



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

HoliDes

**D4.2 – Plan for Integration of Model-Based Analysis
 Techniques and Tools**

Project Number:	332933
Classification:	Public
Work Package(s):	WP4
Milestone:	M1
Document Version:	V0.1
Issue Date:	13.06.2014
Document Timescale:	Project Start Date: October 1, 2013
Start of the Document:	Month 6
Final version due:	Month 7
Deliverable Overview:	Main document: D4.2 – Integration Plan of Model-based Analysis Techniques and Tools into the HF-RTP and Methodology - Public
Compiled by:	Nacho González - ATOS
Authors:	Nacho González – ATOS Jan-Patrick Osterloh – OFF Zdenek Moravek - HON Thierry Bellet - IFS Susanna Donatelli - UTO
Reviewers:	Morten Larsen - AWI Ales Smrcka - BUT
Technical Approval:	Jens Gärtner, EAD-DE-IW
Issue Authorisation:	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.

RECORD OF REVISION		
Date (DD.MM.JJ)	Status Description	Author
21.05.2014	First draft	Nacho González (Atos)
22.05.2014	GreatSPN contribution	Daniel Prun (ENAC)
23.05.2014	RTMaps, Usability evaluator, Concurrency bugs tools, Certification tool	Zdenek Moravek (Honeywell)
23.05.2014	CoSimECS contribution	Jan-Patrick Osterloh (OFF)
04.06.2014	First stable version	Nacho González (Atos)
11.06.2014	Version for Review	Jan-Patrick Osterloh (OFF)
12.06.2014	Integration of Reviews	Jan-Patrick Osterloh (OFF)



Table of Contents

List of Acronyms	7
1 Introduction	8
2 MTTs for Integration	9
2.1 Technique: Model Checking–Tool: GreatSPN (UTO)	9
2.1.1 Purpose	9
2.1.2 Use Cases	9
2.1.3 AdCoS Use-Cases	11
2.1.4 Input.....	12
2.1.5 Output.....	12
2.2 COSMODRIVE and COSMO-SIVIC (IFS)	13
2.2.1 Purpose	13
2.2.2 Use Cases	14
2.2.3 AdCoS Use Cases	15
2.2.4 Input.....	16
2.2.5 Output.....	16
2.3 Technique: Abstract interpretation – Tool: AnaConDa, Race Detector & Healer and SearchBestie (BUT)	17
2.3.1 Purpose	17
2.3.2 Use Cases	18
2.3.3 AdCoS Use-Cases	19
2.4 Input.....	19
2.5 Output.....	20
2.6 Formal simulation tool CoSimECS (OFF)	20
2.6.1 Purpose	20
2.6.2 Use Cases	22
2.6.3 AdCoS Use-Cases	23
2.6.4 Input.....	23
2.6.5 Output.....	24
2.7 RTMaps (INT)	25
2.7.1 Purpose	25
2.7.2 Use-Cases	25
2.7.3 AdCoS Use Cases	26
2.7.4 Input.....	27
2.7.5 Output.....	27
2.8 Usability evaluator (OFF).....	27
2.8.1 Purpose	27



- 2.8.2 Use Cases 31
- 2.8.3 AdCoS Use-Cases 34
- 2.8.4 Input 37
- 2.8.5 Output 38
- 2.9 Tools for profiling the concurrency bugs (BUT) 38
 - 2.9.1 Purpose 38
 - 2.9.2 Use Cases 38
 - 2.9.3 AdCoS Use-Cases 38
 - 2.9.4 Input 40
 - 2.9.5 Output 40
- 2.10 Certification tool (TEC) 40
 - 2.10.1 Purpose 40
 - 2.10.2 Use Cases 41
 - 2.10.3 AdCoS Use-Cases 41
 - 2.10.4 Input 42
 - 2.10.5 Output 42
- 2.11 Atos monitoring system 42
 - 2.11.1 Purpose 42
 - 2.11.2 Use cases 43
 - 2.11.3 AdCoS use cases 45
 - 2.11.4 Input & Output 45
- 3 References 48**

Table of Figures

- Figure 1: 10
- Figure 2: GreatSPN Use Case 12
- Figure 3: General architecture of COSMODRIVE Model 13
- Figure 4: Overview of the COSMO-SIVIC platform (COSMODRIVE + Pro-SIVIC + RT-MAPS) 15
- Figure 5: CoSimECS: DCoS Specification View 20
- Figure 6: CoSimECS - Simulation View 22
- Figure 7: Meta-Model for CoSimECS (DCoS-XML based) 24
- Figure 8: Overview RTMaps Use-Case Aeronautics Domain 27
- Figure 9: UML Use Case Diagram of the Human Efficiency Evaluator 28
- Figure 10: Human Efficiency Evaluator as part of a development process based on the V-Model. 30
- Figure 11: UML Activity Diagram of the Human Efficiency Evaluator overall activities. 32
- Figure 12: Exemplary performance comparison of different design or adaptation variants 33

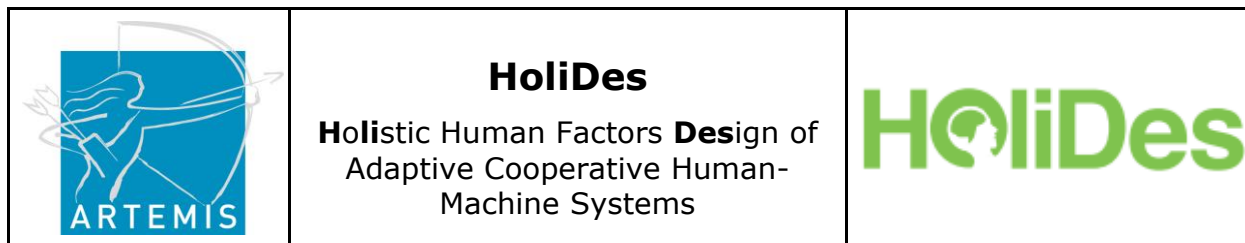




Figure 13: Exemplary workload prediction of the human operator’s visual perception34

Figure 14: Monitoring System Overview43

Figure 15: Monitoring System Architecture.....47

	<p>HoliDes</p> <p>Holistic Human Factors Design of Adaptive Cooperative Human- Machine Systems</p>	
---	--	---

List of Acronyms

AdCoS	Adaptive Cooperative Human-Machine Systems
CASCaS	Cognitive Architecture for Safety Critical Task Simulation
DCoS	Dynamic Distributed Cooperative System
HCD	Human Centred Design
HF-RTP	Human Factors Reference Technology Platform
HMI	Human-Machine Interface
HoliDes	Holistic Human Factors Design of Adaptive Cooperative Human-Machine Systems
MTT	Method, Technique and/or Tool
PED	Procedure Editor developed by OFFIS
UC	Use-Case
WP	Work Package

1 Introduction

This document gives details on the integration of MTTs into the HF-RTP for WP4. Please refer to the common Integration Plan document for further details on the HF-RTP and possible integration techniques.

The objective of WP4 is to develop techniques and tools for model-based formal simulation and formal verification of Adaptive Cooperative Human-Machine Systems (AdCoS) against human factor and safety regulations.

As described in D4.1, verification and validation are two system engineering technical processes (ISO IEC 2008). Verification tries to answer the question "Are we building the system right?", and validation deals with final user and operational related requirements, trying to answer the question "Are we building the right system?". **Model-based analysis** is an approach to support verification and validation processes. The idea is to construct an intermediate representation of the future system – the model – and to search for evidences directly on this representation. With this approach, evidence can be a mathematical demonstration or a global observation performed on all possible states of the system, e.g. with formal verification techniques. More details on model-based analysis can be found in D4.1.

The modelling-languages and their editors are defined in WP2, and instantiated in WP4 for the model-based analysis. WP3 will add adaptation techniques to the models. In the first cycle, WP4 will mainly focus on the automotive AdCoSs for demonstration purposes. In the next cycles, the MTTs of WP4 will be extended to other domains as well.

The following sections describe the initial set of MTTs used within WP4, which are provided to the other work packages. Many of them are coming from WP2 (Modelling Techniques and Tools work package), as they are developed within WP2 and applied in WP4.

2 MTTs for Integration

2.1 Technique: Model Checking–Tool: GreatSPN (UTO)

2.1.1 Purpose

GreatSPN is a software framework for modelling, verifying and evaluating performance measures on systems using Generalized Stochastic Petri Nets (GSPN). The framework is composed of several tools, including a Java-based GUI that allows the modeller to draw an abstract representation of the modelled system using the GSPN formalism.

The framework supports the following functionality:

- Structural analysis – derive properties that only depends on the structure of the nets and not on the initial state, therefore properties that are either true or false for all possible initial marking, as place and transition invariants, deadlocks, place boundedness, mutual exclusion.
- Properties derivable with linear programming - such as upper and lower bounds for number of tokens in places and for transition throughputs.
- State space generation, using state-of-the-art techniques, like decision diagrams.
- Verification of logical and behavioural properties
- Numerical analysis of quantitative properties - such as average place distributions, expected transition throughputs, probability of certain model behaviour (stochastic model-checking of the CSLTA logic
- Simulation techniques available for models for which the construction of the reachability graph is impracticable.
Optimization problem, described by an extension of Petri net called Markov Decision Petri Nets that have a semantic interpretation as Markov Decision Processes (MDP).

2.1.2 Use Cases

In general the tool can be used to model and validate reasonably large finite state machines. For the aeronautics domain a special use-case of 'creating an experimental scenario' is proposed that should make use of the simulation/evaluation capability of the tool. If proved feasible, this use-case can be applied cross-domain.

Building an experiment scenario is a cooperation of the experimenter and the operational expert (a pilot) with use of a simulator. They work together to derive initial parameters (such as position, speed, flight plan, en-route events, other traffic etc.) in order to invoke a specific situation in the experiment, for which they want to collect data.

The work is sometimes cumbersome, as inversion from behaviour to parameters may be ambiguous and the task is done in several iterations. This is especially true for adaptive systems when a particular adaptive feature should be tested.

Our plan is to use GreatSPN to alleviate this problem, as a model can be created and with the tool support we plan to do

- Verification of formal logic consistence of the model (i.e. no deadlocks, place accessibility etc.)
- Sensitivity analysis, in particular
 - Verification of behaviour for various values of input parameters (brute-force)
 - Derivation of values of input parameters that trigger a specific behaviour of the system.

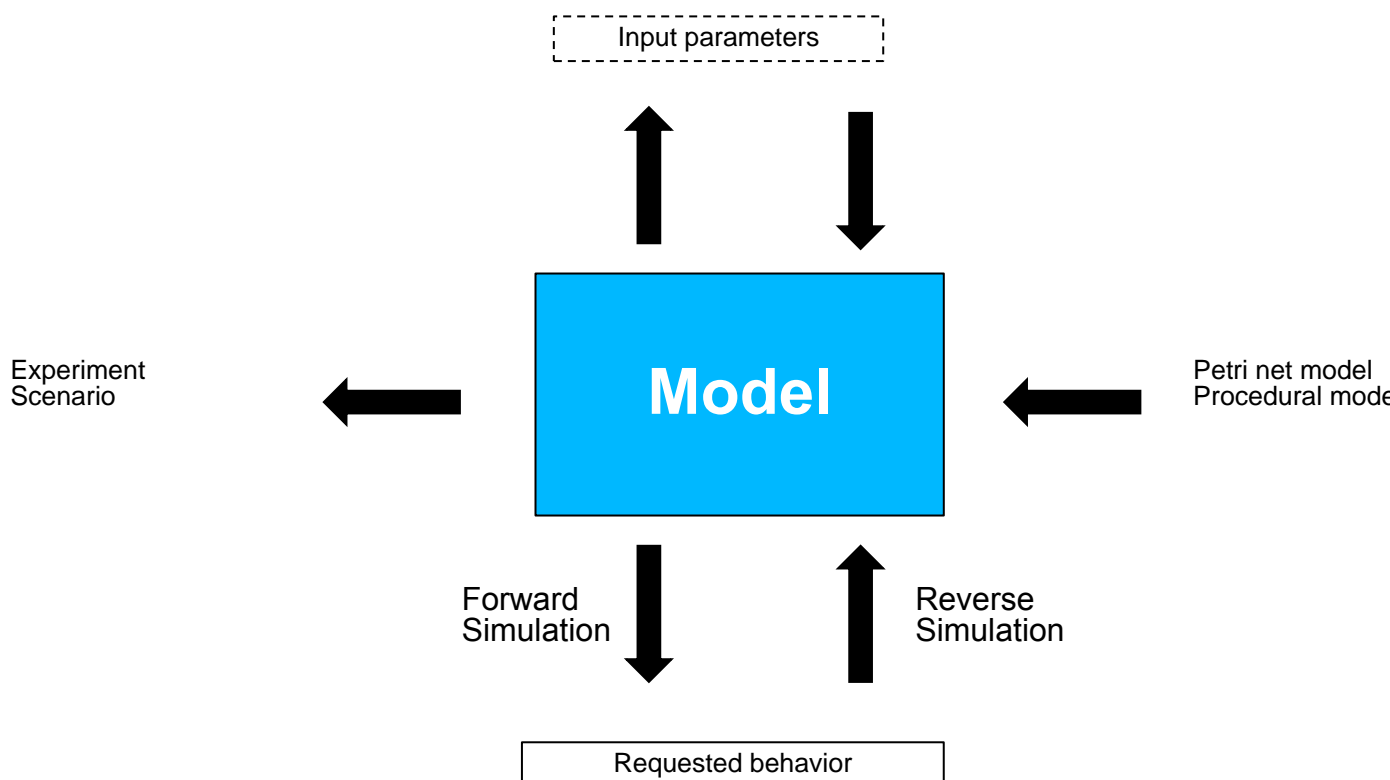


Figure 1: Model-based sensitivity analysis

2.1.3 AdCoS Use-Cases

The user will benefit from the following use-cases:

- Create model scenario - The user creates a model of the experiment scenario. The system provides graphical building blocks of the model and allowed links between blocks. The system also provides basic information about the syntax and meaning. The user picks up the building blocks and constructs the model. The user can annotate building blocks with rationale / explanation. The system verifies the validity of the model on the fly and warns if the user tries to do invalid operations.
- Simulate scenario - The user creates or loads a model, and defines simulation type. The system provides test configuration screen in which the user can manipulate inputs for the test (manipulate inputs UC).
- Manipulation of inputs and simulation may happen interactively - when user defines the inputs, the system processes the model and displays results in forms of color-coded text. There is a number of test types available.
- Manipulate inputs - The system provides a form, in which user can write values for specific variables. The user can change names of variables and add new ones. The user also connects the variables to the model. The systems store the list of variables and their attributes (value, ranges etc.).
- Evaluate logic consistence - The user creates a model and then applies the consistence cross-check. The system knows formal rules of processing to each type of model. It reads in a model and applies the rules to check the validity. For instance, it searches for dead-ends, and infinite loops.
- Report - When the user is satisfied with the performance of the model, the system creates a report. According to the situation, the system may validate the model (evaluate logic consistence UC) and perform simulations to gather the report data (simulate scenario UC). The system also converts the model into a scenario draft (create scenario draft UC).
- Create scenario draft - The system converts the model into human readable form. The user can specify optimum values for the critical input variables (manipulate inputs UC) and the system adds them in the draft. The conversion may be specific to each scenario and the user can define it within the system.

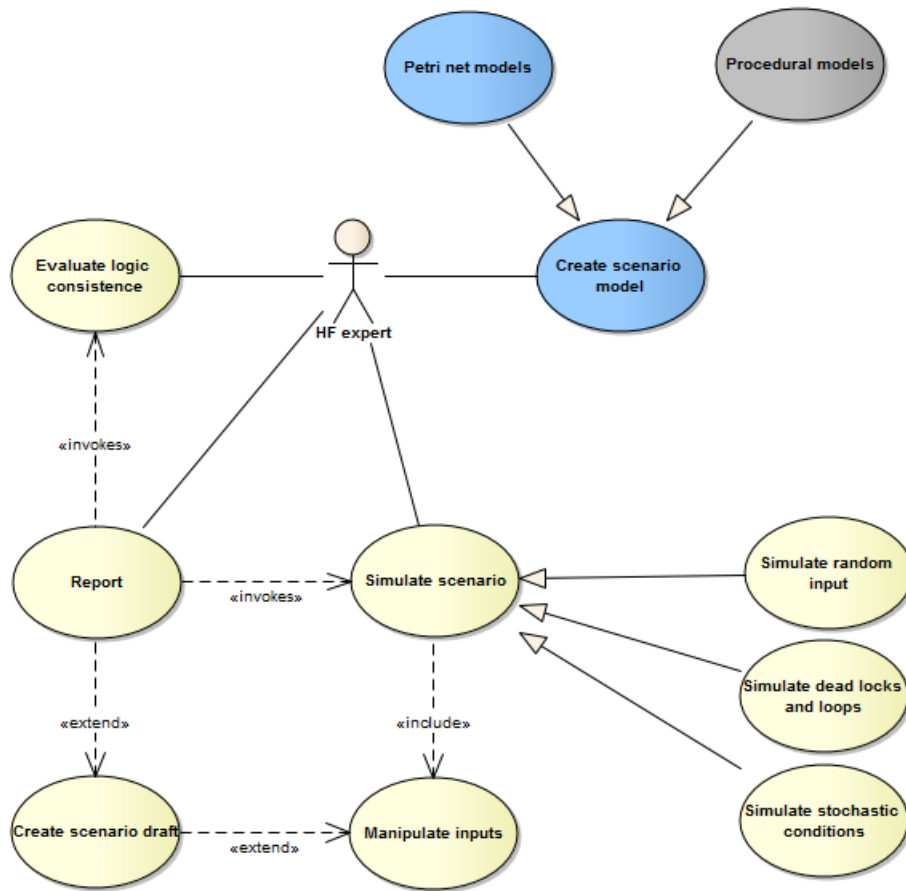


Figure 2: GreatSPN Use Case

2.1.4 Input

Inputs is in the form of a Petri net model, either defined by an expert of the domain (possibly an expert having also some modelling experience) using the GreatSPN GUI or through automatic translation from other high level state-based formalisms (like UML statecharts). We can also envision having a database of models that can either be re-used or combined to build new models.

2.1.5 Output

Result of the use-case is a scenario report – description of how an experiment will be conducted. The tool has no reporting ability, but it would be useful to have one specific for this use-case. Such reporting should read the model and evaluation results (as XML) and build a textual description. All these functionalities are not already present in GreatSPN

and it is an open research issue, what is the best approach to pursue this objective.

2.2 COSMODRIVE and COSMO-SIVIC (IFS)

The main modelling tool to be specifically developed by IFSTTAR in WP2 of HoliDes is a COgnitive Simulation Model of the car DRIVER (named COSMODRIVE). Using mental representations, this model allows building, simulating and visualizing part of the driver's situational awareness. A prototype was implemented based on previous work, and a fully functional version of the tool is scheduled for this project.

Then, in the frame of WP4, this driver model will be interfaced by IFSTTAR with 2 additional tools, that are Pro-SIVIC (provided by CIVITEC) and RT-MAPS (coming from INTEMPORA). The challenge will be to have, at last, a *virtual Human Centred Design (HCD) platform* of AdCoS (named COSMO-SIVIC). Finally, this virtual platform will be used in WP9 (Automotive application domain) in order to virtually design and test future AdCoS, before their implementation on real cars.

2.2.1 Purpose

The general objective of COSMODRIVE model is to virtually simulate the human drivers' perceptive and cognitive activities implemented when driving a car, through an iterative "Perception-Cognition-Action" regulation loop.

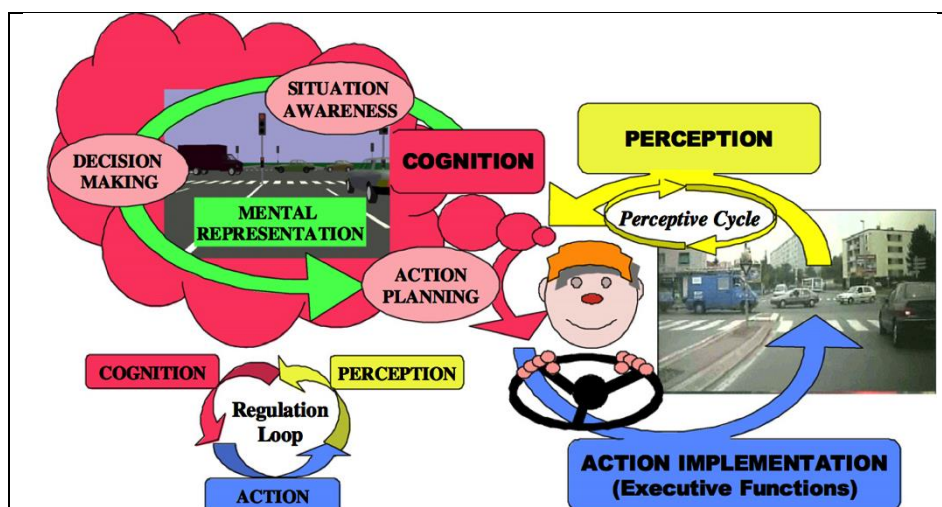




Figure 3: General architecture of COSMODRIVE Model (Bellet et al, 2012)

	<p>HoliDes</p> <p>Holistic Human Factors Design of Adaptive Cooperative Human- Machine Systems</p>	
--	--	---

Through this main regulation loop, the model allows to:

- Simulate human drivers perceptive functions, in order to visually explore the road environment (i.e. *perceptive cycle* based on specific driving knowledge called “schemas”; Bellet et al, 2007) and then to process and integrate the collected visual pieces of information in the Cognition Module.
- Simulate 2 core cognitive functions that are (i) the elaboration of *mental representations* of the driving situation (corresponding to the driver’s Situational Awareness; Bellet et al, 2009) and (ii) a *decision-making* processes (based on these mental models of the driving situation, and on an *anticipation process* supported by dynamic mental simulations)
- Implement the driving behaviours decided and planned at the cognitive level, through a set of effective actions on vehicle commands (like pedals or steering wheels), in order to dynamically progress along a driving path into the road environment.

Moreover, the aim of the use of COSMODRIVE model in HOLIDES project is not only to simulate these perceptive, cognitive and executive functions in an optimal way, but also to simulate some human drivers errors in terms of misperception of event, erroneous situational awareness, or inadequate behavioural performance, due to visual distractions (resulting of a secondary task performed during driving, for instance).

2.2.2 Use Cases

In the automotive domain, IFSTTAR objective will be to interface (in WP4) the COSMODRIVE model (developed in WP2) with Pro-SIVIC and RT-Maps software, in order to support (in WP9) virtual simulations of future AdCoS use by human drivers (as assessed via COSMODRIVE).

The following figure provides an overview of this future COSMO-SIVIC platform to be developed during the project:

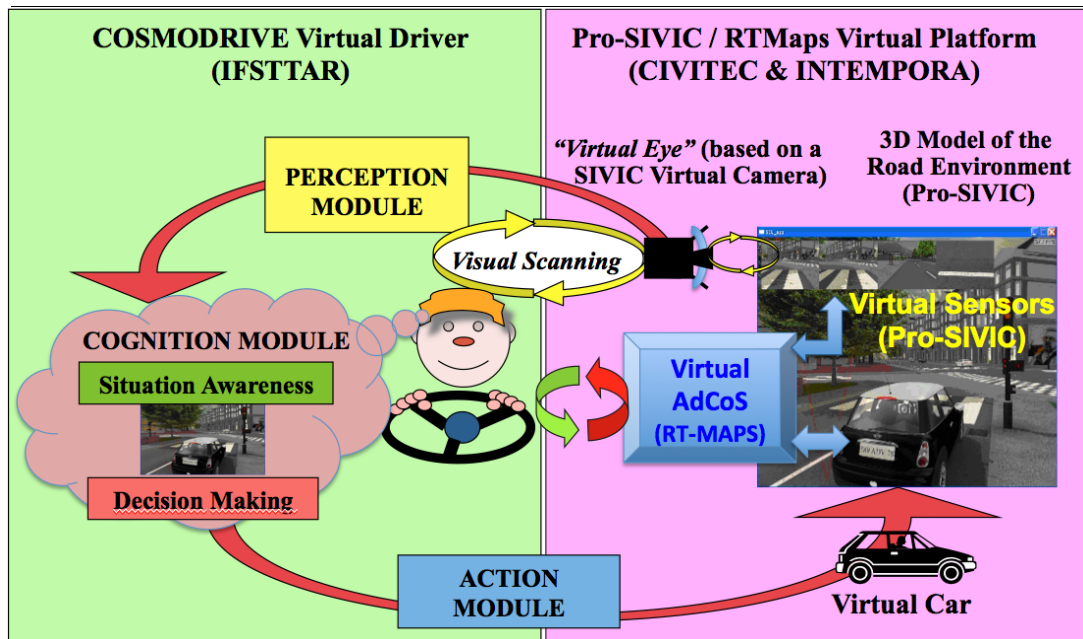




Figure 4: Overview of the COSMO-SIVIC platform (COSMODRIVE + Pro-SIVIC + RT-MAPS)

From the work to be done in WP2 and in WP4 - in partnership with CIVITEC and INTEMPORA - it is expected to have a last a virtual Human Centred Design platform integrating (1) a human driver cognitive model (i.e. COSMODRIVE) with a virtual eye simulating real drivers visual scanning (adapted from a SIVIC virtual camera) and able to drive dynamically (2) a virtual car (3) into a virtual 3-Dimensional road environment (both simulated with Pro-SIVIC).

2.2.3 AdCoS Use Cases

Moreover, Pro-SIVIC and RT-MAPS will be also used by IFSTTAR in WP4, in order to implement and to simulate future virtual AdCoS, in a realistic way (by using Pro-SIVIC virtual sensors, RT-MAPS functionalities, and also integrating Monitoring Functions to be developed by IFSTTAR in WP3, and in charge to adapt the driving Aids according to human errors and/or the driving context/conditions).

From this integrative COSMO-SIVIC platform presented in Figure 4 (COSMODRIVE + ProSIVIC + RT-MAPS), it will be possible to simulate both driving performance of a human driver without any AdCoS (from normal behaviours to critical behaviour due to visual distraction) and driving performance of a "Human-Machine System" integrating jointly a

	<p>HoliDes</p> <p>Holistic Human Factors Design of Adaptive Cooperative Human- Machine Systems</p>	
---	--	---

driver model (COSMODRIVE) and a virtual car equipped of a simulated AdCoS.



According to these set of functionalities, the use of the COSMO-SIVIC Model-Based platform for a Human centred design approach of future AdCoS will mainly concerns two main stages of the design process. At this earlier design step, the COSMODRIVE model will be used to estimate human drivers' performances in case of unassisted driving, in order to identify and specify critical driving scenarios for which a given AdCoS could be provided for supporting the driver. These critical scenarios will correspond to driving situations, identified from simulation, for which the human drivers' reliability (as assessed with COSMODRIVE performance and driving error simulations) seems not sufficient for avoiding accident, or to adequately manage the situational risk. Through these simulated scenarios, it will be possible to provide ergonomics specifications of the human driver needs, in terms of AdCoS functions and/or adaptation strategies (regarding the driving conditions, for instance). Then, during the AdCoS evaluation phases, coming later in the design process, it will be possible to virtually evaluate this AdCoS *effectiveness* for the critical scenarios previously identified, in order to check and to measure the *efficiency* of this AdCoS in these particular critical driving conditions. This will be done using the COSMO-SIVIC virtual HCD platform.

2.2.4 Input

In the frame of the COSMO-SIVIC platform, COSMODRIVE inputs will come (1) from the virtual road environment, as perceived by the virtual eye of COSMODRIVE (to be implemented during the HoliDes project by using a SIVIC virtual camera), and (2) from a virtual AdCoS, to be simulated with Pro-SIVIC and RT-MAPS. In this perspective, several requirements were already done by IFSTTAR among HoliDes partners / tool developers (like WP9_IFS_AUT_REQ03: "Data synchronization coming from different simulation tools"; WP9_IFS_AUT_REQ04: "Having a virtual car able to be dynamically piloted by the driver model", WP9_IFS_AUT_REQ10 (REQ12): Virtual simulation of car sensors (radar, camera, telemeter), as components of AdCoS, to be simulated and tested in WP9).

2.2.5 Output

At the COSMODRIVE level, the outputs generated by this driver model will primarily concerns driving behaviours in terms of both drivers' actions on

	<p>HoliDes</p> <p>Holistic Human Factors Design of Adaptive Cooperative Human- Machine Systems</p>	
--	--	---

vehicle controls (pedals and steering wheel) and human drivers visual scanning simulation (including simulation of visual distraction risks/effects; a collaboration with ERGONEERS is under discussion regarding this specific issue, in order to use COSMODRIVE for generating Eye Tracking data liable to be processed in Ergoneers D-Lab software; cf. WP9_IFS_AUT_REQ07: "Recording/using of eye-tracking data to assess driver' visual distraction").



With COSMO-SIVIC virtual HCD platform, it is then expected to simulate interactions between COSMODRIVE and virtual AdCoS, and also to evaluate the driving performance of the "Human-Machine System" as a whole (in the frame of both normal and critical driving situations/scenarios). Regarding human centred design issues (Bellet et al, 2011), one of the core objective of IFSTTAR will be to compare drivers performance without *versus* with AdCoS (or with different types of AdCoS), in order to virtually assess future AdCoS effects (in terms of both "benefit" versus "new risks" liable to occur) before implementing them on a real car. For that, it will be needed to define in WP4 specific measures and/or metrics of COSMO-SIVIC outputs for assessing AdCoS *Efficiency* (based on an "Engineering" point of view: does and work the AdCoS as "technically expected") and *Effectiveness* (based on an "Ergonomics" point of view: does the AdCoS provide an assistance to the Human that really corresponds to the "Aid function" initially specified in order to support their real needs).

2.3 Technique: Abstract interpretation – Tool: AnaConDa, Race Detector & Healer and SearchBestie (BUT)

2.3.1 Purpose

Abstract interpretation:

Creating a model of a system in order to verify the system under investigation is time consuming and, in addition, can be a source of errors (spurious errors can be introduced into the model during the modelling while some real errors can be hidden, for instance, by the model abstraction). In later phases of software development when source code and/or executable application is available, other approaches to system verification may be considered. Actually, the system itself can be understood as its own model or a model can be inferred from the system automatically. Such approaches are exploited in model checkers for common programming languages like Java exploited in model checkers for

	<p>HoliDes</p> <p>Holistic Human Factors Design of Adaptive Cooperative Human- Machine Systems</p>	
---	--	---

common programming languages like Java (Visser, Havelund, Brat, Park, & Lerda, 2003) or C (Henzinger, Jhala, Majumdar, & Sutre, 2003; Clarke, Kroening, Sharygina, & Yorav, 2005) as well as in various kinds of static analysis (Kam & Ullman, 1976; Cousot & Cousot, 1992; Nielson & Nielson, 1999).

Another issue for verification is concurrent software that is now very popular due to wide usage of multi-core processors and multi-processors. Verification of concurrent systems is more difficult due to huge number of possible process interleaving. It causes that precise approaches like model checking do not scale well while common testing methods are not able to reveal rarely manifesting bugs. In order to increase chances to spot these errors, techniques such as injection of noise into the scheduling of concurrent processes have been proposed and supported by tools like IBM ConTest (Edelstein, Farchi, Nir, Ratsaby, & Ur, 2002) or ANaConDA (Fiedor & Vojnar, 2012). Furthermore, various dynamic analyses have been proposed. They are used to analyse the behaviour seen in a testing run in order to detect concurrency bugs like data races or deadlocks.



AnaConDa, Race Detector & Healer and SearchBestie

These tools are provided by VeriFIT research group from BUT and are intended to support checking of concurrent software.

ANaConDA is a framework that simplifies the creation of dynamic analysers for analysing multi-threaded C/C++ programs on the binary level. The Java Race Detector & Healer is a prototype for run-time detection and healing of data races and atomicity violations in concurrent Java programs. SearchBestie (Search-Based Testing Environment) is a generic infrastructure that is designed to provide environment for experimenting with applying search techniques in the field of program testing (e.g. to find optimal settings of injected noise to increase efficiency of AnaConDa and Race Detector & Healer).

2.3.2 Use Cases

AnaConDa + EventReader analyser: EventReader analyser is a plugin for AnaConDa tool which, provided a binary to analyse, logs to the standard output internal events of a program run. The events are for instance memory reads and writes, lock acquisitions, and exception throws. The log can then be analysed for task-specific properties.

	<p style="text-align: center;">HoliDes Holistic Human Factors Design of Adaptive Cooperative Human- Machine Systems</p>	
--	---	---

AnaConDa + AtomRace analyser: AromRace analyser is a plugin for AnaConDa tool which, provided a binary to analyse, monitors for non-atomic operations on variables and possible data races.

Race Detector & Healer: Java program under test can be analysed via the tool and the tool automatically with no user intervention provides a run-time detection of data races and atomicity violations.

SearchBestie: Let, in a case to be analysed, there are multiple tests of an application, each has multiple parameters of different type and one wants to find the best (in a given circumstances) combination of tests and their parameters. There are plenty of possible combinations. Search and analysis engines in SearchBestie helps to automatically explore state space of possible combinations in a reasonable way and find a combination (or a set of combinations) that fits ones requirements given by some fitness function.

2.3.3 AdCoS Use-Cases

The technique and tools are applicable for any system with concurrency and Java/C/C++ programming language used. The tools targets safety issues concerning concurrency bugs.

In particular, AnaConDa or Race Detector & Healer (depending of type of program under test) will be used in the Aeronautics within Electronic Flight Bag AdCoS, Use-case 1 "Airport Diversion Assistant" (cf. Deliverable 7.1, Section 2.1). The tools will be used for evaluation of the system code of an automated planning of diversion in case of emergency, in particular in evaluation of correct implementation of parallel tasks.

2.4 Input

For software of (part of) AdCoS, the input are binaries either for Intel processors or for Java Virtual Machine.

To properly perform purpose-specific analysis (in particular for SearchBestie), selected properties to be checked must be provided (refined from requirements such that they can be checked with this approach).

2.5 Output

The outputs are of two types. Either it is a data from simulation / testing in form of textual log files, or, in a case of found bugs, program traces, and/or memory or variable identifications.

2.6 Formal simulation tool CoSimECS (OFF)

2.6.1 Purpose

CoSimECS is a tool for Composing, Simulating and Evaluating distributed Cooperative Systems, and will be extended during HoliDes to cover also adaptive systems. For this, it allows to specify a system in terms of agents, tasks and resources including the relations between these components (i.e. the possible task and resource allocations).

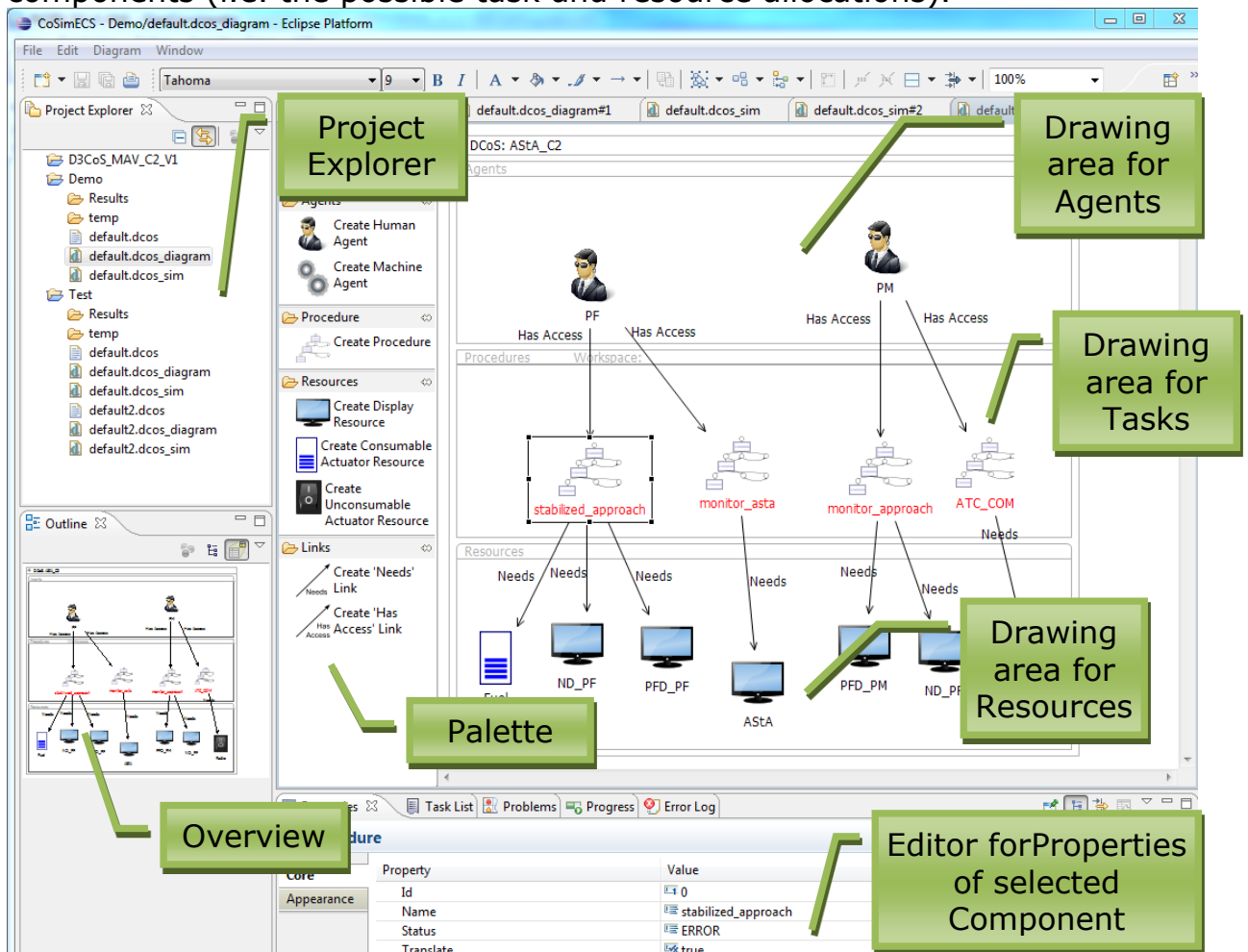
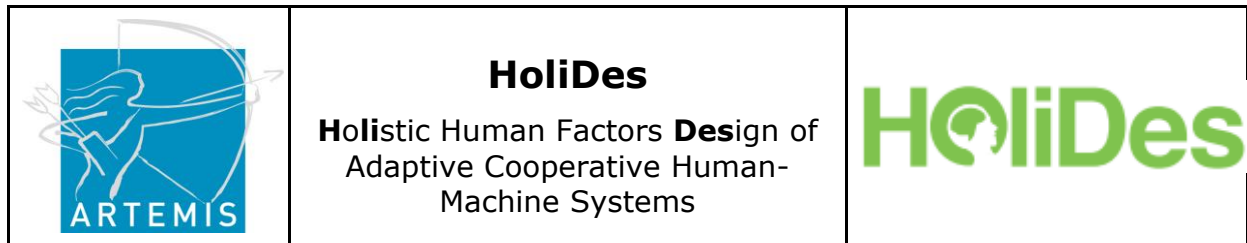


Figure 5: CoSimECS: DCoS Specification View



CoSimECS also allows setting up and controlling the simulation, based on the OFFIS simulation platform and CASCaS (Cognitive Architecture for Safety Critical Task Simulation). The OFFIS simulation platform (Puch et al. 2012) is based on the High Level Architecture (HLA) standard (IEEE1516). The configuration files for the HLA simulation are created from the composed system, and automatically deployed to the simulators. Figure 6 shows the editor for the simulation in CoSimECS. On the right side of the editor, the agents and resources of the Dynamic Distributed Cooperative System (DCoS) are shown. The tasks are input for the simulators, and don't need to be instantiated by a tool. Then, in the middle part of the editor, the user can model the simulation, by adding concrete simulation computers and the simulation tools installed on them, e.g. CASCaS for modelling of human agents (see also D2.2 for CASCaS). For the simulation of machine agents and the resources, different tools can be used (via a simplified HLA interface provided by the OFFIS Virtual Simulation Platform):

- Matlab Simulink
- Silab (a driving simulator)
- X-Plane and FSX as flight simulators, and
- any tool that uses the C-Interface of the HLA interface.

At the left part of the editor, an overview on the current simulation setup is displayed, when the simulation is started. When a simulation is finished, the output of the simulation tools is collected and stored for further analysis in a structured form.

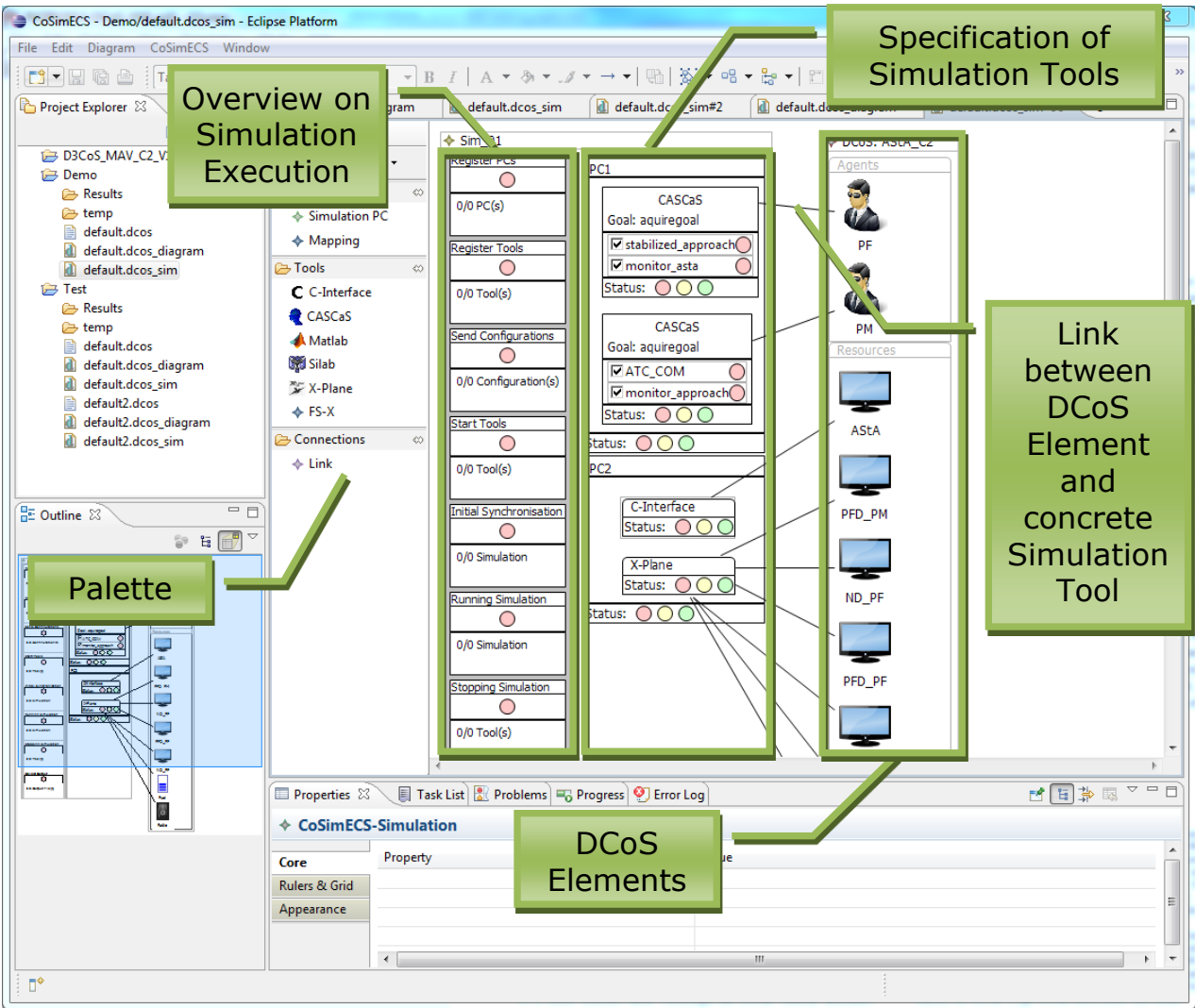




Figure 6: CoSimECS - Simulation View

2.6.2 Use Cases

2.6.2.1 Virtual Human-in-the-Loop Evaluation Use-Case

CoSimECS allows fostering the use of the cognitive architecture CASCaS for the evaluation of an adaptive system via simulation. The workflow is as follows:

1. Specification of the adaptive system (Agents, Tasks and Resources)
2. Specification of the tasks (link to the PED tool)
3. Specification of the simulation (Simulators to use for agents and resources; simulation may be distributed in a network)

	<p>HoliDes</p> <p>Holistic Human Factors Design of Adaptive Cooperative Human- Machine Systems</p>	
---	--	---

4. Execution of the simulation with virtual human in the loop, including collection of simulation output
5. Definition of Analysis (Gaze Distribution, Scanning Behaviour, Task Execution Time, ...)
6. Execution of Analysis and report creation (log files)
7. Interpretation of results and system improvement

2.6.3 AdCoS Use-Cases

This MTT will be used in the Aeronautics Use-Case 1 “Airport Diversion assistant” for evaluation of the system (cf. Deliverable D7.1, Section 2.1).

2.6.4 Input

No initial input (e.g. from an RTP) needed per se, i.e. the user of the tool starts with an empty model and defines it during use of the tool. An import for agents, tasks and resources may be possible to implement in future version, when connected to the RTP, i.e. from UML descriptions. Figure 7 shows the meta-model behind CoSimECS. The meta-model is based on the DCoS-XML, which has been specified in D3CoS.



HoliDes

Holistic Human Factors Design of Adaptive Cooperative Human-Machine Systems

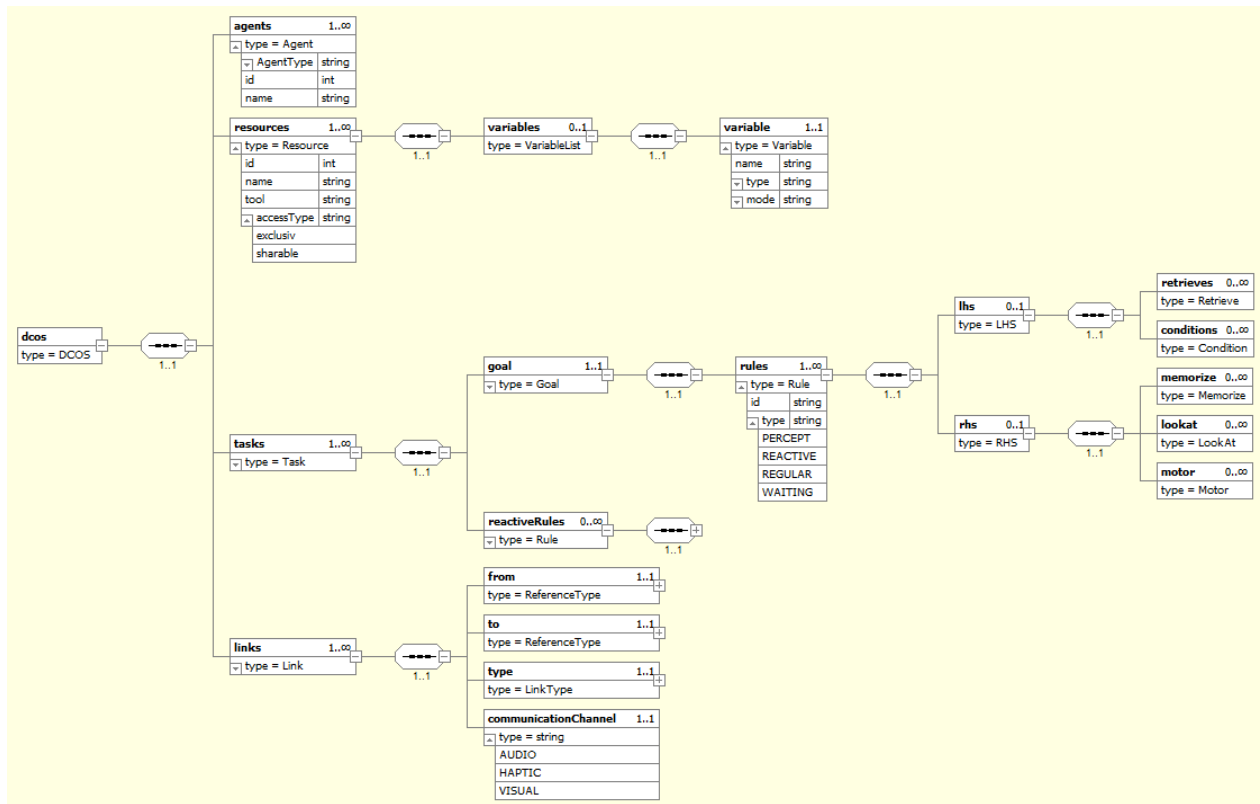




Figure 7: Meta-Model for CoSimECS (DCoS-XML based)

2.6.5 Output

Output of the CASCAs simulation is:

- Log-file for each CASCAs instance and each simulation participant
- CASCAs record of each variable of a resource either subscribed or published by CASCAs, including selected rule, selected goal, goal agenda, gaze, and motor actions.
- Picture of working memory content at last step of CASCAs
- XML description of CASCAs working memory content

This can be analysed in several aspects, e.g. task execution time, reaction times, gaze distribution, workload, or situation awareness.

	<p>HoliDes</p> <p>Holistic Human Factors Design of Adaptive Cooperative Human- Machine Systems</p>	
---	--	---

2.7 RTMaps (INT)

2.7.1 Purpose

RTMaps is a real time, multisensory prototyping software. It supports

- Any kind of sensors and actuators
- Asynchronous data acquisition
- Data time stamping
- Real time data processing and fusion
- Record/playback capabilities
- Latency measurements
- Multithreading

2.7.2 Use-Cases

RTMaps are part of the use-case for the experiment preparation, experiment execution and experiment analysis. The following use-cases are supported:

- Register new sensor - IT expert can define a new measuring device. The system provides basic interface that the IT expert can easily reuse to create a connector for the specific device. When ready the system stores the connector and inserts the device to the list of available devices. The system also registers the device in the experiment database.
- Connect measuring devices - The system provides a list of devices that it can handle. The experimenter selects those relevant to the experiment. The system displays them on a separate sheet and arranges software handlers. The user physically connects the devices and the system verifies status of connection. The system sets common time frame (set common timeline UC) to all connected devices and allows user to register the experiment in a database (register test to database UC).
- Set common timeline - The system reads data from various data sources in any time rate specific for each device. Internally it stamps each data record with a common time so that a specific event can be reliably localized in any of the data sources.
- Record data - The experimenter prepares the experiment and starts its execution through the system. The system stores the start time and starts recording the selected devices. Such device can be the flight simulator or event simulator as well. When finished, the experimenter can store them in the experiment database (store data in database UC).

- Replay data - The analyst selects what data the user is interested in and in what time range. The system provides interface to specify the request and reads requested data from the experiment database (read data from database UC) and displays them for the user. The user can pause, move forwards or backwards and replay the data. There may be several data sources replayed at a time.

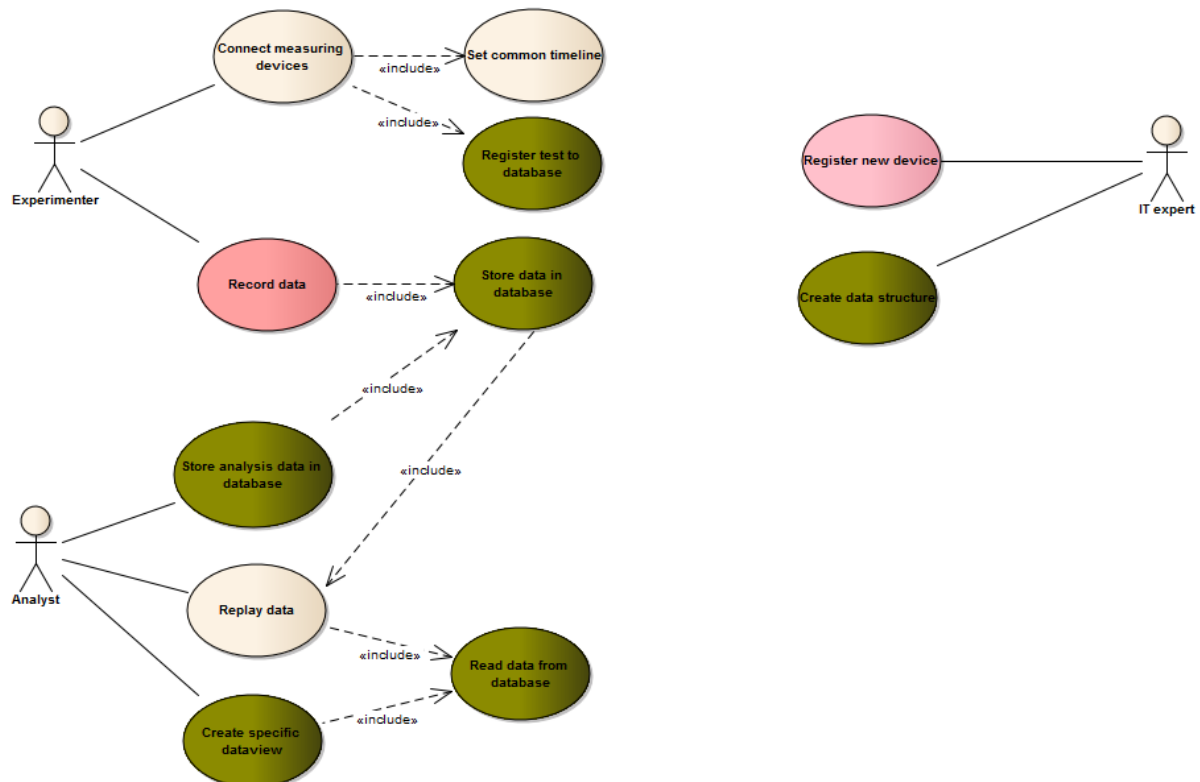


Figure 8: RTMaps use-cases

2.7.3 AdCoS Use Cases

The system fits in a gap in the aeronautics validation workflow: organization and synchronization of the data recording from various data sources, and data fusion/storage/playback, as depicted in Figure 9.

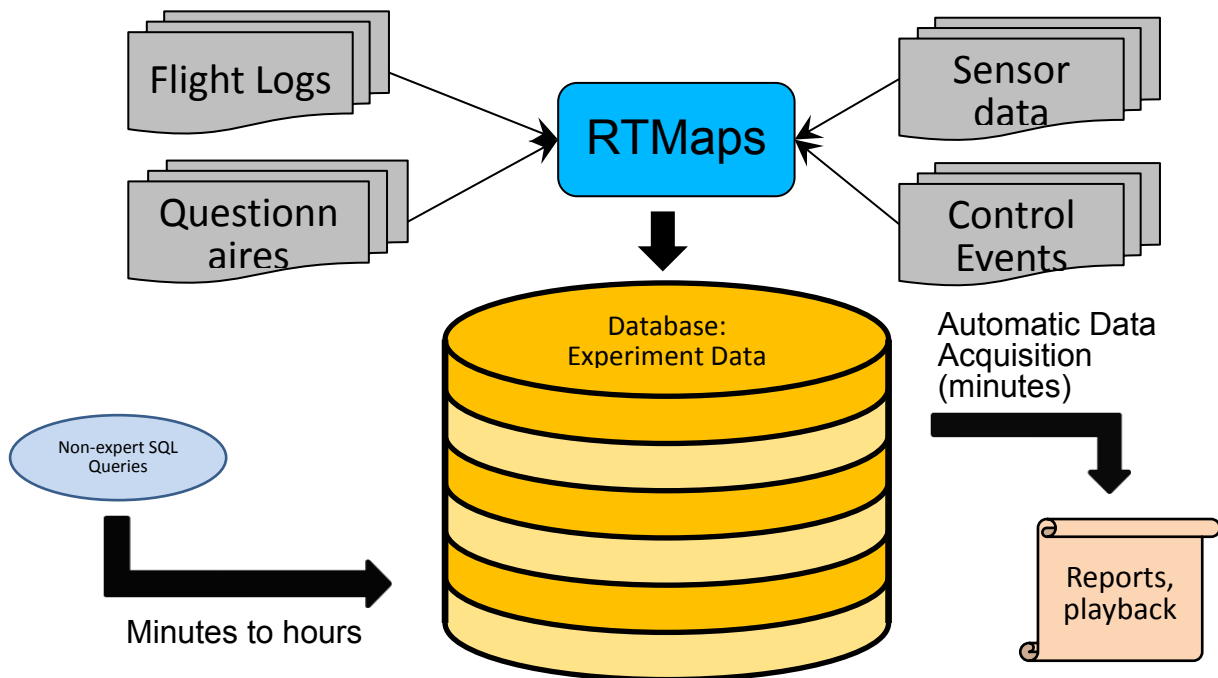


Figure 9: Overview RTMaps Use-Case Aeronautics Domain

2.7.4 Input

Specific sensors need to provide interface specifications to be implemented in the RTMaps. Connection and deployment scheme needs to be prepared to work efficiently.

2.7.5 Output

RTMaps manage data acquisition and storage. As a result there is a data stored filled with synchronized experiment data.

2.8 Usability evaluator (OFF)

2.8.1 Purpose

The Cognitive Analysis of Adaptive Cooperative Systems (AdCoS) depends on complex architectures and simulations and is still driven by proprietary notations. The creation of cognitive models requires in depth cognitive modelling knowledge and is currently only accessible to experts.

New methods and techniques are therefore needed in order to ease analysing the impact of new instruments, new display designs and their

supported adaptations with respect to human factors and to make these techniques available to users without an cognitive modelling background.

Typical design questions that can be answered by performing a cognitive analysis are:

- How does the task execution performance of the operator change with each adaptation?
- How is the workload of the operator affected by different versions of the AdCoS and the adaptation?

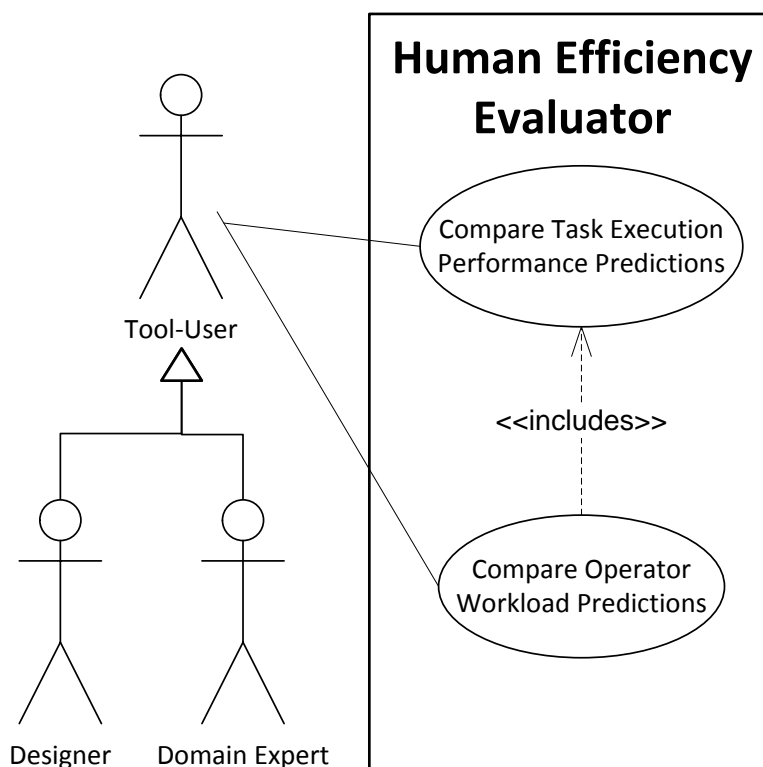


Figure 10: UML Use Case Diagram of the Human Efficiency Evaluator

These questions are typically answered by doing tests with real users performing their task with system prototypes. User testing can result in extensive information that is useful for discovering common errors and usability problems and that provides a feedback before the final system is implemented.

But user testing is expensive in terms of time and money. Test users that represent the targeted audience need to be recruited and paid, which is problematic in safety-critical system domains as specifically and extensively trained operators are needed (i.e. pilots and physician).

Moreover, user testing can only be scaled to a very limited extent: Often, because of costs and time, only a few variants of a design can be tested, especially if these tests require a functional prototype to be implemented.

The HEE is part of a modelling tool chain that consists of several tools:

- Task Editor – to identify interaction tasks between the operator and system.
- SCXML – State Chart Editor for instrument modelling
- Human Efficiency Evaluator – to model the interaction capabilities of the environment, to demonstrate procedures for common tasks and to execute them
- CASCaS – a cognitive architecture for prediction of human behaviour, allowing analysis of HF Metrics

The tool chain supports evaluation in early design phases to predict task performance and workload evaluation of different HMI designs by simulating the human behaviour with a cognitive architecture based on low-fidelity prototypes such as photos, screenshots or sketches as input.

It is possible to analyse and compare HMI designs:

- without the involvement of real users,
- without implementing a system prototype,
- with a huge amount of different variants in a short amount of time, and
- without the need to involve experts in user testing or cognitive analysis.

Offering such an early task and workload prediction gives the opportunity to consider even the most “creative” or “different” designs for an evaluation since efforts for performing such a cognitive analysis are low.

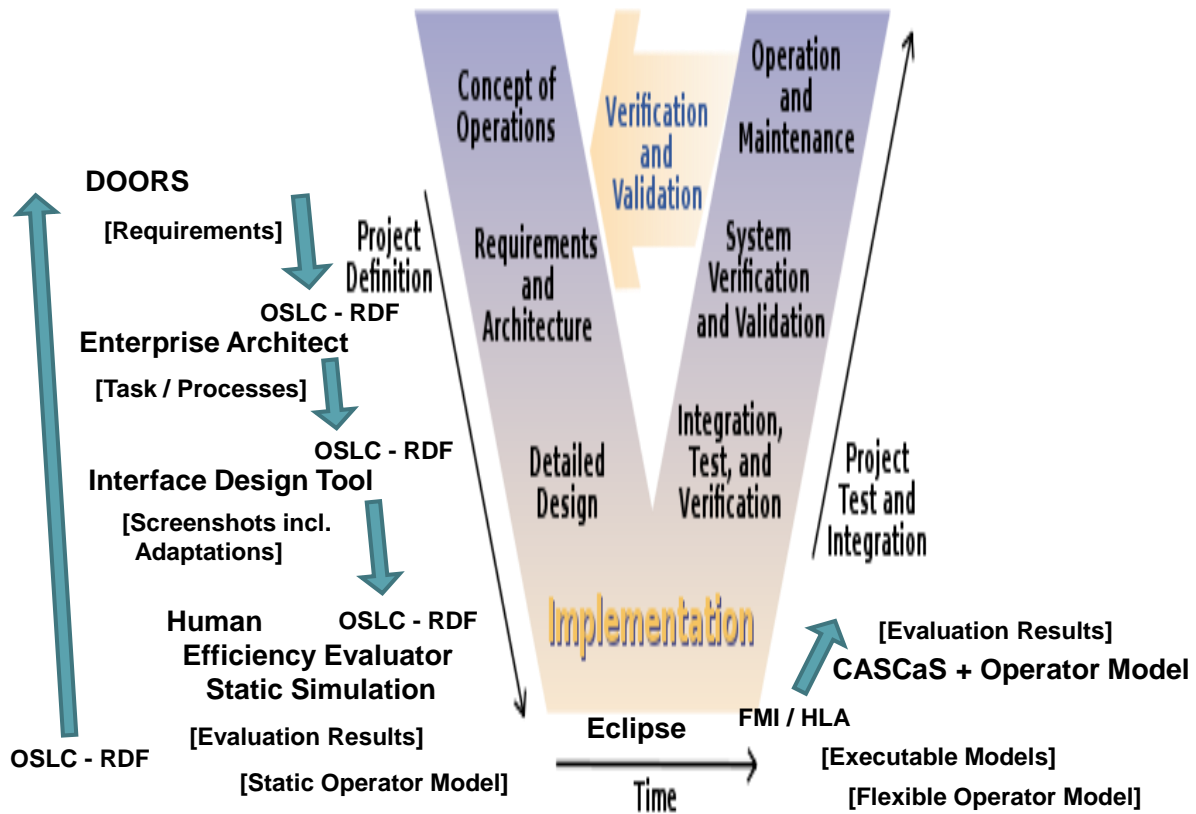




Figure 11: Human Efficiency Evaluator as part of a development process based on the V-Model.

The measurement result quality clearly depends on the quality and amount of the input data. Thus, an absolute measurement (e.g. how much faster is variant X compared to Y) cannot be exactly stated in such an early phase of the design with only limited data available. Instead comparative results is the analysis goal (which variant is faster) and the most convincing variants can then be part of an extended cognitive analysis to get absolute measurements.

Figure 11 depicts the V-model process and illustrates an exemplary use of the HEE. Based on an initial requirements analysis and an optional task design the HEE tool requires sketches or images of the user interface, instrument or display designs created by a graphics editing program as the basic input. The task performance and workload predictions generated by the HEE then could be considered to improve the design to a final

	<p>HoliDes</p> <p>Holistic Human Factors Design of Adaptive Cooperative Human- Machine Systems</p>	
--	--	---

version that then is implemented. After the implementation has been finished the HEE could be used again by feeding it with screenshots of the implementation to evaluate if the design matches the expected task performance.

The HEE is expected to simulate a skilled (trained) user to predict task performance and operator workload. Thus, Human Error predictions have not been considered for the prediction so far.

2.8.2 Use Cases

The HEE tool chain supports modelling and evaluation of early HMI designs. The designer starts with a set of possible designs and wants to compare them with respect to usability, workload etc. upon typical use procedures. The tools can be used instead of expensive activities with operators and the early designs can be in form of photos or even sketches.

Figure 12 depicts the three basic activities that are supported by the tool: (1) Topology definition, (2) Procedure specification, and (3) Operator behaviour simulation. These activities depend on the required input (photos and an optional task or process model) to produce task performance and operator workload predictions as outputs.

Internally, the tool accesses two domain specific repositories: First, an instrument and display widget repository and second, a procedure repository of common operator procedures that cover a specific AdCoS domain.

Like illustrated in Figure 10, two different actors define the target audience of the tool: First, interface and instrument designers and second, domain experts (which are AdCoS users).

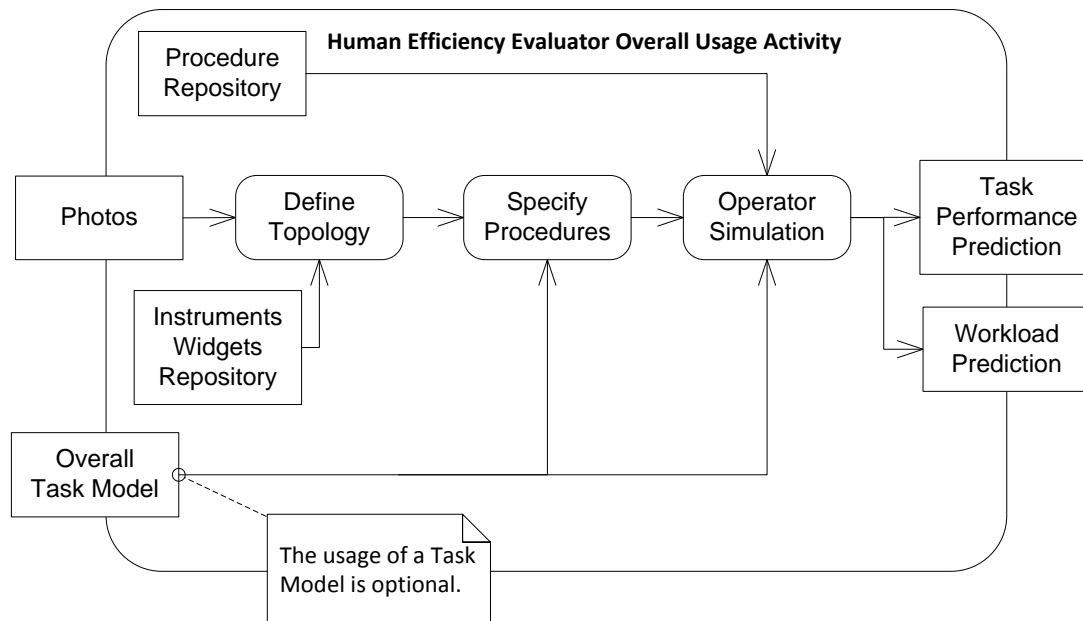


Figure 12: UML Activity Diagram of the Human Efficiency Evaluator overall activities.

The former ones design either specific instruments and interface widgets or the entire AdCoS HMI. They are interested in learning how their designs affect the operator's performance and workload. The latter, the domain experts, fill the procedure repository with their procedures that specify in detail all their interactions with the interface.

Following the activities supported by the tool (Figure 12) the actors first create a topology by identifying the instrument or widgets in the photos and associating them with the ones available in the instrument repository. Thereafter the operator procedures are specified by demonstrating the instrument interactions with the annotated photos.

Finally both the topology and the procedures are fed into the CASCaS cognitive architecture, to simulate the operator performing these procedures, to identify workload peaks and to evaluate the task performance.

The HEE tool can be applied in two use cases: (1) To predict and compare the total task execution times of these alternatives and (2) to predict and compare operator workloads of different design alternatives or adaptation scenarios (figure 1).

2.8.2.1 Compare Task Execution Performance Predictions

The Human Efficiency Evaluator (HEE) tool enables designers and domain experts to predict AdCoS task execution times and compare these performance predictions with alternative designs.

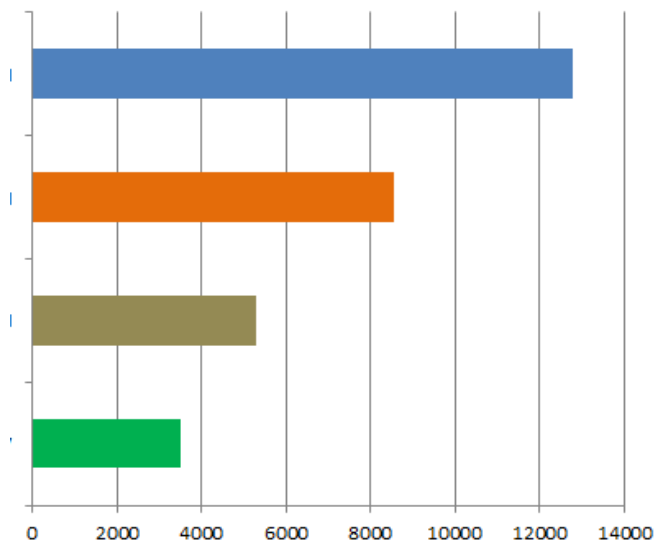


Figure 13: Exemplary performance comparison of different design or adaptation variants.

Figure 13 illustrates an exemplary task performance prediction of the HEE. Different design variants are compared on a millisecond scale.

2.8.2.2 Compare Operator Workload Predictions

After the task performance has been predicted for each design or adaptation variant, the operator workload can be predicted. Figure 14 depicts an exemplary workload prediction for the operator's visual perception. Beneath visual perception, the cognitive and psycho-motoric workload predications of different design or adaptation variants can be compared to identify workload peaks

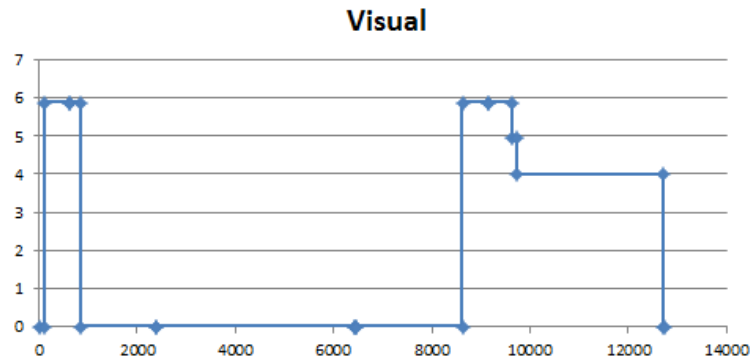


Figure 14: Exemplary workload prediction of the human operator’s visual perception

2.8.3 AdCoS Use-Cases

So far several potential use cases have been initially identified and will be briefly discussed in the following subsections. For all use-cases the concrete applicability of the HEE needs to be further elaborated together with the AdCoS partners. So far, it could be used in the following AdCoS use cases:

WP6, Healthcare	
<i>Activity Management Guidance</i>	WP6_IGS_HEA_REQ06 WP6_IGS_HEA_REQ09
<i>OpenEHR system</i>	WP6_ATO_HEA_REQ08 WP6_ATO_HEA_REQ09 WP6_AWI_HEA_REQ03 WP6_AWI_HEA_REQ14
WP7, Aeronautical	
<i>Adaptive diversion airport advisory</i>	WP7_AER_UC1_HON_v01
WP8, Control rooms	
<i>Operator Overload and Underload</i>	WP8_ADS_CTR_REQ016_v0 WP8_ADS_CTR_REQ019_v0 WP8_ADS_CTR_REQ018_v0 WP8_ADS_CTR_REQ027_v0
	WP8_IRN_CR_REQ008_v0 WP8_IRN_CR_REQ010_v0
WP9, Automotive	
<i>Support agent in LC manoeuvre</i>	WP9_CRF_AUT_UC1



	<p>HoliDes</p> <p>Holistic Human Factors Design of Adaptive Cooperative Human- Machine Systems</p>	
--	--	---

Table 1: The identified use cases (left column) and relevant requirements (right column, needs to be further elaborated with the partners).

2.8.3.1 Healthcare AdCoS

Activity Management Guidance:

“Handheld device for guidance of medical staff in various situations in the hospital, including connection with the OpenEHR, staff availability system, and hospital resources (including radiology facilities, like MRI scanners)” (Integrasys)

The HEE could be applied to compare the task performance of the current design of the guidance system with the context-based guidance adaptations for different user profiles. Interesting specific targets of the performance analysis could be an inspection of the user interface navigation options of the mobile device, to figure out how they affect the overall task performance.

OpenEHR system

“Ability to effectively access and modify patient data in EHR independent from the hospital accessed putting together data coming from different sources such as allergies, medications and images using health standards in the same extract.” (ATOS Healthcare)

Beneath other important aspects, the OpenEHR system should:



- support fast and reliable access to all EHR data
- be easy to use for the doctors allowing automatic generation and extract predictions.

The HEE could be used to evaluate the performance of the main tasks that are supported by the OpenEHR system and that should be performed by doctors. Moreover, the cognitive workload required to use the application could be evaluated to identify those functions that could be harder to perform in stress situations.

2.8.3.2 Control Rooms AdCoS

Operator Overload and Underload (Airbus DS target scenario)

“The Control Room AdCoS detects that the operator is overloaded by one single event or many simultaneous events or one escalating event,

	<p>HoliDes</p> <p>Holistic Human Factors Design of Adaptive Cooperative Human- Machine Systems</p>	
--	--	---

considering e.g. the operator status, or his delayed response, or a system-operator interaction showing less effectiveness resp. efficiency."

The HEE could be used to predict the system-operator's task performance and visual, cognitive and psycho-physiological workload for the most common operator procedures. These results then could be used:

- to identify workload peaks for situations in which several simultaneous events occur.
- to propose either user interface changes that reduce the workload in specific situations or that indicate those parts of the procedure that require load (re)balancing.
- to evaluate to which extent interface adaptations (e.g. re-laying out or help functions) impact the task performance and workload of the operator.



These results are obtained by using the HEE tool to generate a static operator behaviour model (confirm Figure 11 – the left side of the V-model). Such a model does not consider specific user errors, individual behaviours, or any specific educational and cultural backgrounds of the operators, but should offer metrics that can support HMI design decision in an early phase.

Based on the availability and quality of real operator data (e.g. obtained by log file analysis or by real users testing) the static operator model generated by the HEE could then be improved to consider more specific operator characteristics and common user errors (see figure 2 – the right side of the V-model).

Collection of relevant information for the correct interpretation of the malfunctioning (Iren Use case 3)

"The operator receives the call in normal operational conditions, i.e., when the call frequency has an average value. He/she then has to collect from the caller all the necessary information in order to correctly understand the malfunctioning and to solve the problem consequently. An issue that often happens is that the operator sees that some useful information is missing when the call is already ended."

The HEE could be used to measure the task performance of the most common operator procedures that he/she performs while receiving a call. The cognitive and visual perception workload could be specifically

	<p style="text-align: center;">HoliDes</p> <p style="text-align: center;">Holistic Human Factors Design of Adaptive Cooperative Human- Machine Systems</p>	
--	--	---

investigated, to indicate procedure and/or user interface changes in an early design phase.

Based on the availability and quality of real operator data (e.g. obtained by log file analysis or by real users testing) the static operator model generated by the HEE could then be improved to consider more specific operator characteristics and common user errors (see figure 2 – the right side of the V-model).

2.8.3.3 Aeronautics AdCoS

Airport diversion assistant (Use Case 1: Honeywell)

"Electronic Flight Bag (EFB) is an electronic device (aircraft mounted or a tablet) used for information management with ultimate goal to remove paper from the cockpit. Paper charts, reports and various calculators used by pilots to plan their flight are being continuously replaced by software EFB applications."

The HEE could be used to measure the impact of the EFB application for automated planning of a diversion airport regarding pilot workload and task performance in the context of the most common pilot procedures.



The obtained results could help to figure out the relevant information that should/could be visualized on the EFB application, as well as to decide about the most effective adaptations of the interface in the cockpit to minimize the pilots' workload and to maximize their task performance.

2.8.4 Input

Like illustrated in Figure 12 the HEE tool requires: Images of the HMI, which could be photos, screenshots, or sketches as inputs. Optionally, for comprehensive AdCoS that cover a lot of procedures, a task or process model can be used as an input to ease the procedure generation inside the tool.

Requirements for input format:

Images should be accessible from the file system in a common used file format (png, gif, jpg).

	<p>HoliDes</p> <p>Holistic Human Factors Design of Adaptive Cooperative Human- Machine Systems</p>	
---	--	---

The task model should conform to the W3C Concur Task Trees (CTT) format (<http://www.w3.org/2012/02/ctt/>).

2.8.5 Output

Results from the tool will be used in evaluation of designs and selection of the best candidate(s). Therefore, there should be a way to load the data in common tools for data analysis.

The raw evaluation data results for task performance and workload prediction can be exported in comma-separated value format (CSV) to be used by external data visualization software. Such visualization software should display several data sets at the same time to ease the comparison of different candidates.

2.9 Tools for profiling the concurrency bugs (BUT)

2.9.1 Purpose

A set of tools that work on source codes or binary executables in order to

- Increase a probability of thread race conditions
 - by noise insertion into scheduler and/or
 - stochastic simulation of values of parameters used by the application
- Detect thread races and deadlocks
- Detect atomicity violations
- Identify the bug



2.9.2 Use Cases

Each development cycle contains a phase of testing. The tools are applied as part of the tests and they target in the testing parallel processing.

2.9.3 AdCoS Use-Cases

The AdCoS use-cases are depicted in Figure 15:

- Select tool - The user navigates through graphical interface to a list of available tools. Each tool has a description that contains its purpose, mode of use and results it provides. The user goes through the list and picks up a tool for his goal. When a tool is selected it appears in a tool area so that the user is aware of his selection.

	<p>HoliDes</p> <p>Holistic Human Factors Design of Adaptive Cooperative Human- Machine Systems</p>	
---	--	---

- Execute test - The user executes the test to get information about the tested object. As each test may use a set of parameters that affect it, the user may decide to configure the test (*configure test UC*). The decision however is up to the user and if configuration is not requested, default values will be used. The test requires an object and the system asks the user automatically to load the object (*load assembly UC*). The system provides information about the object, which is given by the selected tool, for example a tool for inspection of the source code may look specifically for a project file.
- View results - When the execution is finish, the user can inspect the collected output. The system provides a test summary and test details. System presents details as a log with coloured lines using orange for warnings and red for errors. The user decides whether the results should be stored (*store results UC*).
- Store results - When user decides to store results the system formats the summary into the database and creates log file from the details The log file is attached to the summary tables together with the parameterization of the test. The system connects to the versioning system to add related revision (*load revisions from repository UC*). In case system cannot do it automatically the user is asked to specify the revision.
- Inspect test history - The user inspects history of tests and can compare current results to those obtained in past (*load results UC*). The system provides access to the database of results and when the user selects a record, the system displays if side by side to the current (or other selected) results (*view results UC*). The user can invoke a connection to versioning system to see that are the difference between the two result sets (*load revisions from repository UC*).
- Load revisions - The system take versioning information from two records and connect to the versioning system. The system asks for differences in revisions and the versioning system computes and displays them. If there is no versioning information, the user is informed.

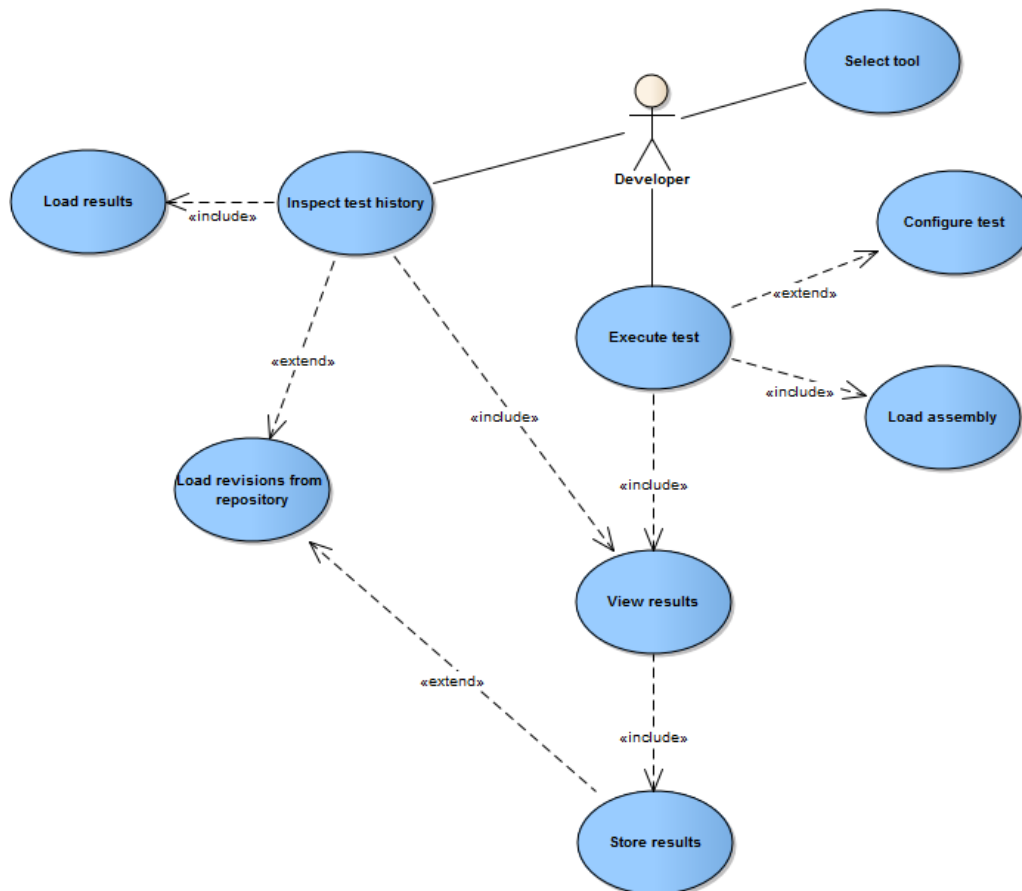


Figure 15: Use-Cases Overview

2.9.4 Input

The tools require a set of configuration parameters and a binary executable to be tested.

2.9.5 Output



Tools produce test logs with details of detected problems.

2.10 Certification tool (TEC)

2.10.1 Purpose

Certification tool assembles regulation documents and procedures that are required to certify a product. The tool guides a user step by step, and provides related restrictions and requirements. Expected features are:

- List regulations for a given type of a product.

	<p>HoliDes</p> <p>Holistic Human Factors Design of Adaptive Cooperative Human- Machine Systems</p>	
--	--	---

- Full-text search of keywords in regulation documents.
- Web based step-by step guidance through a process.
- Artefact validation.

2.10.2 Use Cases

Tool guides a user through the appropriate certification procedure helping to find relevant regulations, validate artefacts and follow the right procedure.

2.10.3 AdCoS Use-Cases

The AdCoS use-cases are depicted in Figure 16:

- Load certification procedure – The system offers a set of certification procedures in various domains. The user selects procedure for his task. He may start a new procedure or open one previously stored (*continue open procedure UC*). When a procedure is selected, the system displays available data and starts the procedure from the last completed step.
- Execute certification procedure – The user works with the dialog of current procedure step. He may navigate to previous or future steps. The system provides user with currently inserted data, with guidelines, with links to related regulations and with all artifacts that have already been attached to the procedure. User can attach artifacts (pictures, email, documents) to any item in the dialog. When attached, the system validates the artifact if it has validation rules for the type of the artifact (*validate artifacts UC*). At any time the user can store inserted data and interrupt the work.
- Report - The user wants to create actual state report. He selects a procedure and the system formats information into a report structure. If the procedure is not completed, the system creates action items from the incomplete steps (*create action items UC*).
- Create action items - The system finds the last completed step. For each of the following steps, it lists what is required for a given procedure. Each requirement should be commented so that the user understands what he needs to do and what artefacts need to be worked out.

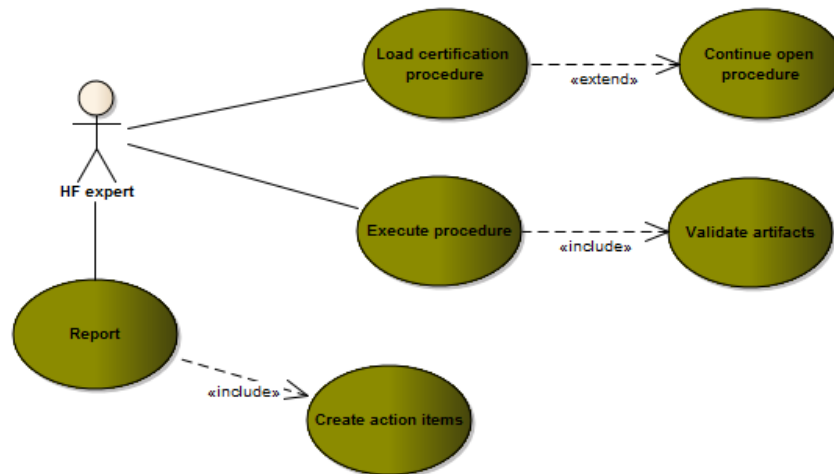


Figure 16: AdCoS Use-Cases Overview Certification Tool

2.10.4 Input

The tool asks for mandatory artefacts (documents) during the process of usage.

2.10.5 Output

The tool creates a report that can be used during the negotiations with the certification agencies.

2.11 Atos monitoring system

2.11.1 Purpose

The **Monitoring system** will be developed to give a vision in real time to end users, administrator users and different actors about specific use-cases (which are given in the document "D9.1- Requirements Definition for the HF-RTP, Methodology and Techniques and Tools from an Automotive Perspective").

This system will be connected to the HoliDes communication module to send/receive data information from/to devices, actuators and servers, providing an easy interface to monitoring use cases proposed.

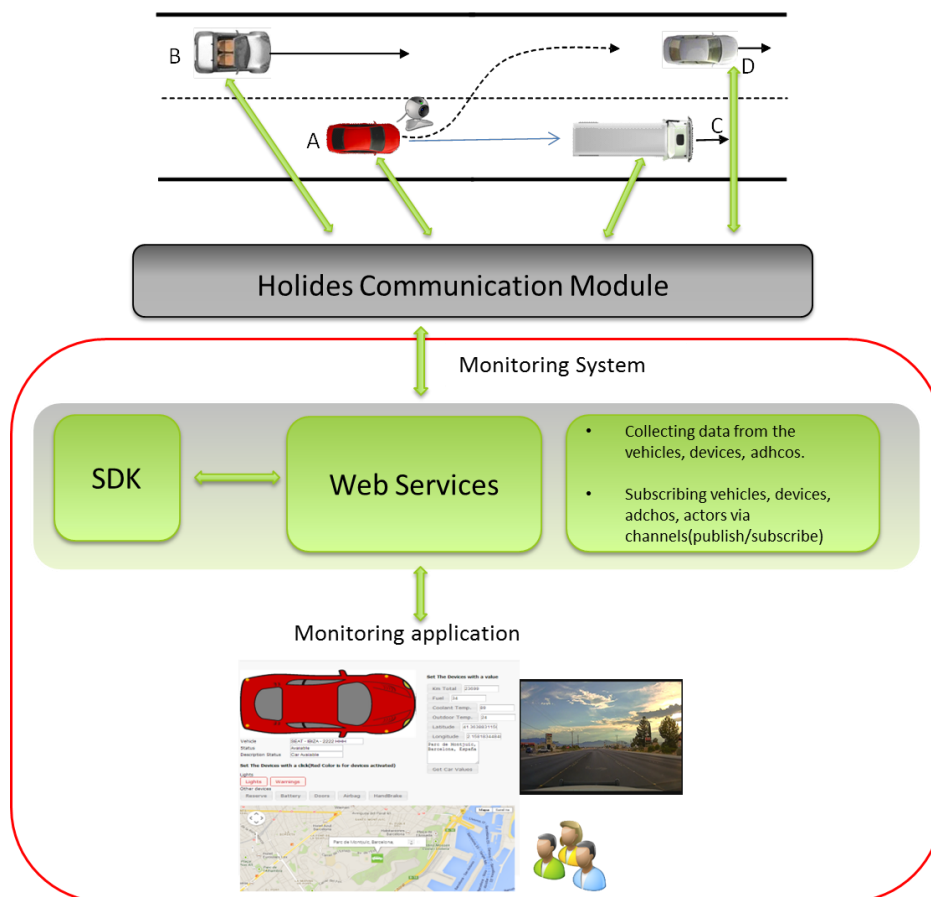




Figure 17: Monitoring System Overview

2.11.2 Use cases

The circles in Figure 18 are the use cases in the monitoring system grouped in common functionalities from AdCoS use cases.

Use Cases in monitoring systems:

Data AdCoS: Data provided by sensors vehicles, gateways, and information data from others sources (emergency services, police, etc.) via HoliDes communicator module. The driving manoeuvre (driver actor) in an overtaking use-case or cross-traffic use-case will influence the information sent to the monitoring system.

	<p>HoliDes</p> <p>Holistic Human Factors Design of Adaptive Cooperative Human- Machine Systems</p>	
---	---	---

Simulate Scenario: Use-case to manage data AdCoS. Receiving the information sent from HoliDes communication module, the information can be treated by Monitoring System User to know 'what it's happening' and 'what it has happened' during cross-traffic or overtaking situation.

Data Channel monitoring: The user of the Monitoring system can link different data entities provided by the HoliDes communication Module. In an overtaking scenario it can be the drivers involved in an overtaking, sensors status, and the different vehicles. It will show data at real time grouped by channels. Each channel contains some entities involved (cars, drivers, sensors, etc.).

Data entity monitoring: Each HoliDes entity can be evaluated individually (Vehicles Webcams, devices, and sensors) by the user of the Monitoring System.

Database Storage: All data information received from HoliDes communication module will be saved.

Report Data: Simulation scenarios created by Monitoring System User can be saved and treated with different tools to make an easy interface to analyse different situations (e.g.: an accident happened in a overtaking).

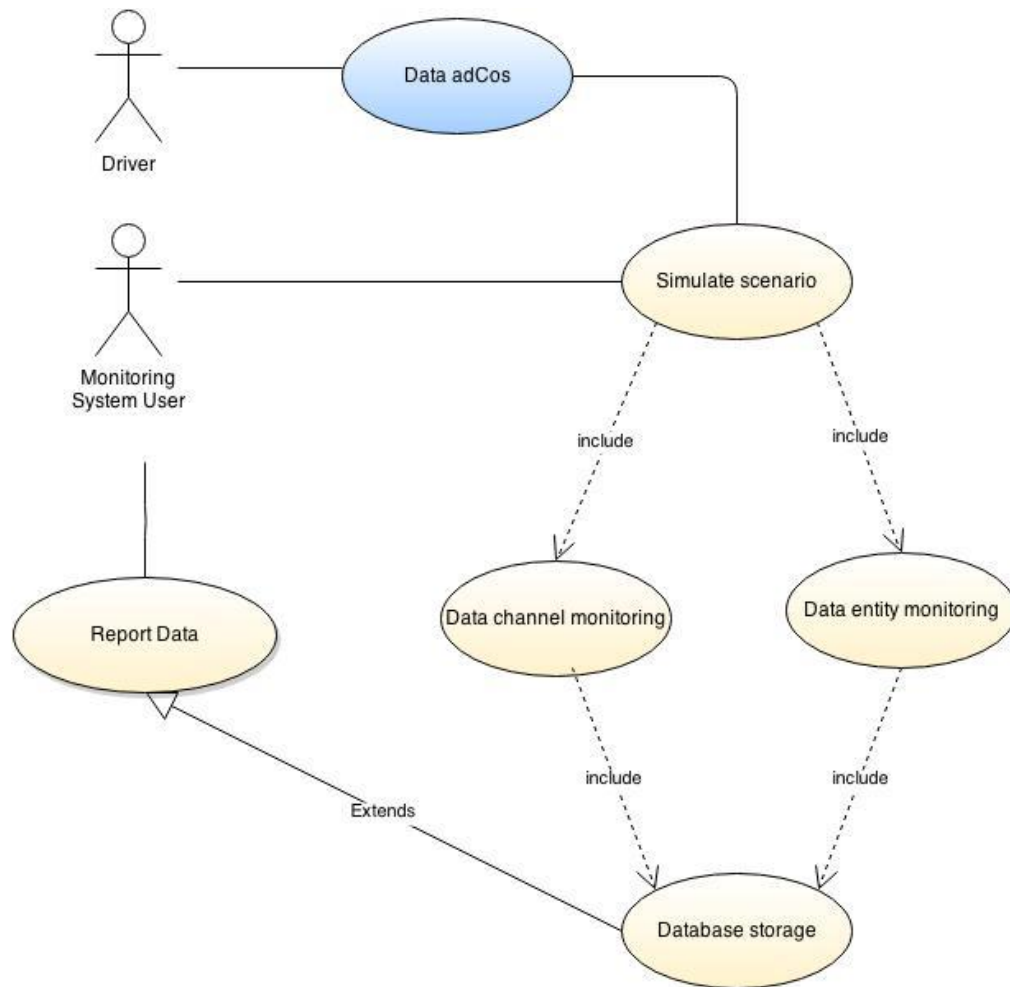


Figure 18: Monitoring System Use-Case Overview

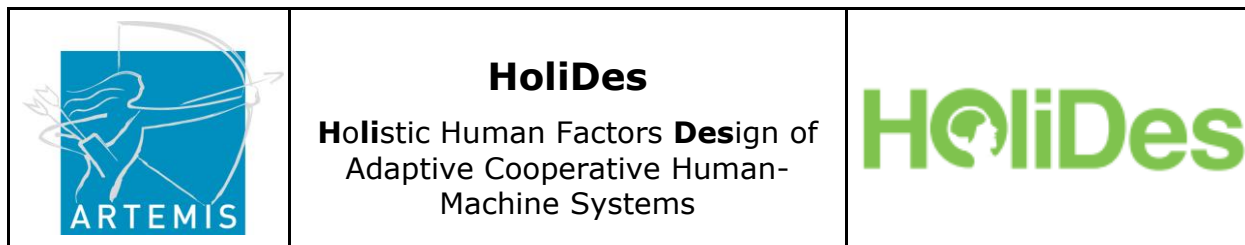
2.11.3 AdCoS use cases

Monitoring an overtaking use-case: showing data from “overtaking use case” devices, use case actors, vehicles and overtaking situations.

Monitoring a cross-traffic use-case: showing data from “cross-traffic use case” devices, use case actors and cross-traffic situations.

2.11.4 Input & Output

Figure 19 shows the connection between monitoring system and Holidess communication module. Each layer has an input/output connection between them or external modules.



Input data provided by HoliDes Communication Module:

- Car devices status
- Drivers information
- Vehicles involved in the AdCoS use cases (overtaking, cross-traffic)
- Other information emergency services, weather forecast,...

Figure 19 shows the communication between different Monitoring System Layers to transmit this information.

The Monitoring System can link information and group entities (devices, sensors, car information, webcam streaming, driver information) to support evaluation in an easy interface. Data output can be filtered to show specific results, like: graphics diagrams, data grids, video screens etc, to make a useful and helpful tool for User Monitoring.

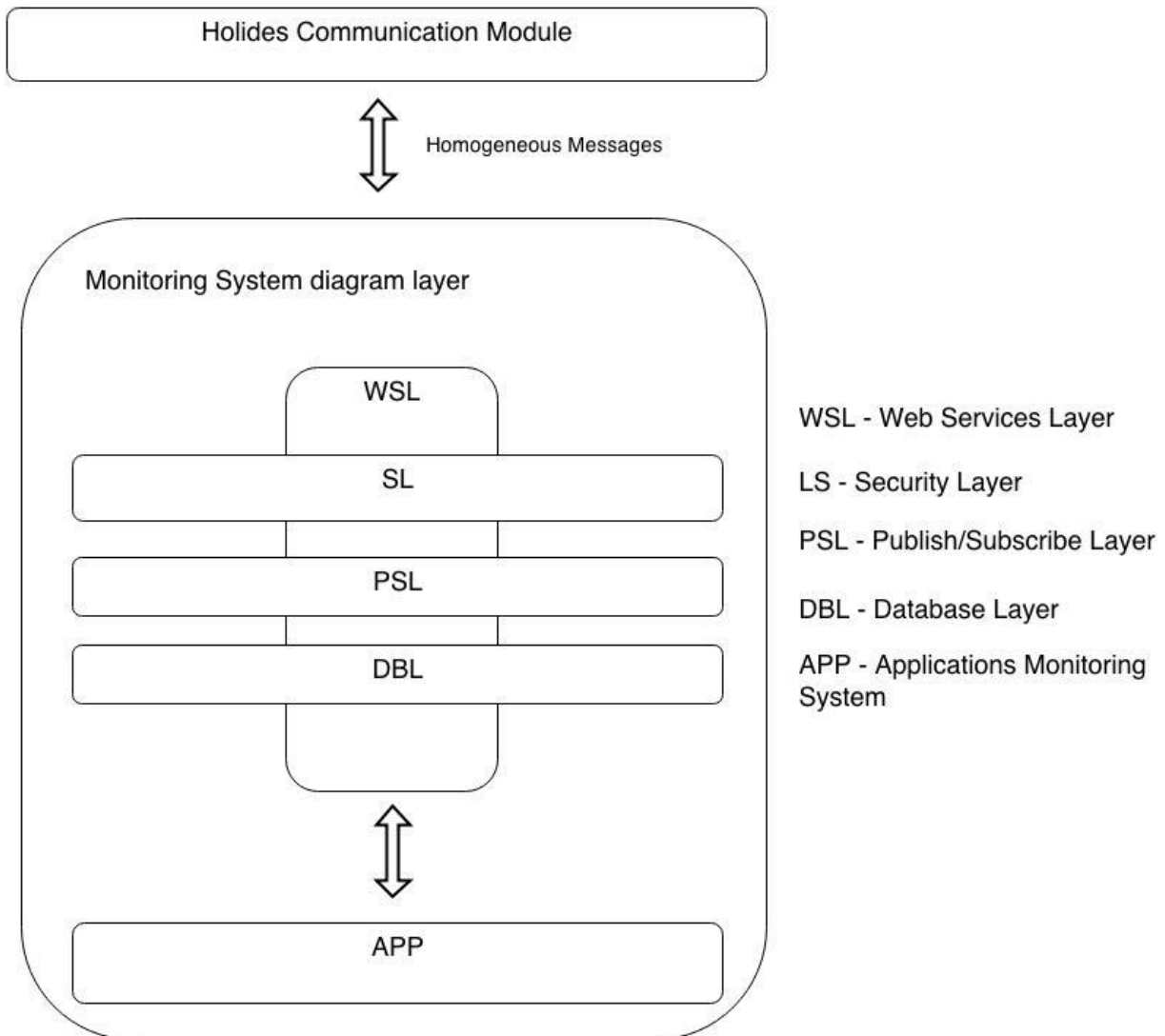
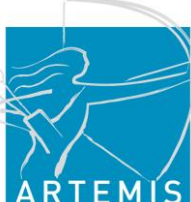



Figure 19: Monitoring System Architecture

Definitions of acronyms in the figure:

- WSL – Web Services Layer
- SL – Security Layer
- PSL – Publish / subscribe layer
- DBL – Database Layer
- APP – Applications monitoring system

WSL – the main goal of this layer is to communicate Monitoring System with Holidés communication module, also has communication with the rest of layers SL, PSL, DBL and APP to give services to them. All the connections are I/O (e.g. write and read in a database, i.e. communication with DBL).

	<p>HoliDes</p> <p>Holistic Human Factors Design of Adaptive Cooperative Human- Machine Systems</p>	
---	--	---

SL – Security Layer – communicates with WSL. All the communications are I/O.



PSL – Publish and subscribe layer will connect with a server, this one manage the queues and the messages synchronism. WSL and SL have communication I/O with PSL.

DBL – Database layer has connections I/O with WSL.

APP – The main goal is to show and manage all data and for it, its Applications are connected to WSL.

3 References

- Bellet T., Bailly B., Mayenobe P., Georgeon O. (2007). Cognitive modelling and computational simulation of drivers mental activities. In P. C. Cacciabue & C. Re (Eds.), *Critical Issues in Advanced Automotive Systems and Human-Centred Design*, Milan, Springer, pp.317-345.
- Bellet T., Mayenobe P., Bornard J.C., Paris J.C., Gruyer D., Claverie B. (2011). Human driver modelling and simulation into a virtual road environment. In Cacciabue, P. C.; Hjalmdahl, M.; Lütke A.; Riccioli, C. (Eds.). *Human Modelling in Assisted Transportation: Models, Tools and Risk Methods*, Milan, Springer, pp. 251-262.
- Bellet, T., Bailly-Asuni, B., Mayenobe, P., Banet, A. (2009). A theoretical and methodological framework for studying and modelling drivers' mental representations, *Safety Science*, 47, pp. 1205-1221.
- Bellet, T., Mayenobe, P., Bornard, J.C., Gruyer, D., Claverie, B. (2012). A computational model of the car driver interfaced with a simulation platform for future Virtual Human Centred Design applications: COSMO-SIVIC, *Engineering Applications of Artificial Intelligence*, 25, pp. 1488-1504.
- Clarke, Kroening, Sharygina, & Yorav, SATABS: SAT-based predicate abstraction for ANSI-C, 2005
- Cousot & Cousot, Abstract Interpretation Frameworks, 1992
- Edelstein, Farchi, Nir, Ratsaby, & Ur, Multithreaded Java program test generation, 2002

	<p>HoliDes</p> <p>Holistic Human Factors Design of Adaptive Cooperative Human- Machine Systems</p>	
---	---	---

- Fiedor & Vojnar, ANaConDA: A Framework for Analysing Multi-threaded C/C++ Programs on the Binary Level, 2012
- González--Calleros, J. M., Osterloh, J. P., Feil, R., & Lüdtkke, A. (2013). Automated UI evaluation based on a cognitive architecture and UsiXML. Science of Computer Programming Journal, In Press. doi:<http://dx.doi.org/10.1016/j.scico.2013.04.004>
- Henzinger, Jhala, Majumdar, & Sutre, Software verification with BLAST, 2003
- Kam & Ullman, Global data flow analysis and iterative algorithms, 1976
- Nielson & Nielson, Principles of Program Analysis, 1999
- S. Puch, M. Fränzle, J.-P. Osterloh, and C. Läsche, "Rapid virtual-human-in-the-loop simulation with the high level architecture", in Proceedings of Summer Computer Simulation Conference 2012 (SCSC 2012), ser. Simulation Series Vol, A. Bruzzone, Ed., vol. 44, no. 10, Society for Modelling & Simulation International (SCS). Genoa, Italy: Curran Associates, Inc., 07 2012, pp. 44–50.
- Visser, Havelund, Brat, Park, & Lerda, Model checking programs, 2003