGO_SPEED	The current velocity of the ego-vehicle	[km/h]
6	SPEED_LIMIT	The current speed limit. Derived from the pre-processing component.	[km/h]
7	LATERAL_DISTANCE	Lateral distance from the left lane edge, derived from the pre-processing component	[m]
8	HEADING_ANGLE	Angle between the ego-vehicle’s heading and the course of the road, derived from the pre-processing component	[deg]
9	YAW_RATE	Rate of change of the heading angle.	[deg/s]
10	CURVATURE	Curvature of the road at the current position of the ego-vehicle, derived from the pre-processing component	[m <sup>-1</sup> ]

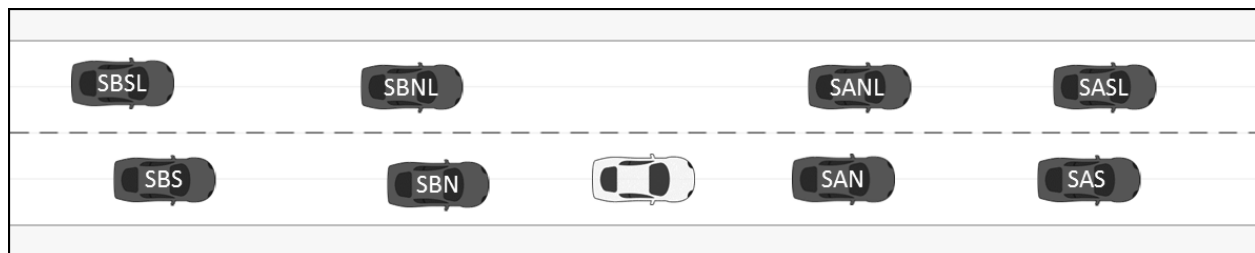


# HoliDes

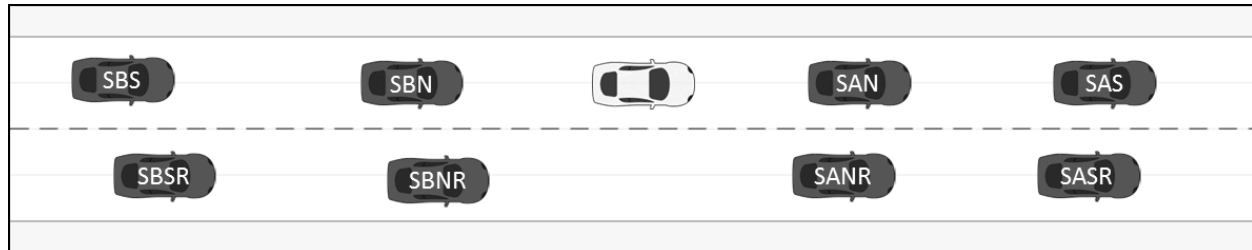
**H**olistic Human Factors **D**esign of Adaptive Cooperative Human-Machine Systems



11	INDICATOR_STATUS	Status of the indicator	[#] 0: not used; 1:right 2: left
12	EGO_LANE	Steering wheel angle	[#] 0: Fast lane 1: Slow lane
13	SANL_X	Global x-coordinate of the ANL vehicle	Meters with two decimal places.
14	SANL_Y	Global y-coordinate of the ANL vehicle	Meters with two decimal places.
15	SANL_SPEED_X	Global z-coordinate of the ANL vehicle	Meters with two decimal places.
16	SANL_SPEED_Y		
17	SANL_ID	ID of the ANL vehicle	Integer, -1 if no vehicle exists.
18	SANL_TYPE	Velocity of the ANL vehicle	Km/h with three decimal places.
19	SANL_LENGTH	Bumper-to-bumper distance between the ego-vehicle and the ANL vehicle.	Meters with two decimal places within the interval [0,200]
20	SANL_WIDTH	Unused	
...			
101	SBSR_X	Global x-coordinate of the BSR vehicle	Meters with two decimal places.
102	SBSR_Y	Global y-coordinate of the BSR vehicle	Meters with two decimal places.
103	SBSR_SPEED_X	Global z-coordinate of the BSR vehicle	Meters with two decimal places.
104	SBSR_SPEED_Y	The lane, the BSR vehicle is currently inhabiting	4: Fast lane 5: Slow lane
105	SBSR_ID	ID of the BSR vehicle	Integer, -1 if no vehicle exists.
106	SBSR_TYPE	Velocity of the BSR vehicle	Km/h with three decimal places.
107	SBSR_LENGTH	Bumper-to-bumper distance between the ego-vehicle and the BSR vehicle.	Meters with two decimal places within the interval [0,200]
108	SBSR_WIDTH	Unused	



**Figure 71: Classification of potential alter-vehicles (dark) in the vicinity of the ego-vehicle (light) in relation of to the position of the ego-vehicle when driving on the slow lane.**



**Figure 72: Classification of potential alter-vehicles (dark) in the vicinity of the ego-vehicle (light) in relation of to the position of the ego-vehicle when driving on the fast lane.**

The output of the DIR module tailored to the AdCoS “Adapted Assistance” is as follows:

- A vector of probabilities representing the belief state over target lane intentions  $p(I^t|e^{1:t})$ , i.e. the probability for each intention  $i^t \in Val(I)$  given the currently available and past evidence  $e^{1:t}$ .
- A vector of probabilities representing the belief state over behaviours  $p(B^t|e^{1:t})$ , i.e. the probability for each driving manoeuvre/behaviour  $b^t \in Val(B)$  given the currently available and past evidence  $e^{1:t}$ .

Intention recognition on highway scenarios is primarily concerned with the recognition of *lane-change* intentions (e.g., recognizing that the driver intends to perform a lane-change to the fast lane to overtake a slower vehicle). The DIR module internally uses a slightly different concept in trying to recognize *target lane* intentions (e.g., recognizing that the driver intends to drive on the fast lane), but knowing the current lane, the ego-vehicle inhabits, the target lane intentions can easily be mapped onto lane-change intentions (e.g., an intention to drive on the fast lane, while driving on the slow lane implies the intention to change to the fast lane). For the AdCoS “Adapted Assistance”, the belief state over target lane intentions  $p(I^t|e^{1:t})$  is therefore mapped onto a belief state over lane change intentions. Additional components allow deriving the most probable lane change intention, which is used as an input for the co-pilot of the AdCoS “Adapted Assistance”.

### Additional input:

Providing the input depicted in Table 8 basically allows the utilization of the DIR module without the need for the data pre-processing component. Unfortunately, while such input can easily be provided in simulator environments, in real-world scenarios, the classification of the alter-vehicles is not provided directly, it needs to be derived from the limited available sensor information, based on the current lateral position and heading angle of the ego-vehicle and the curvature of the road,

and other information like, e.g., the speed limits, the current lane, or curvature information may be missing or unreliable in real-world setting.

As such, the data pre-processing component currently implements a *world-model* that enriches the actual sensor input available by the lateral distance from the left lane edge, the lane the ego-vehicle is currently inhabiting, the heading angle, the curvature, the current speed limit, and the classification of the alter-vehicles using the input depicted in Table 9. The world-model is based on a particle filter to estimate the current pose of the vehicle and utilizes on a manually constructed mapping of the distance travelled since entering the highway to the curvature profile and the speed-limit derived from the experimental data provided by CRF.

**Table 9: Additional input for the DIR model, required for data pre-processing when utilized on the CRF demonstrator vehicle.**

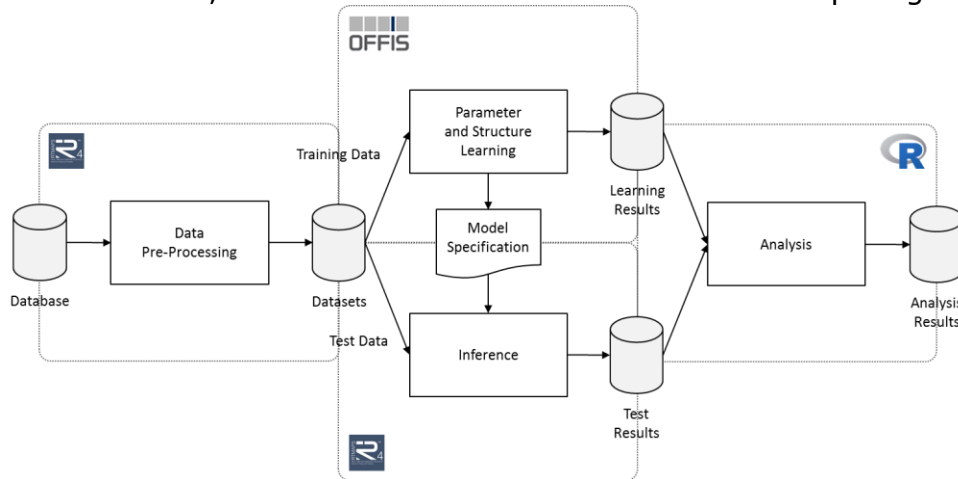
Index	Name	Description	Unit
1	TIME	Timestamp	[ms]
2	XPOS	X-position of the ego-vehicle in respect to an unknown origin, provided by the IBEO sensors.	[mm]
3	YPOS	Y-position of the ego-vehicle in respect to an unknown origin, provided by the IBEO sensors	[mm]
4	COURSE_ANGLE	Yaw angle of the ego-vehicle in respect to some unknown origin axis, provided by the IBEO sensors	[deg]
5	STATIC_OBJECT_OUTLINES	A vector of x-y-z-r-g-b points, where each couple of points represents a coherent segment, processed from the scan points, provided by the IBEO sensors.	[m],[#]
6	SCAN_XYZ_POINTS	A vector of x-y-z-coordinates for each scan point, provided by the IBEO sensors	[m]
7	NB_OBJECTS	Number of objects detected by the IBEO sensors, provided by the IBEO sensors	[#]
8	OBJECT_BOX_CENTERS	Vector of x-y-coordinates defining the centre of the bounding box for each detected object in respect to the ego-vehicle, provided by the IBEO sensors	[m]
9	OBJECT_BOX_SIZES	Vector of width and lengths defining the size of the bounding box for each detected object, provided by the IBEO sensors	[m]
10	OBJECT_BOX_ORIENTATIONS	Vector of angles defining the heading angle for each detected object in respect to the ego-vehicle, provided by the IBEO sensors	[deg]

11	ABSOLUTE_VELOCITIES	Vector of x-y-velocities for each detected object, provided by the IBEO sensors	[m/s]
12	CLASSIFICATIONS	Vector of class identifications (e.g., PKW, LKW) for each detected object, provided by the IBEO sensors.	[#]

### 3.3.3.3 Tools used

#### Driver Intention Recognition

An overview of the tool-chain used for the development of the DIR module is shown in Figure 73. The DIR module is implemented as a set of RTMaps packages that can be utilized within an RTMaps AdCoS model or in isolation (if sufficient sensor input is provided). For parameter and structure learning of BAD MoB models during the training phase, we use a software solution developed by OFF in Visual Studio. For performance evaluation, OFF uses the software for statistical computing R.



**Figure 73: Overview of the tool-chain used for the DIR module.**

### 3.3.3.4 Integration

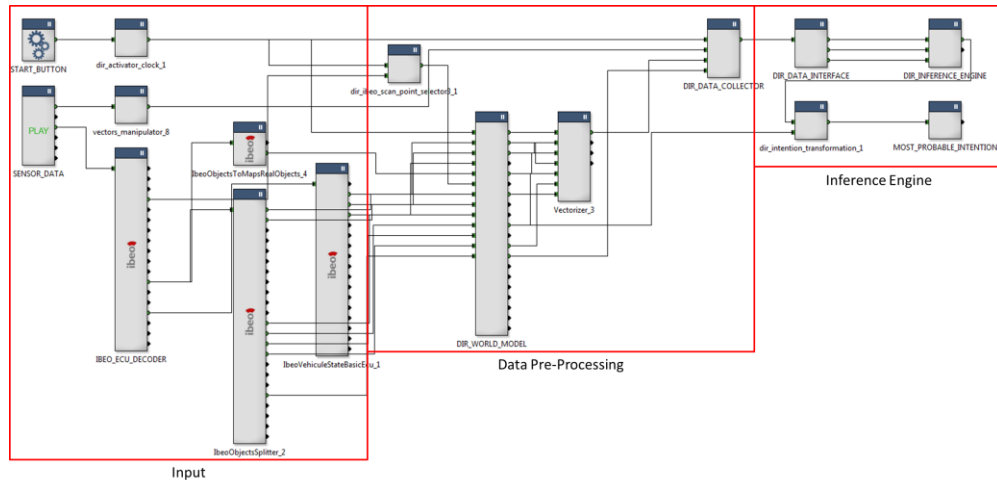
#### Driver Intention Recognition

Both the domain-dependent and the domain-independent parts of the DIR module are implemented in terms of RTMaps packages, providing sets of RTMaps components that can be used for AdCoS modelling and utilization in RTMaps. Using RTMaps, the DIR module has been successfully integrated into the AdCoS “Adapted Assistance” and has been tested on the CRF demonstrator vehicle.



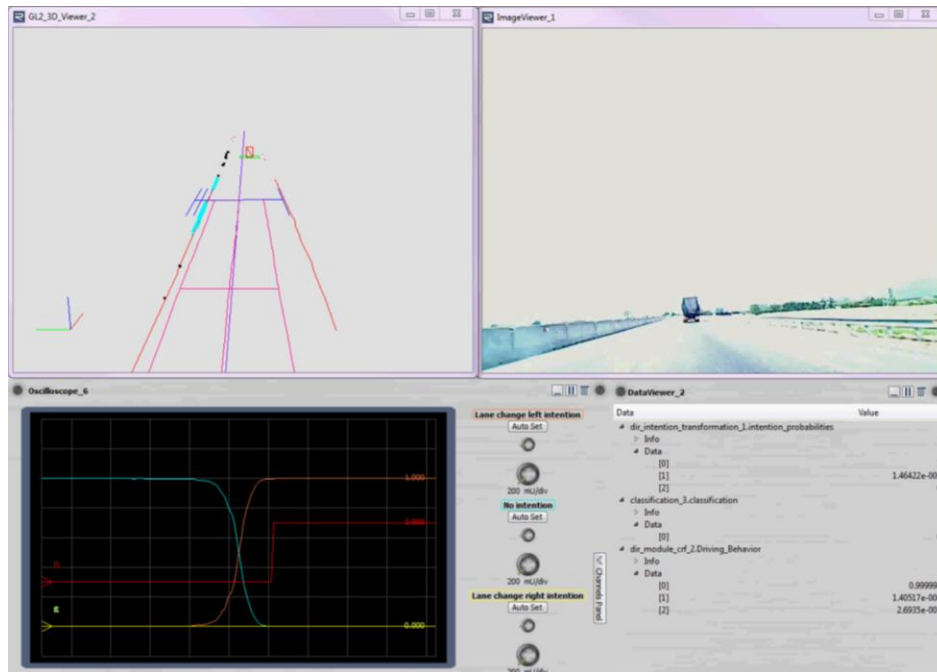
# HoliDes

## Holistic Human Factors Design of Adaptive Cooperative Human-Machine Systems



**Figure 74: (Simplified) overview of the DIR module integrated in RTMaps, arranged to highlights RTMaps components for the Data Pre-Processing and Inference Engine of the DIR module.**

Figure 74 shows an overview of the DIR module modelled in RTMaps, connected to an RTMaps player that provides the experimental data obtained in the CRF demonstrator vehicle. For utilization of the DIR module in the final AdCoS “Adapted Assistance”, the player is replaced by RTMaps components providing sensor information in real-time. Note that components dedicated for visualization of the DIR module have been removed to reduce clutter. Figure 75 shows a screenshot of the DIR module during runtime (using pre-recorded experimental data provided by CRF), where the top left image shows a visualization of the data pre-processing component for enhancing the available sensor input and classification of vehicles in the vicinity of the ego-vehicle, the top right image shows the on-board camera installed on the CRF demonstrator vehicle, and the bottom shows a visual and textual summary of the output of the DIR module.



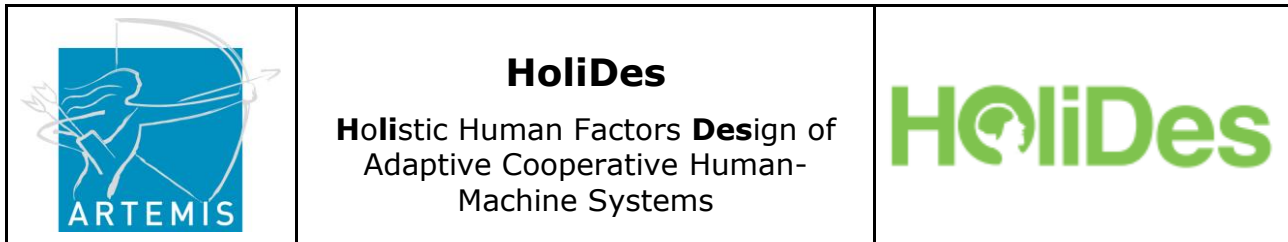
**Figure 75: Example of driver intention recognition module in RTMaps. In this example, the intention to perform a lane change to the left (bottom, orange line) is recognized approx. 1 sec prior to the activation of the indicator (bottom, red line).**

### 3.3.3.5 Results of Proof of concept

The AdCoS as a whole and the HMI have been under evaluation during the third year of the project. In particular, separate studies in the REL simulation environment have been performed for the Adapted Assistance AdCoS and for the HMI.

The AdCoS has been evaluated in comparison with the baseline of the driving assistance system to date (i.e., system that merges the functionalities of the blind spot for the rear and lateral directions and the forward collision warning but that are non-adaptive to the distraction and the intention of the driver). The purpose was to show the benefit of using the information about user's state (i.e., the distraction and the intention) for the adaptation, according to both quantitative (e.g., number of accident during the driving session, percentage of time the driver spent with time to collision under a safe threshold, and others) and (e.g., perceived workload, perceived ease of use, and others) qualitative indicators.

On the other hand, other evaluation studies have been performed about the communication strategies, introduced in D3.6, Communication Guidelines and completed in D 3.7a Annex I. Experimental analysis has been applied to evaluate the benefit of having the communication of why performed by means of the haptic



channel, even in this case from both a subjective and an objective point of views. The evaluation from the subjective perspective has been aimed at comparing, by means of specific questionnaires derived from well-known questionnaire models, the cognitive effort (perceived workload in terms of fatigue and distraction), perceived ease of use, usability, attitudes toward using and intention to use in both cases.

Details about both the evaluation activities are provided in D9.9 and in D9.10. They are here shortly reported.

The results of the AdCoS evaluation compared with the baseline showed that the adaptation had a great benefit on the performance indicators. For the technical assessment, the AdCoS has improved all the performance indicators related to safety by almost 50%. For what concerning the user-related assessment, also in this case the AdCoS showed a good benefit with respect the baseline (see D9.9). For the HMI evaluation (D9.10), results reported the cognitive effort (perceived workload in terms of fatigue and distraction), perceived ease of use, usability, attitudes toward using and intention to use in the compared communication cases. Results showed that the both solutions do not have significant differences in these terms, indicating that, even if the why haptic warning represents a cooperation mode the subjects are not used to, it is judged acceptable as other more familiar warning alarms. Besides, for the haptic warning, a dedicated questionnaire has been created for the assessment of the comprehensibility, distinguishability, perceptibility and effectiveness of the chosen signal. This specific questionnaire reveals that, even if the results of comprehensibility, distinguishability and perceptibility are satisfactory, the effectiveness, defined as the property of conveying the information about the direction of the danger, does not show the same positive results. The novelty of this functionality, unusual for a driver, has influenced the effectiveness for half of the participants.

**Driver Intention Recognition**

To collect data for the development of the co-pilot and the DIR module, CRF performed a free-driving study with the CRF demonstrator vehicle in September 2015, consisting of 28 separate drives on the Italian A55. Participants entered the two-lane highway “A55 Torino-Pinerolo” at the “SP142” entry and exited after approx. 17km at the “Via Maestra Riva” to travel back in the opposite direction to then change to the three-lane section of the “A55 Tangenziale Sud di Torino”, turning at the “Tangenziale Sud-Nord” to travel back to the starting point.

As the target scenario for the DIR module focusses on two-lane highways and as file size limits rendered the sensor information inaccessible after approx. 15 min. of driving, we focused on the first section of each trial, beginning with entering the



“A55 Torino-Pinerolo” at the “SP142” entry and ending when exiting the “A55 Torino-Pinerolo” at the “Via Maestra Riva” exit.

For each trial, we used the data pre-processing component of the DIR module in RTMaps to calculate the input depicted in Table 8 and build up a database of experimental data for the development of the DIR module. We manually annotated each sample of this experimental data with whether the driver was performing a *lane change to the fast lane*, a *lane change to the slow lane*, or just *lane-keeping* driving behaviour. After this manual annotation, we automatically annotated each data sample with whether the driver intended to drive on the fast or on the slow lane, assuming that a change in the target lane intention was present up to 1000ms prior to the annotated beginning of a lane change manoeuvre.

From the annotated experimental data, we then randomly selected 17 trials as training data (169274 samples or approx. 141 min of driving) and reserved seven trials for testing purposes (69953 samples or approx. 58 min of driving). The remaining trials were discarded due to insufficient data quality or out-of-sync errors during data recording.

Given the training data, we used machine-learning methods to learn a probabilistic model (based on BAD MoB models provided by WP2) for driver intention recognition to be utilized in the DIR module. Details on the underlying models and learning algorithms will be provided in Deliverable D2.7 “Modelling Techniques and Tools Vs2.0”. To provide a brief description, let define:

- $L$  denote a binary random variable with the possible values  $\text{Val}(L) = \{\text{slow\_lane}, \text{fast\_lane}\}$ , representing context in terms of the lane, the driver is currently inhabiting,
- $I$  denote a binary random variable behavioural intentions of the driver are represented by a binary random variable  $I$ , with the possible values  $\text{Val}(I) = \{\text{slow\_lane\_intention}, \text{fast\_lane\_intention}\}$  that represents the behavioural intentions of a driver in respect to the lane he/she intends wants to inhabit,
- $B$  denote a discrete random variable with the possible values  $\text{Val}(B) = \{\text{lane change left}, \text{lane change right}, \text{lane-following}, \text{car-following}\}$ , representing a set of four potential behaviours/manoeuvres,
- $A$  denotes a continuous random variable representing the position of a combined acceleration-braking pedal,
- $S$  denotes a continuous random variable representing the steering wheel angle,
- and  $P$  denote a set of discrete and continuous variables  $P = \{P_1, \dots, P_n\}$ , representing a selection of perceptual features that are hypothetically available and important for driver intention recognition.



## HoliDes

Holistic Human Factors Design of  
Adaptive Cooperative Human-  
Machine Systems

Based on previous versions of the DIR module developed for simulation environments (described in D3.6 “Techniques and Tools for Adaptation Vs1.8 incl. Handbooks and Requirements Analysis Update”), we focussed on a generative modelling approach, where the underlying probabilistic model is based on the assumption that the joint probability density  $p(L^{1:T}, I^{1:T}, B^{1:T}, A^{1:T}, S^{1:T}, \mathbf{P}^{1:T})$  can be factorized as:

$$\begin{aligned}
 & p(L^{1:T}, I^{1:T}, B^{1:T}, A^{1:T}, S^{1:T}, \mathbf{P}^{1:T}) \\
 &= \prod_{t=1}^T p(L^t) p(I^t, B^t, \mathbf{P}^t | I^{t-1}, B^{t-1}, L^t) p(A^t | A^{t-1}, L^t, B^t, \mathbf{P}^t) p(S^t | S^{t-1}, L^t, B^t, \mathbf{P}^t).
 \end{aligned}$$

As the quality of the training data was not sufficient to reliably learn models for predicting the control-behaviour for lateral and longitudinal control, and as such output was not planned to be used within the AdCoS “Adapted Assistance”, we focussed on the intention and behaviour recognition aspects and provided the control inputs of the driver as additional input features ( $\mathbf{P}_* = \mathbf{P} \cup \{A, S\}$ ), resulting in the following factorization:

$$\begin{aligned}
 & p(L^{1:T}, I^{1:T}, B^{1:T}, \mathbf{P}_*^{1:T}) \\
 &= \prod_{t=1}^T p(L^t) p(I^t, B^t, \mathbf{P}_*^t | I^{t-1}, B^{t-1}, L^t)
 \end{aligned}$$

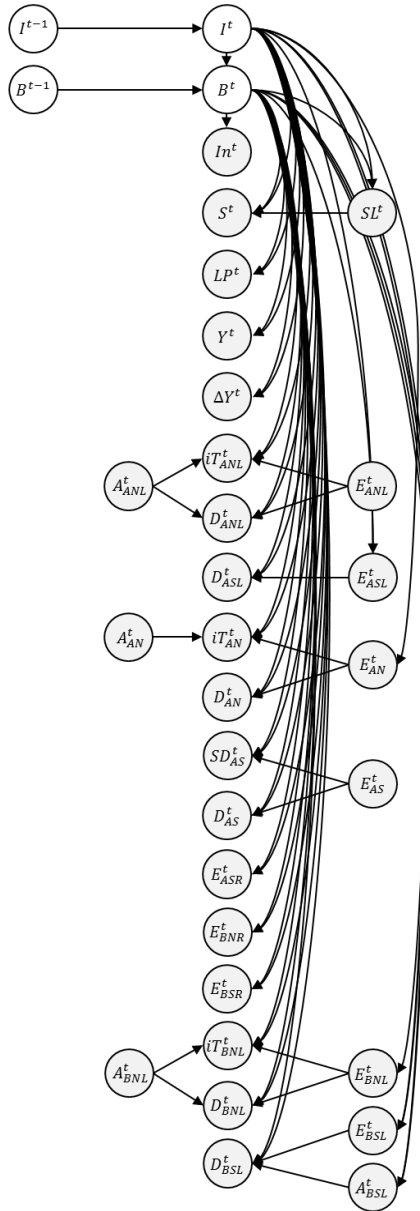
We additionally assumed that the further factorization  $p(I^t, B^t, \mathbf{P}_*^t | I^{t-1}, B^{t-1}, L^t)$  may be described in terms of a (factorized) dynamic model  $p(I^t, B^t | I^{t-1}, B^{t-1}, L^t)$  and an observation model  $p(\mathbf{P}_*^t | I^t, B^t, L^t)$  and that for each  $l^t \in Val(L)$ , the exact factorization of  $p(\mathbf{P}_*^t | I^t, B^t, L^t)$  may differ (i.e., we assume the existence of context-specific independencies). As such, the resulting structure can be understood as a factorized *Hidden Markov Model* where the observation model  $p(\mathbf{P}^t | I^t, B^t, L^t)$  is a so-called *Bayesian Multinet*.

In the context of BAD MoB models, the learning task can be understood as feature selection, in that we try to find a suitable subset of  $\mathbf{P}_*$  important for recognizing intentions and behaviours by learning a corresponding graph-structure factorizing  $p(I^t, B^t, \mathbf{P}^t | I^{t-1}, B^{t-1}, L^t = \text{slow\_lane})$  and  $p(I^t, B^t, \mathbf{P}^t | I^{t-1}, B^{t-1}, L^t = \text{fast\_lane})$ . Figure 76 shows the learned graph-structure factorizing  $p(I^t, B^t, \mathbf{P}^t | I^{t-1}, B^{t-1}, L^t = \text{slow\_lane})$ , Figure 77 shows the learned graph-structure factorizing  $p(I^t, B^t, \mathbf{P}^t | I^{t-1}, B^{t-1}, L^t = \text{fast\_lane})$ .



## HoliDes

Holistic Human Factors Design of Adaptive Cooperative Human-Machine Systems



### Variables:

$I$ : Intentions of the driver  
 $B$ : Behaviours/manoeuvres of the driver  
 $In$ : Indicator signal  
 $S$ : Velocity of the ego-vehicle  
 $SL$ : Speed limit  
 $LP$ : Lateral position of the ego-vehicle in respect to the lane edge of the fast lane  
 $Y$ : Yaw (or heading) angle of the ego-vehicle  
 $\dot{Y}$ : Yaw-rate of the ego-vehicle  
 $E_X$ : Existence of a vehicle X in the vicinity of the ego-vehicle  
 $A_X$ : The area (near or far) a vehicle X is inhabiting in respect to the ego-vehicle  
 $iT_X$ : Inverse time to collision to a vehicle X  
 $SD_X$ : Speed difference between the ego-vehicle and a vehicle X  
 $D_X$ : Distance to a vehicle X

### Vehicle identifiers:

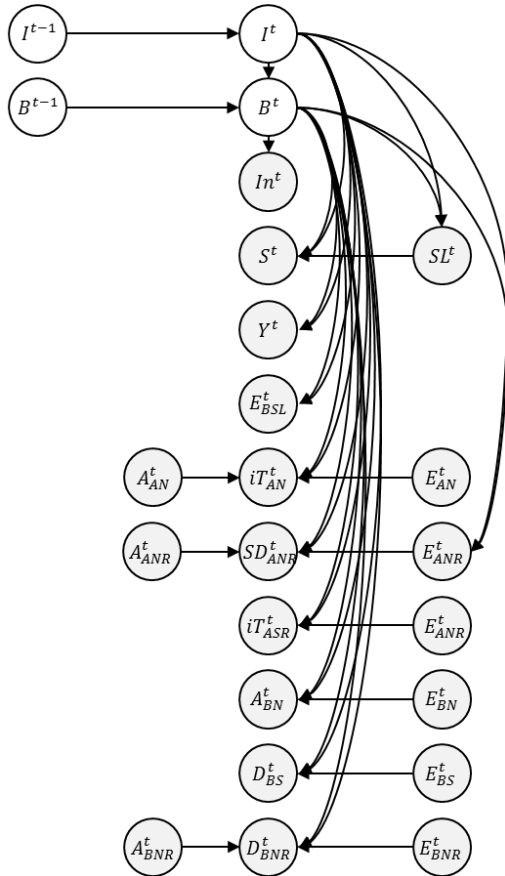
$AN$ : Lead-vehicle on the slow lane  
 $AS$ : Lead-vehicle of the lead-vehicle on the slow lane  
 $ANL$ : Lead-vehicle on the fast lane  
 $ASL$ : Lead-vehicle of the lead-vehicle on the fast lane  
 $BNL$ : Following-vehicle on the fast lane  
 $BSL$ : Following-vehicle of the following-vehicle on the fast lane  
 $ASR$ : Lead-vehicle of the lead-vehicle on the lane right to the slow lane (entries, exits, and sensor failures)  
 $BNR$ : Following-vehicle on the lane right to the slow lane (entries, exits, and sensor failures)  
 $BSR$ : Following-vehicle on the lane right to the slow lane (entries, exits, and sensor failures)  
 $BNL$ : Following-vehicle on the lane right to the fast lane (entries, exits, and sensor failures)  
 $BSL$ : Following-vehicle of the following-vehicle on the fast lane (implying the existence of a following vehicle)

**Figure 76: Learned graph-structure representing  $p(I^t, B^t, P^t | I^{t-1}, B^{t-1}, L^t = \text{slow\_lane})$ . The additional parent  $L^t = \text{slow\_lane}$  and variables not conditioned by  $I^t$  or  $B^t$  are omitted to improve visibility.**



## HoliDes

Holistic Human Factors Design of Adaptive Cooperative Human-Machine Systems



### Variables:

$I$ : Intentions of the driver  
 $B$ : Behaviours/manoeuvres of the driver  
 $In$ : Indicator signal  
 $S$ : Velocity of the ego-vehicle  
 $SL$ : Speed limit  
 $Y$ : Yaw (or heading) angle of the ego-vehicle  
 $E_X$ : Existence of a vehicle X in the vicinity of the ego-vehicle  
 $A_X$ : The area (near or far) a vehicle X is inhabiting in respect to the ego-vehicle  
 $iT_X$ : Inverse time to collision to a vehicle X  
 $SD_X$ : Speed difference between the ego-vehicle and a vehicle X  
 $D_X$ : Distance to a vehicle X

### Vehicle identifiers:

$AN$ : Lead-vehicle on the fast lane  
 $ANR$ : Lead-vehicle on the slow lane  
 $ASR$ : Lead-vehicle of the lead-vehicle on the slow lane  
 $BN$ : Following-vehicle on the fast lane  
 $BS$ : Following-vehicle of the following-vehicle on the fast lane  
 $BNR$ : Following-vehicle on the slow lane  
 $BSL$ : Following-vehicle of the following-vehicle on the lane left to the fast lane (due to sensor errors)

**Figure 77: Learned graph-structure factorizing  $p(I^t, B^t, P^t | I^{t-1}, B^{t-1}, L^t = \text{fast\_lane})$ . The additional parent  $L^t = \text{fast\_lane}$  and variables not conditioned by  $I^t$  or  $B^t$  are omitted to improve visibility.**

Details on the evaluation of the DIR module will be provided in D9.9 "Empirical Evaluation of the Automotive AdCoS and HF-RTP Requirements Definition Update (Feedback)".

## **4 Holistic Human Factors Design Guidelines**

### **4.1 Introduction**

#### **4.1.1 Objective**

The adaptation framework offers a structured way for the consideration of human factors in systems development after a detailed design is known to the developer. The human factors guideline targets at complementing the framework by support developers in systems development from the very beginning. It is designed to guide novices in the field of human factors in the most initial design and conceptualization phases by creating awareness for the importance of an early anticipation of human operators' strengths and weaknesses and user-centred automation design. The guideline aims to invoke 'human factors thinking' in developers with little experience in this area by providing an instructional wizard; however, it will be no means be able to replace the need for consulting in-depth literature on specific human factors that are critical for the system to be built.

Similar to the adaptation framework, the guideline also serves human factors specialists as a reference book to ensure completeness and thoroughness of their design models. Its focus lies on lending a hand to system developers when dealing with human factors in the design of adaptive cooperative human-machine systems (AdCoS) from scratch. It is closely connected to the adaptation framework as it serves as a structured process for the creation of adaptive loops. Designers and developers are encouraged to consult the human factors guideline before the initial concepts of the AdCoS are sketched. Also, the assumption is that when provided with quantitative technical specifications and functional requirements, developers lack awareness or experience to deal with rather qualitative human factors requirements that are defined by users' needs and capabilities, context characteristics and task specific factors. This report is designed to help designers and developers to create a rich human factors view on their AdCoS design and to deal with human factors in a holistic way.

#### **4.1.2 Holistic Design**

Rather than providing design recommendations for single interface elements, the human factors guideline is supposed to take a holistic perspective. An AdCoS' elements may therefore not be treated as isolated, but the system should be designed as a whole. Designers and developers are encouraged to focus on interrelations system layers and components [4]. The guidelines are required to be applicable across domains and designs; therefore a trade-off between universal validity and level of detail has to be taken into account.

As a simplification, the basic concepts such as adaptivity and adaptation will be identified and defined as separate layers of an AdCoS in a reductionist approach. Next, the specification of their components will be facilitated by introducing the five Ws and one H method. First, individual components' behaviour, that is all possible states they can take, should be specified. Next, the interrelations between components and layers need to be depicted.

## 4.2 Requirements analysis

### 4.2.1 Guideline Design

Holistic, domain independent guidelines are unable to provide specific quantitative design recommendations; instead helpful data sources are selected, but the data have to be extracted based on requirements known to the developer. The guideline also has to cope with the completeness / precision trade-off. According to Campbell [2], the guidelines should aim at giving recommendations on all topics of importance and in the next step focus on what level of precision can be achieved.

In the history of human factors engineering, there have been many attempts at designing human factors guidelines that are applicable across domains [2]. The general consensus however was that most human factors are situation specific and generic guidelines are of little help. This resulted in guidelines for specific domains that have been developed following Campbell's [2] approach. For the requirements analysis, potential user interviews on current handling of human factors and information sources are recommended in order to guide the guideline compilation process, the actual guideline formulation and its format.

### 4.2.2 Target Group Interviews

Structured interviews have been conducted with an AdCoS-owner and system designer from the automotive and a developer from the control room domain. Each of the interviews took about 20 minutes and covered 4 questions on the status quo of addressing human factors in design and conceptualization, the introduction of human factors requirements, information sources and their expectations towards the guidelines. After answering the questions, interviewees were given the opportunity to give comments.

According to the interviewees, human factors are introduced before, during and after the implementation of the design. This situation however is not perceived as optimal. One specifically stressed the need for the anticipation of human factors in early system design. Interviewees would like to know what differences between individual users they have to consider, what drives technology acceptance, what the key human factors are for the system planned to be built and what their

implications for the user interface are in early design. However, time pressure and other factors do often prohibit thorough consideration of human factors issues before or during development. Naturally, these issues do not become apparent until prototyping and testing.

When needed, information is acquired by communication with experienced partners and colleagues (if available), literature research and, quite often, introspection and intuition. Literature research is considered helpful, but effortful and time consuming. Also, effective literature research requires experience to identify and recognize the key terms for abstract human factors problems. If applicable, ISO standards and formal testing are used as a starting point. Apart from design, interviewees expressed a need for guidance in testing and evaluation of AdCoS with respect to human factors.

### **4.2.3 Guideline Requirements**

Based on the formal requirements and the insights from the interviews, the guidelines need to reduce the costs of human factors thinking at the design stage and to enable designers and developers to get access to the required information. The guidelines will provide a structural classification of adaptive components that provides a quick and easy access in order to reduce the perceived costs of human factors awareness in design. Also, the guidelines will occasionally highlight typical human factors pitfalls and provide design literature recommendations for extensive consideration.

## **4.3 Adaptive Components in AdCoS**

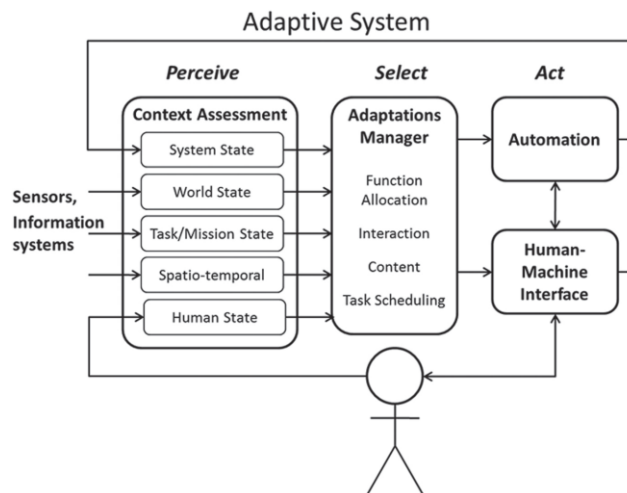
### **4.3.1 Adaptability, Adaptivity and Adaptation**

Most automation systems are built for a specific purpose that they are able to fulfil when used in a predetermined context. For static automation systems, it is up to the designers to define what task requirements their system can cope with and what its boundaries are. Whenever its boundaries are exceeded, the system needs to be adapted according to the new demands of the task environment. The ability to modify the functionality of a system based on one or multiple operators' commands is referred to as *Adaptability* (see Deliverable 3.3 – “Framework for Adaptation”). While adaptable automation systems do not need to be aware of the context nor able to adapt themselves to it, they require a human operator to take over these tasks. The ability to adapt oneself to new requirements posed by the context, also called *Adaptivity* or *Adaptiveness* (D 3.3), is considered the essence of intelligence [11]. Note that in contrast to adaptability, here the control of the behaviour modification lies entirely with the automation. In order to be recognized as intelligent, a system does not only need be sensitive to context requirements, but

also to display intelligent behaviour. That is, the system should be able respond to changing requirements in observable and adequate manner. This process is referred to as *Adaptation* (D 3.3).

### 4.3.2 Adaptive systems

In their framework of generic adaptive systems (see Figure 78), [3] provide a taxonomy of objects of adaptations that classifies how the automation can adapt. Adaptive systems can be sensitive to a number of factors called “adaptation triggers” [3]. In the framework, triggers are divided into operator human, system, world and task states and spatio temporal characteristics. In order to adapt, the automation can *distribute tasks* between agents. Also, the automation can modify the *user interface* (e.g. change communication channels), *content* or the nature of the task itself (e.g. map details, etc.) or the priority or *task scheduling*.





**Figure 78: Framework for adaptive human-machine-systems [3]**

### 4.3.3 Level of control

As pointed out in 4.2, the difference between adaptability and adaptivity is a question of authority or the level of control. The R-A-A (“Role-Agent-Authority”; [1]) framework provides a structured classification of adaptive components of intelligent adaptive systems. First, the role of the modified component will be defined by the task in the information processing cycle (cognitive loop) it fulfils. Next, the agent who takes the role (human or machine agent) is specified. Finally, the authority dimension denotes who controls the agent responsible for performing the task.

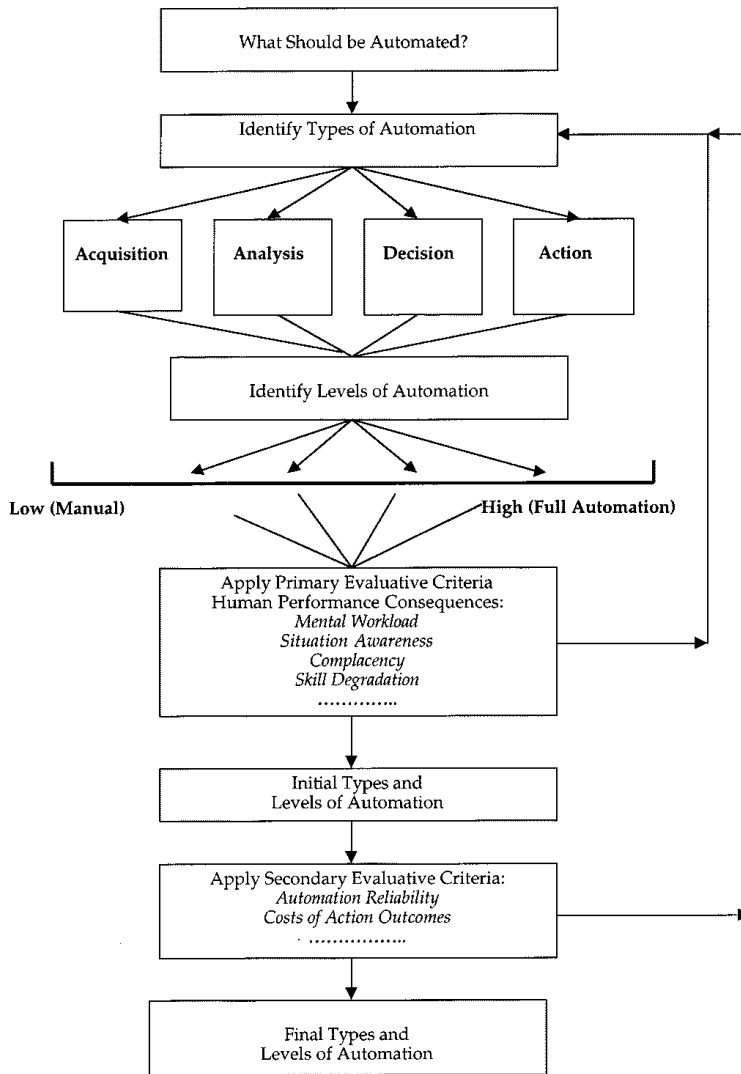


	<p><b>HoliDes</b></p> <p><b>H</b>olistic Human Factors <b>D</b>esign of Adaptive Cooperative Human- Machine Systems</p>	
---	---	---

#### **4.3.4 Automation**

Automation systems can be classified by means of the role or stage of the automated process in the information processing cycle (“type of automation”; [6]) and the level of authority the automation has in a task (“level of automation”; [6, 10]). For optimal aiding, fitting type and level of automation and associated human factors need to be considered in automation design.

The framework for automation design [6] already provides guidance in a number of issues that are discussed throughout this reports. Designers and developers from the field of (adaptive) automation design are therefore strongly encouraged to familiarize themselves with this classic model. It takes the reader stepwise through the design stages. First, the to be automated process has to be specified by means of the type of automation. Then, the appropriate authority distribution between machine and human agent is identified (level of automation). The concept will be evaluated with respect to human factors and adjusted and reevaluated iteratively if necessary.



**Figure 79: Framework for Automation Design [6]**

### 4.3.5 Classification of adaptive components

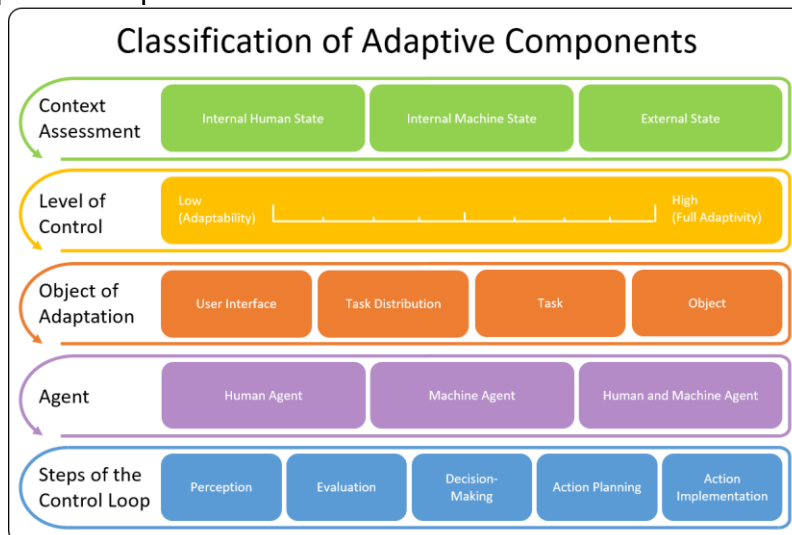
The presented adaptive systems [3], role-agent-authority [1], the automation design [5] and the HoliDes adaptation framework share significant overlap. In the context assessment category (“triggers”; [3], human and system states can be mapped onto the internal state, where the adaptation framework distinguishes between human and machine agents. Spatial aspects are an intrinsic part of the world or environment state while temporal characteristics can be accounted for in relation to task progress, which in the adaptation framework are both referred to as external states (task and environment). While the assessment of the external

context or the machine agent's states represent technology-centred approaches, this guideline will focus on the assessment of the human operator's state.

Modifications are described as *controlled objects* in the adaptation framework. Here, modifications in the function allocation and task scheduling category as described by Feigh et al. [3] can be mapped onto *task distribution*. Changes in the content will be referred to as changes in the *task* itself, *user interface* adaptations are described identically in both Feigh et al. [3] and the adaptation framework. Adaptations that do not fall in any of the categories described here will be classified as modifications of *objects*.

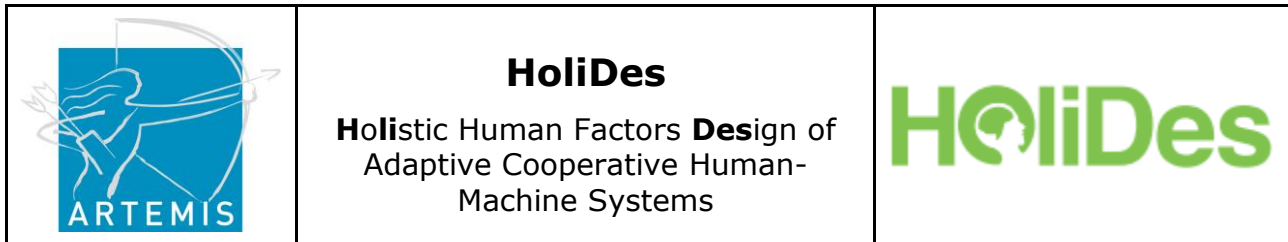
The role and type of automation dimensions of Banbury et al. [1] and Parasuraman et al. [6] are based on four stage information processing cycles while the adaptation framework allows for a higher resolution by splitting up the action stage into *action planning* and *action implementation*. The agent dimension of the role-agent-authority framework describes who (human agent, machine agent or both) is responsible for fulfilling the role, that is, *who closes the control loop* on the object of adaptation. The r-a-a and automation design frameworks' last categories (authority, level of automation) correspond to each other and provide a classification of the authority or control level of the adaptivity component on a scale ranging from adaptable to fully adaptive.

Taken together, we provide the following structural approach to define the loops for an AdCoS' adaptive components:



**Figure 80: Classification of Adaptive Components**

Working with this classification, the guideline presented in this report will help the designer to quickly translate his/her design ideas into cognitive loop primitives of



the adaptation framework and to thereby address human factors for all adaptive components. First, the designer should decide on what variable he/she wants to adapt the system to. This requires designers and developers to take a look into potential issues in the real-time assessment of context (internal operator state), so the automation can adapt to it and support the operator in critical situations. While doing so, the guideline needs to give recommendations on human factors for operator functional state assessment, what aspects of it can be used as “triggers” and how these can be extracted during an operation.



Next, it should be specified who has control over initiating the adaptation. Note that except for extremes of authority (full adaptability or full adaptivity), both agents need to be sensitive to the context trigger variable. Then the designer should find out what are the ideal means for adaptation to the changing context by choosing an object of adaptation and specifying who of the cooperation agents will be in charge of modifying that object. Automation systems adapting to the external or the machine agent’s state have been around for a while and shed light on a number of human factors challenges (e.g., “automation surprise”) that are of relevance for systems adaptive to operator states, too. While a few examples will be discussed later on, additional readings on automation systems (e.g., [6]) are recommended. Last, the component needs to be located at a process stage in the system operation cycle. This presents the interface between the adaptive loop of the component to be designed and other adaptive or executive loops of the entire system.

#### 4.4 ‘Five Ws and one H’

##### 4.4.1 Method

In general terms, automation can be designed to adapt all of its actions to all of the factors imaginable. It is up to the system designer to determine what design is the best fit for his/her purposes. After starting the design process with a loosely formulated objective and a classification, the designer should create a full and comprehensive story around the system to be built. Instead of introducing yet another framework the novice developer has to familiarize with, a well-known and intuitive technique to enrich the developer’s ideas is presented here. Originating from the field of journalism, the ‘Five Ws’ or ‘Five Ws and one H’ have become a popular technique used across research areas whenever a full picture needs to be created [7, 14]. The method consists of a set of questions that should be addressed with respect to the classification dimensions and answered in whatever order suits the situation. That way, the designer or developer substantiates his/her design ideas and is enabled to anticipate human factors by reflecting and reasoning about the design. The five Ws and one H are:

- What? (E.g. “What should be done?”)
- Why? (E.g. “Why should it be done?”)

	<p><b>HoliDes</b></p> <p><b>H</b>olistic Human Factors <b>D</b>esign of Adaptive Cooperative Human- Machine Systems</p>	
---	---	---

- Who? (E.g. “Who should it be done?”)
- Where? (E.g. “Where should it be done?”)
- When? (E.g. “When should it be done?”)
- How? (E.g. “How should it be done?”)

Note that the answers to these questions can only be as accurate as the formulation of questions themselves. The guideline’s dedicated focus is on the issue of adaptive automation; questions on what processes to automate or general guidelines on system design are not subject of the guideline. In the following, each dimension is explained for adaptivity and adaptation before a comprehensive overview of operator functional states and human factors in adaptive automation is given. Then, guideline design issues and their significance for the human factors are discussed. The chapter closes with the human factors guidelines for the design of AdCoS and a description of their integration in the adaptation framework.

#### **4.4.2 Five Ws and one H for Adaptive Components**

In this section we present a structured way for the acquisition of a full picture of the AdCoS to be designed. Rather than considering each question as a discrete design step, the guideline tries to convey a rich and holistic view of the system to the developer. Therefore instead of formulating specific questions for each dimension, developers are encouraged to think about the significance of each single dimension for the topic in question. Overlap between the contents of dimensions will be common and if not, developers should try to find connections between them.

##### **4.4.2.1 What?**

Deciding what an adaptive system should adapt to is one of the most vital parts of AdCoS design. Central to it is the question of which aspect of the internal human agent’s state moderates system performance to such an extent that the system should adapt itself depending on the aspect’s dynamics, if possible.

Operator functional states can be defined in a number of direct and indirect measures and constructs. Most commonly, operators’ functional capabilities are defined by means of workload, a mental construct that is supposed to reflect directly how much spare resources in terms of attention and working memory can be spend on a task [13]. In contrast, the second most popular trigger in the dimension operator state is performance, an indirect measure of operator functional state. Performance is defined by a behavioural variable that is related to an operator’s performance on the task in question. Its underlying assumption is that reduced functionality of the operator will have a direct effect on performance. The disadvantage however is that ideally operator functional state assessment will detect risks and hazards before performance decreases. Other criteria used as

indicators of an operator’s functional state are fatigue, emotional state or task engagement [9].

For adaptive automation, Schwarz et al. [9] recommend taking a holistic point of view rather than focusing on one single aspect. In their view, the operator functional state is not a unidimensional construct that can be defined according to situative needs, but results from the interplay of multiple variables such as workload, task engagement, emotions, attention, situation awareness and fatigue. Hence, when designing an AdCoS adaptive to more than one dimension of the operator’s states, the interrelations between dimensions need to be accounted for. Aside from context assessment, one or more system components or objects should be specified that will be modified dynamically. It is important to define the object itself, its function by means of steps within cognitive loop and its behaviour in terms of states that it can take and what they depend on. When defining what object and function to modify, taking a human point of view is most important. Rather than using technological feasibility as main criterion, human factors aware designers ask themselves what functions are the hardest to execute for human agents and what modification can be the most helpful to them. After deciding on a specific adaptation behaviour, interaction effects should be anticipated by describing interrelations between agents, adaptive and non-adaptive components in a control loop model.

#### **4.4.2.2 Why?**

If employed adequately, adaptive automation can have lots of beneficial effects such as reduction of operator workload, enhanced situation awareness, etc... However, suboptimal adaptation design can have negative consequences that easily outweigh the positive factors, e.g. when monitoring the adaptation creates more workload than is reduced by the automatically triggered higher level of automation. That an adaptive solution is feasible does not mean that it will be a better option than static automation. Therefore, when designing an AdCoS, the designer should make sure that there is an added value of making an automated system adaptive. Addressing the “Why” should not only focus on potential benefits, but also address risks of adding an adaptive component. The expected benefits should significantly outweigh costs to justify the vast increase in the system’s complexity.

Whenever a system adapts automatically, it should signal its new state to the operator in order to prevent automation surprise [8]. That however requires the operator to monitor the automation, which can cause additional workload. E.g., when the operator is overloaded, an adapting system might relieve the operator of less workload than it induces by forcing him/her to monitor the automation. Adapting systems can cause out-of-the-loop problems by dynamic function allocation, which asks for means to get the operator back in the loop. Ultimately adapting systems can lead to deteriorating operator skills as he/she does not have to control the system manually in critical situations.

#### 4.4.2.3 Who?

This part should be concerned with who triggers adaptive automation that is sensitive to the human agent's state. In single operator setups, this part deals with characteristics and capabilities of the anticipated operator. The developer needs to find out what operator requirements are critical for his/her design and what limitations might exclude potential users from operating the system. Ideally, the developer aims at a "design for all" to maximize the group of potential users [12]. Also, designers and developers have to specify whether the system is designed for single or multi operator setups. In single operator settings, the automation monitors one designated operator and adapts to his/her changing states when needed. However, this is greatly different for automation systems with collaborative interfaces. A good example is Use Case 5 of the Airbus Control Room AdCoS in WP 8, where the workload of multiple operators is balanced by shifting tasks from overloaded operators to their less stressed colleagues.

The designer needs to determine who will hold the authority over triggering the adaptation. As mentioned earlier, this should also depend on who of the agents involved in the cooperation is most sensitive to the trigger variable. If all agents are capable of detecting changes in the internal or external state, the designer should anticipate conflicting interpretations of the context what might lead to reliability and trust related issues.

The 'Who' also addresses what agents are responsible and what agents are affected by the adaptation. If multiple components will be adapted in response to a trigger, interaction effects need to be taken into account. Ideally, adaption takes place in a fashion that it brings the operator back below the threshold of incapacitation. That is, expected benefits have to be balanced with potential costs (e.g., taking the operator out of the loop) when designing an AdCoS.

#### 4.4.2.4 Where?

This questions deals with where the system will observe the trigger. For environmental and task triggers, this might be in the nature of the trigger; for operator states it is not as simple. Indicators for an operator's functional states can only be found in his/her behaviour, let it be neurophysiological or voluntary, e.g. speech production. For instance, abstract concepts such as mental workload have been measured by detecting changes in brain activity, pupil dilation, respiratory rate, eye movement, heart rate (variability), electro dermal activity, speech, and other ways [13].

When specifying the where, the developer should not only think about what type of (physiological) behaviour is most sensitive to changes in the selected trigger

variable, but also about restrictions imposed by the task environment. E.g., confined operating spaces do not allow for bulky sensor equipment, protective glasses can obscure eye movements and humid environments can affect electro dermal activity.

Not only the components responsible for making the adaptive systems sensitive to the trigger variable should consider the context, but also the adapting components. Take into account where in the AdCoS the adaptive components might connect to other components and what physical environment the adaption is embedded in. Direct effects of the adaptation on the AdCoS and the environment should be anticipated as well second order effects such as adaptive operator behaviour.

#### **4.4.2.5 When?**

The timing of the context assessment depends on the step of the control loop and the type of trigger. E.g., if the trigger can occur at any time of the operation, constant monitoring is recommended. If the monitoring method is intrusive or expensive, anticipated risks and costs of (constant) monitoring have to be considered.

Ideally, the adaptation is triggered before performance decreases. However, often the causes for drops in performance can only be observed in retrospect. The timeline should not only take an absolute perspective, but also take other system processes and their duration into account.



The timing of adaptation is a delicate issue that has to be treated with caution. If the automation adapts by itself, the operator needs to be aware in order to adapt his/her behaviour to the new context demands. Depending on the step of the control loop, the operator's function itself can change, which might introduce a different nature of human factors issues.

If automation adapts according to an external variable, adaptation states or thresholds need to be defined as adaptation triggers. Rather than chosen arbitrarily, these triggers should be tested empirically and closely related to performance. If the trigger variable changes values or states and thereby triggers adaptation almost constantly, this can be the cause of confusion and excessive workload for the human operator who needs to monitor the automation's state. In order to avoid such switching back and forth between automation states, tolerance ranges should be chosen based on experience and testing.

#### **4.4.2.6 How?**

After specifying when and where to observe the dynamics of what trigger, the context assessment methodology should be chosen at this point. The *Human Factors Method Library* (WP 1) will offer a variety of methods and tools to monitor



	<p><b>HoliDes</b></p> <p><b>H</b>olistic Human Factors <b>D</b>esign of Adaptive Cooperative Human- Machine Systems</p>	
---	---	---

the operator during the operation. The selected trigger narrows down the range of options, another important criterion is the ability to detect the trigger in real-time. Also, it should be specified how the object of adaptation will be modified. Often, this emerges from the choice of the trigger variable and the step in the control loop the adaptive component takes, e.g. increase display light when ambient lighting turns bright to support the operator in perception. If there is no obvious answer to how to ideally adapt the AdCoS, approach (potential) users of the design. Human factors requirements for adaptation are usually qualitative in nature and hard to translate into more quantitative terms.

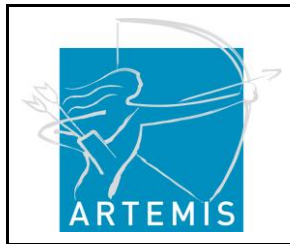
#### 4.5 Guidelines

The guidelines can be found in Annex III.

### 5 Requirements update

In this section, you will find an update version of the requirements with their final status. Requirements that not reflect final AdCoS use case have been removed. The full version of the requirements status included explanation and comments are in the annex IV.

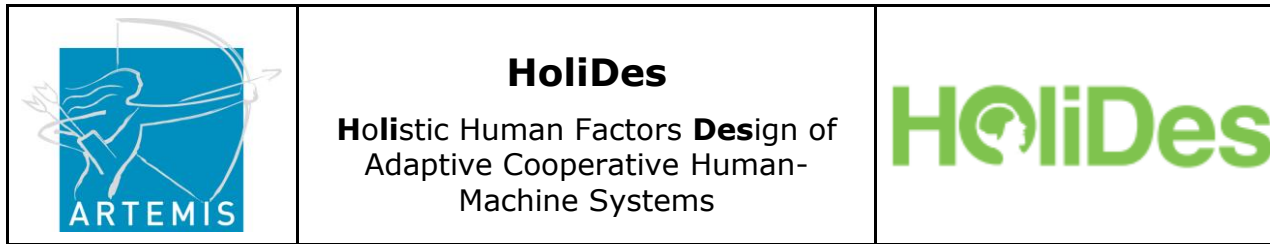
ID	Definition	Rationale	Final Status	available tools
WP6_PHI_HEA_REQ15	Dedicated guidance for the operator shall be provided via a display on the magnet that instructs the operator about the actions to take during positioning the patient	Effective guidance prevents mistakes that can lead to safety issues, discomfort, and low diagnostic quality	Covered by UC WP6_HEA_MRI_UC02_guided_patient_positioning . UC has been user tested. The new feature has not been released to the product yet, hence no feedback from real use.	In the context of the development of the use case tools from WP5 have been used (Means-end modelling, U_DAT and HF-Filter)
WP7_HON_AER_REQ28	The system should accompany the provided solution with explanation on why it was selected.	The system will display explanation of the provided decision aid to keep the user in the loop and optional re-evaluation of the solution.	Achieved	multidimensional optimization problem
WP7_HON_AER_REQ78	Create a tool/methodology that is able to classify an action of agent (human, machine) being either appropriate or erroneous. It is assumed that the tool has a task/procedure model with all supported alternate actions for a given situation.	At a given situation an agent may apply a number of actions. Some are correct, some may be erroneous. A generic classification against a defined procedure and accepted behavior is needed.	Achieved	multidimensional optimization problem
WP7_HON_AER_REQ84	Analyse and develop strategies for using the pupil information measured by eye-tracker in environment with - unstable level of illumination that can change rapidly - person changing often direction of view and focus	The parameters of the pupil are well related to the mental state, but are sensitive to eye accommodation and illumination. We need to know under which conditions pupil can be safely used or what algorithms and methods can be applied to filter out the workload relevant information.	Aborted	
WP7_HON_AER_REQ87	Define methods and tools for classification of measured physiological signal and related level of stress/workload. Do it in real time.	Real time classification of physiological inference of the pilot state is a prerequisite for any adaptivity based on the physiological measures.	Achieved	RTMAPS
WP8_ADS_C2_UC4.6	The system shall support the automatic start of a search for patterns in the database at a selected time.	To increase the rate that exploitable patterns are discovered.	Achieved	KNIME
WP8_ADS_C2_UC4.7	The system shall notify the supervisor when a pattern has been recognised during automatic search.	To allow the control centre to act on a possible exploitable behaviour.	Achieved	SIE



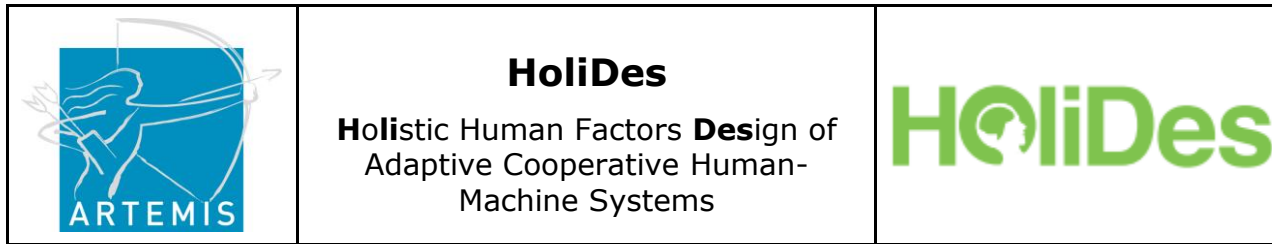
**HoliDes**  
**Holistic Human Factors Design of  
 Adaptive Cooperative Human-  
 Machine Systems**



WP8_ADS_C2_UC4.8	The system shall store historic data on all operators for at least two years.	Following a breach in security by enemy forces, it is considered prudent to be able to analyse historical data for lessons that can be learnt.	Achieved	SIE
WP9_CRF_AU_T_REQ03	The classifier of the driver cognitive state shall be able to do that with a CR <sup>3</sup> (80÷85)%.	The AdCoS is based on driver's status, so this classification is used for the adaptive strategies	Achieved	Driver's Distraction Classifier module
WP9_CRF_AU_T_REQ08	HMI shall be appropriate and distinguishable by the driver, with different channels and modes, depending on the internal (state) and external (environment) situation.	Different HMI strategies are required for warning and actuation, depending on drive's state.	Achieved	Adaptive HMI for AdCoS
WP9_CRF_AU_T_REQ09	When the driver has indicated his/her intention to change lane and there is not a side lane, or there is a side obstacle, or there is an incoming obstacle from the rear on the side lane, the driver should be warned so that he/she does not start the lane change maneuver. Driver's state shall be considered as well.	Driver support in lane changing or lane departure, depending on his/her status.	Achieved	Driver Intention Recognition module
WP9_CRF_AU_T_REQ10	When the vehicle aims at leaving the current lane (e.g. for an overtaking) the system shall assist her/him, indicating the right time and moment, taking into account the internal and external situation.	Driver support in lane changing or lane departure, for overtaking maneuver.	Achieved	MDP Co-pilot module
WP9_CRF_AU_T_REQ11	HMI shall be appropriate and distinguishable by the driver, with different channels and modes, depending on the internal (state) and external (environment) situation.	Different HMI strategies are required for warning and actuation, depending on drive's state.		
WP9_CRF_AU_T_REQ14	When the driver is changing lane in order to avoid a dangerous front obstacle, he/she should be supported in the lane change maneuver.	The goal of this function is to assist the lane change avoiding maneuver, taking also into account the driver's state.	Achieved	MDP Co-pilot module
WP9_CRF_AU_T_REQ16	When the driver is facing at the same time with more conditions that could generate an indication or a warning from the system, only the most critical indication should be given to the driver.	A prioritization is needed between several information, taking into account the driver's state.	Achieved	MDP Co-pilot module



WP9_OFF_AU T_REQ01	The parameters and structure of an initial Bayesian Driver model must be learned offline in order to classify a number of (yet to be defined) maneuvers, maneuver intentions, driving styles, and driving behaviors (e.g. steering wheel angle sequences) .	Since structure learning is computational expensive, offline structure learning is preferred. Furthermore, the use of dedicated datasets for offline parameter and structure learning guarantees a well-defined functionality prior to possible online parameter adaptation.	Achieved	Driver Intention Recognition module
WP9_OFF_AU T_REQ04	The Bayesian driver model must be able to return meaningful results after a fixed amount of computation time.	As computation time is limited, the Bayesian driver model must be able to return a meaningful result, even if the computation time is insufficient for exact inference. This can be achieved by the use of approximate inference techniques that can be interrupted to return preliminary results.	Achieved	Driver Intention Recognition module
WP9_OFF_AU T_REQ05	The Bayesian driver model must always be granted a fixed amount of time in order to return meaningful results.	Depending on the complexity of inferences, a certain minimal amount of computation time is required in order to produce first meaningful results. The actual guaranteed minimal computation time depends on the complexity of the model and the confidence in the approximation and will be specified during design time.	Achieved	Driver Intention Recognition module
WP9_OFF_AU T_REQ07	After an initial offline learning phase, the driver model must be able to classify the currently shown driving maneuver (e.g. lane-following, car-following) with a Correct Classification Rate (CCR) <sup>3</sup> of (80÷85)%.	The AdCoS is based on driver's status, so this classification is used for the adaptive strategies.	Achieved	Driver Intention Recognition module
WP9_OFF_AU T_REQ08	After an initial offline learning phase, the Bayesian driver model must be able to classify the driver's maneuver intention (e.g. lane-change) with a Correct Classification Rate (CCR) <sup>3</sup> of (80÷85)%.	The AdCoS is based on driver's status, so this classification is used for the adaptive strategies.	Achieved	Driver Intention Recognition module
WP9_OFF_AU T_REQ11	The Bayesian driver model must be able to provide its confidence in its maneuver classification.	Since the maneuver classification can be wrong and the AdCoS is using the maneuver classification of the Bayesian Driver model, the AdCoS must be provided a mean to assess the confidence in the classification.	Achieved	Driver Intention Recognition module
WP9_OFF_AU T_REQ12	The Bayesian driver model is able to provide its confidence in its maneuver intention classification.	Since the maneuver intention classification can be wrong and the AdCoS is using the intention classification of the Bayesian Driver model, the AdCoS must be provided a mean to assess the confidence in the classification.	Achieved	Driver Intention Recognition module



WP9_DLR_AU T_REQ01	After several manual driven overtaking maneuvers the driver model has learnt the natural driving behavior of the driver.	Since the driver model adapts the preference of maneuvers to the driver a learning phase is mandatory.	Achieved	Driver Intention Recognition module
WP9_DLR_AU T_REQ02	The driver model is able to improve stepwise over several overtaking maneuvers its current knowledge of the driver by considering inputs by the driver (steering angle, brake pedal position, throttle position) while driving highly automated. The driver model than updates its maneuver preferences.	Since the driver model adapts the preference of maneuvers to the driver a learning phase is mandatory.	Achieved	Driver Intention Recognition module

## 6 Conclusion

The formal grammar based on AdCoS primitives initially presented in previous deliverable and based on the Framework of Adaptation, has been ameliorated, with a deeper level of description. It is a first step to prove the capability to generate semi-automatically a kernel of Human Factor requirements during the early phases of design and modeling of adaptive systems. Taking into the current ontology/CMM developed by WP1 during the project, it can potentially be extended to fulfil a larger spectrum of issues if needed. It has been integrated in the Platform Builder.

The main idea of this framework is based on basic primitive functions (executive, adaptive loops) that we assemble together to build more complex models. It's the core concept of many collaborative and multi-agents approaches and the fundament of the "Mind" according to "Marvin Minsky" in the book "Society of Mind". A description of Holides use cases has been done using this framework for adaptation and it reveals generic enough to design any adaptive systems.

During this last year, integration and experiments have been pursued in particular in Aeronautical, Control rooms and automotive domains where whole resolution processes have been effective. These resolution processes are mainly focused on context assessment as it is the first prerequisite to allow decision making for any adaptation. Many learning algorithms have been used to assess dynamically the status of the operators according to the operational context. It shows that decisions are based on dynamic and adaptive computations of the context assessment instead of basic automations.

It could be noticed that common needs, transversal to domains have been identified, for some tested and for other planned in a near future. This is particularly obvious for perception (eyes tracking system), evaluation (learning machine techniques for classification).

## 7 References

- [1] Banbury, S., Gauthier, M., Scipione, A., Kanata, O. N., and Hou, M. 2007. Intelligent adaptive systems. Defence R&D Canada (DRDC) Toronto, CR, 75(269), 41.
- [2] Campbell, J. L. 1996. The development of human factors design guidelines. *International Journal of Industrial Ergonomics* 18, 5-6, 363-371.
- [3] Feigh, K. M., Dorneich, M. C., and Hayes, C. C. 2012. Toward a Characterization of Adaptive Systems. A Framework for Researchers and System Designers. *Human Factors* 54, 6, 1008-1024.



## HoliDes

**H**olistic Human Factors **D**esign of  
Adaptive Cooperative Human-  
Machine Systems

HoliDes

- [4] Lawall, J. L., Probst, C. W., and Schultz, U. P. 2006. Issues in holistic system design. *Proceedings of the 3rd workshop on Programming languages and operating systems: linguistic support for modern operating systems*.
- [5] Parasuraman, R. and Riley, V. 1997. Humans and Automation. Use, Misuse, Disuse, Abuse. *hum factors* 39, 2, 230–253.
- [6] Parasuraman, R., Sheridan, T. B., and Wickens, C. D. 2000. A model for types and levels of human interaction with automation. *IEEE Transactions on Systems, Man, and Cybernetics* 30, 3, 286–297.
- [7] Rich, C., Sidner, C. L., and Lesh, N. 2001. Collagen: applying collaborative discourse theory to human-computer interaction. *AI Magazin* 22, 4, 15–25.
- [8] Sarter, N. B., Woods, D. D., and Billings, C. E. 1997. Automation surprises. *Handbook of human factors and ergonomics* 2, 1926–1943.
- [9] Schwarz, J., Fuchs, S., and Flemisch, F. 2014. Towards a more holistic view on user state assessment in adaptive human-computer interaction. Proceedings of the 2014 IEEE International Conference on Systems, Man, and Cybernetics (SMC).
- [10] Sheridan, T. B. 1992. *Telerobotics, automation, and human supervisory control*. MIT press.
- [11] Simon, H. A. 1990. Invariants of human behavior. *Annual review of psychology* 41, 1–19.
- [12] Stephanidis, C., Antona, M., Savidis, A., Partarakis, N., Doulgeraki, K., and Leonidis, A. 2012. Design for All: Computer-Assisted Design of User Interface Adaptation. In *Handbook of human factors and ergonomics*, G. Salvendy, Ed. John Wiley & Sons, Inc, Hoboken, NJ, USA, 1484–1507. DOI=10.1002/9781118131350.ch54.
- [13] Young, M. S., Brookhuis, K. A., Wickens, C. D., and Hancock, P. A. 2015. State of science: mental workload in ergonomics. *Ergonomics* 58, 1, 1–17.
- [14] Zhang, Z., Wang, B., Ahmed, F., Ramakrishnan, I. V., Zhao, R., Viccellio, A., and Mueller, K. 2013. The Five Ws for Information Visualization with Application to Healthcare Informatics. *IEEE Transactions on Visualization and Computer Graphics* 19, 11, 1895–1910.