HoliDes

Holistic Human Factors **Des**ign of Adaptive Cooperative Human-Machine Systems

D2.2- Integration Plan for Modelling Techniques and Tools into the HF-RTP and Methodology

Project Number:	332933
Classification:	Public
Work Package(s):	WP2
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: D2.2_Integration_Plan_for_MTT_into_HF-RTP **Annex I:** D2.2-D5.2_ Integration_Plan_Common_Part.docx - Public
Compiled by:	Robert Hofsink, PHI
Authors:	Jan-Patrick Osterloh, OFF Sebastian Feuerstack, OFF Lars Weber, OFF Zdenek Moravek, HON Cristobal Curio, TWT Thierry Bellet, IFS
Reviewers:	Susanna Donatelli, UTO David Käthner, DLR
Technical Approval:	Jens Gärtner, EAD-DE-IW
Issue Authorisation:	Sebastian Feuerstack, OFF

DISTRIBUTION LIST		
Copy type[1]	Company and Location	Recipient
T	HoliDes Consortium	all HoliDes Partners
E	ARTEMIS	Patrick Vandenberghe

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

RECORD OF REVISION		
Date (DD.MM.JJ)	Status Description	Author
14.05.2014	Initial Version	JPO
02.06.2014	Integration of Content	JPO
03.06.2014	Added COSMODRIVE/COSMO-SIVIC	TB

Table of Contents

1 Introduction

This document gives details on the integration of MTTs into the HF-RTP for WP2. Please refer to the common Integration Plan document for further details on the HF-RTP and possible integration techniques.

The objective of WP2 is to develop modelling techniques and tools for all components of Adaptive Cooperative Human-Machine Systems (AdCoS) in order to formalize the capabilities and strategies for dynamic adaptation of the overall system at a global and local level.
It gets as input the requirements and feedback from the demonstrator applications in WP6-9 and the requirements from the HF-RTP definition in WP1. Furthermore, data for model development and model validation is received from WP 6-9, too.
All modelling techniques will be targeted to formalize adaptation strategies in order to make them explicit and thus analysable (with techniques from WP3, 4 and 5). Adaptation strategies are considered in HoliDes at a global and local level and thus have to take into account all elements of AdCoS and their interaction: AdCoS tasks and resources, cooperation between agents, behaviour and cognitive processes of human operators, human-machine interfaces and operator training.
With regard to their application in Embedded System development and qualification processes the model can fulfil two purposes: (1) they can be used as components of AdCoS to realize adaptive features (e. g. models to infer the state of a human operator based on measurements of her/his actions and mental condition) and (2) they can be used to formalize and evaluate the AdCoS at early design stages before a prototype is available. To support AdCoS designers and/or human factor experts in using the notations, modelling editor tools will be built. In order to integrate all model types and to prepare their interoperable usage across the AdCoS engineering life cycle a common conceptual modelling framework will be defined.
In WP3 the models will be used as a basis for building adaptation mechanisms; in WP4 analysis techniques will be developed to enable their execution and computation; in WP5 they will be used to define empirical model development and model validation techniques.

The following sections describe the initial set of MTTs developed within WP2, which are provided to the other work packages.

2 MTTs for Integration

2.1 PED (OFF)

2.1.1 Purpose

The Procedure Editor PED enables rapid prototyping of cognitive task models. Based on a Hierarchical Task Analysis, normative behavioural models may be defined with a simple graphical notation. It is intended to be a tool for experienced modellers, as well as for domain experts with little experience in cognitive task modelling. PED has been used already to model tasks from the automotive, aeronautic, and maritime domains.

PED consists of a graphical editor and a simulation component, as well as a validation component for model integrity.

The graphical task definition language provides a high-level abstraction away from the concrete procedural and declarative structures needed otherwise for a particular production system. Models may be simulated directly in PED (Figure 1), which shortens the evaluation cycle for the inexperienced modeller. Furthermore, models are validated constantly during the editing process to ensure formal constraints. Thus, the modeller is assisted actively by the application.

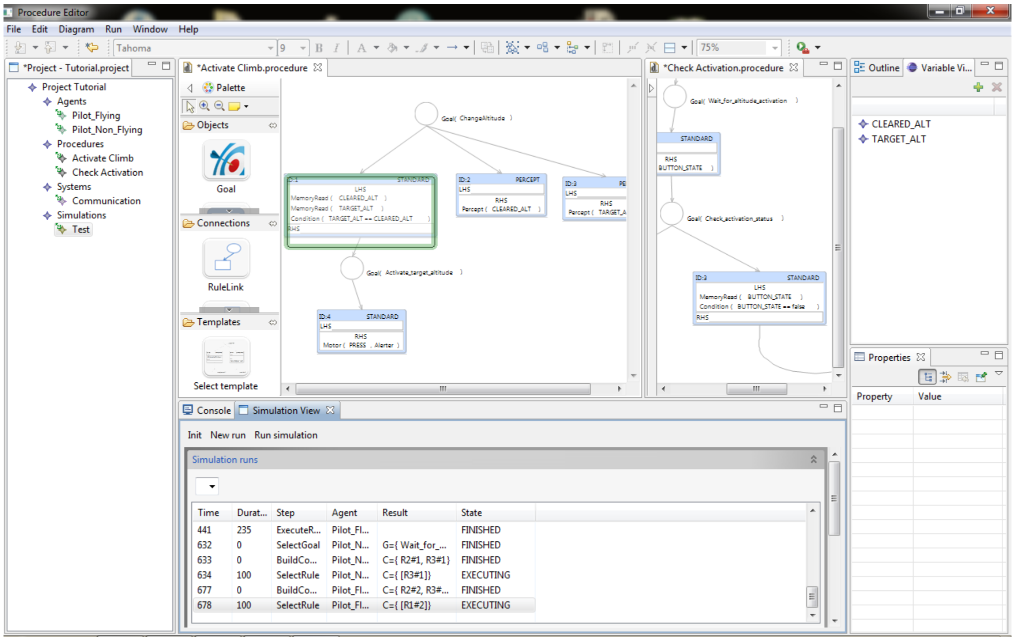


Figure 1: PED editor with simulation of two procedures

Additionally, models may be transformed into a format that is readably by CASCaS, a fully functional cognitive architecture. These models may serve the modeller as proofs of concept or starting points for more refined, manually enhanced models. Models may be reused as templates or be incorporated into larger models. The simulation component is also used for Workload estimation for specific tasks.

Hierarchical task models from PED may be transformed into Finite State Machines for formal model-checking. While this feature is still evolving, it will eventually allow PED models to be used for formal safety analyses.

PED has been implemented with up-to-date model-driven software development (MDSD) technologies. Conceived as an Eclipse RCP application, the high-level language is implemented with an Ecore metamodel and transformed with Model-2-Model transformation language, and the graphical editor has been implemented using the Graphical Modelling Framework (GMF). The usage of these technologies ensures future scalability and extensibility.

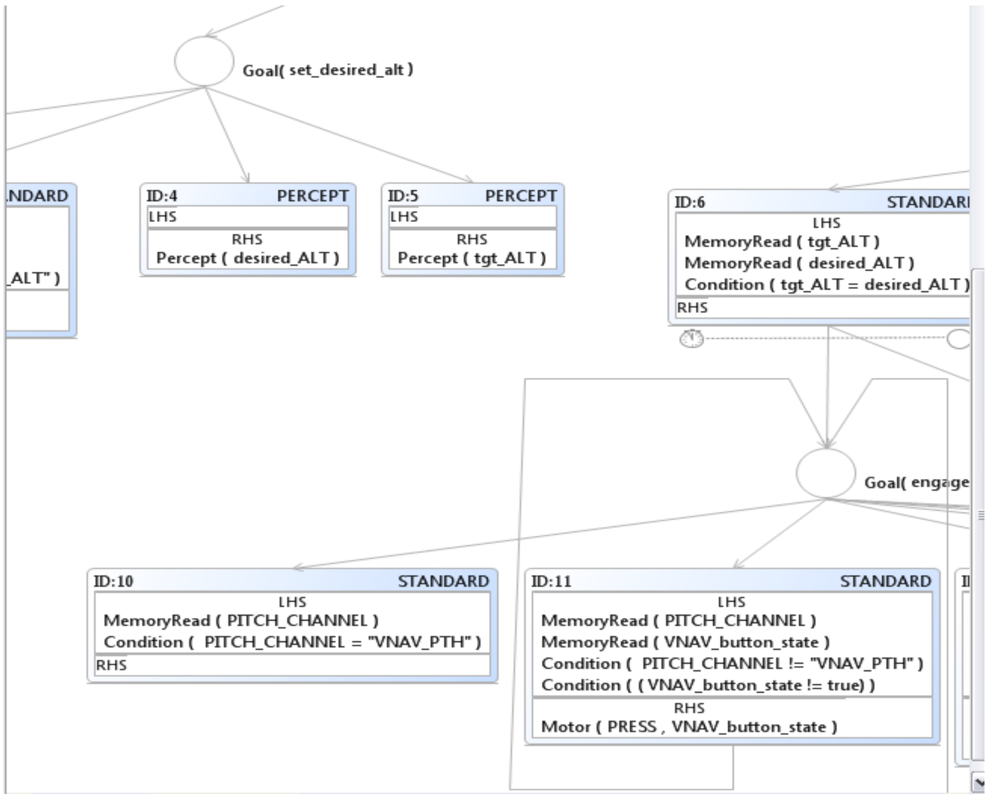


Figure 2: PED Model (extract)

2.1.2 Use Cases

Currently, PED allows fine granular modelling of hierarchical tasks, i.e. for pilot tasks in aeronautics or maritime domain. This formal modelling allows Designers to specify in detail how to operate a certain system, which can serve not only as a communication mean with customers and with other team members, but also to foster HMI design (presenting only the relevant information for the current task). The simulation component of PED allows testing of the procedure and calculation of workload. The workload calculation can be used, for example, to compare different task models and their associated design. Furthermore the model can be used as input for CASCaS, which allows a more detailed simulation of the procedure.

Formal task models have multiple benefits, e.g.:

- Communication means with customer or team colleagues
- Executable model for communication with supplier

- Support for task-focused HMI design
- Model based Validation & Verification
- Calculation workload for a task model allows objective comparison of models

Currently OFF is working on the implementation of an UML profile, allowing modelling the same information in an UML tool, like MagicDraw, thus having a broader integration with the usual design process in industry.

2.1.3 AdCoS Use-Cases

PED will be used in MAV domain for the Use Case 2 "Adaptive Flight Crew Simulator Transition Training" for modelling the training content.

2.1.4 Input

All domain information obtained in advance, i.e. through a Hierarchical Task Analysis, Cognitive Work Analysis, either formalised or as text, which allows the modeller to model the tasks. A list of used systems and it's interactions (in form of variables) is necessary for accessing the systems in the procedure.

2.1.5 Output

A fine granular formal model of tasks, which an operator has to perform to reach a certain goal, i.e. flying an aircraft or driving a car. More details on PED can be found in Lenk et al (2012).

2.2 CASCaS (OFF)

The Cognitive Architecture for Safety Critical Task Simulation (CASCaS) is a framework for modelling and simulation of human behaviour. Its purpose is to model and simulate human machine interaction in safety-critical domains like aerospace or automotive, but in general it is not limited to those specific domains.

2.2.1 Purpose

Figure 3 shows the current version of the architecture with all its components. Basically, the architecture consists of 5 components: a *Goal Module* which stores the intentions of the model (what it wants to do next). The *Central Processing* is subdivided into three different layers: the cognitive layer which can be used to model problem solving, the associative layer executes learned action plans and the autonomous layer simulates highly learned behaviour. The memory component is subdivided into a procedural (action plans) and a declarative knowledge (facts) part. The *Perception* component contains models about physiological characteristics of the visual, acoustic and haptic sensory organs, for example models about peripheral and foveal vision. To interact with the external environment the *Motor* component of CASCaS contains models for arm, hand and finger movements. It also comprises a calculation for combined eye / head movements that are needed to move the visual perception to a specific location. In general the model starts observing its environment via the perception and receives input which is stored in the memory component. Depending on its current intention and on the perceived information from the environment, it selects action plans and tries to achieve its current goal. It may generate new goals and further actions, which can be triggered by events perceived from the environment or the model may itself create new goals, based on its own decision making process, to initiate a certain behaviour.

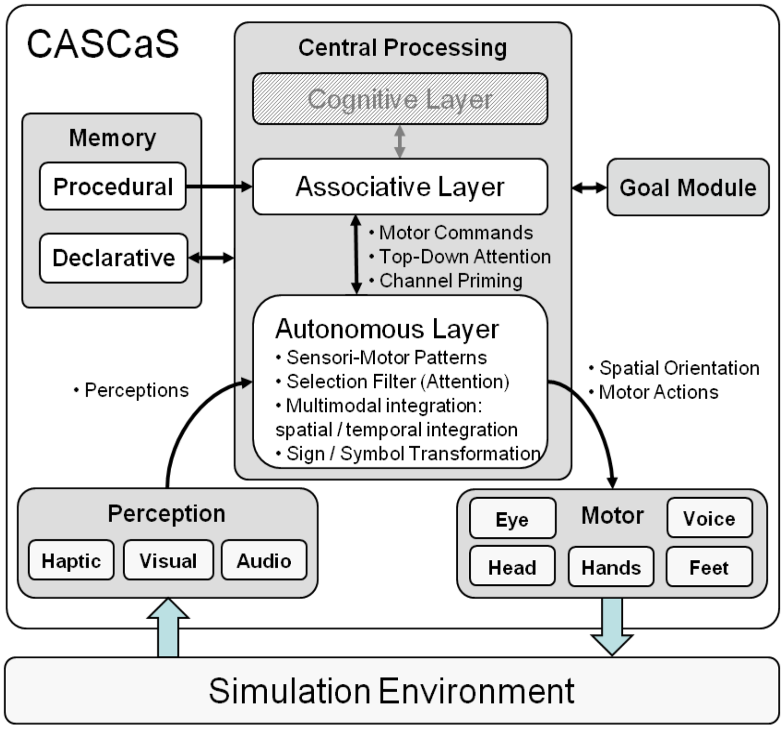


Figure 3: structure of the cognitive architecture CASCaS with all internal components and the major data flows.

2.2.2 Input

The description above gives a very rough overview of how the architecture simulates human behaviour. Two specific input files are required for a simulation: a procedure and a variable specification file. Both files are loaded into the architecture at start-up. The procedure file specifies task and domain

specific knowledge about how the model should interact with its environment, for example: “If display X shows value Y I have to press button Z”. The architecture itself is domain and task independent: only by loading an appropriate procedure file it becomes, for example, a driver model or a pilot model. For both of those domains (automotive / aeronautic) OFFIS has developed models which can interact in specific scenarios and driving / flight simulator environments, e.g. a driver model which can drive on a two lane German Autobahn, performing free-flow, car-following and lane change manoeuvres, or a pilot model which can simulate the cockpit interaction necessary by a pilot for take-off or approach scenarios. The second file (the variable specification file) is used to define the data which is exchanged between one or more simulators and CASCaS, as well as the topology of the environment (i.e. where certain information is located in space).
Integration of CASCaS into simulation environments can be done either by point to point connections using UDP or TCP/IP sockets or by integrating all components into an HLA simulation platform. For the latter one OFFIS uses the open source CERTI High-Level Architecture (HLA) Implementation and has implemented several different HLA federates. CoSimECS, supports setting up the simulation by allowing graphical configuration of the HLA simulation, see also D4.2 for details on CoSimECS.

2.2.3 Use Cases

In the aeronautics domain CASCaS was already used to simulate procedural tasks for specific flight manoeuvres. The task execution times as well as the gaze behaviour are outputs which can be used for statistical analysis purposes. Alternative task procedure designs can be simulated and compared against each other. Designing a new cockpit system always requires the specification of operating procedures. With such a simulation, engineers can check if a new procedure for a system covers the necessary functions in certain test scenarios. At a very first pure software simulation stage it allows first feedback about possible interaction problems, e.g. if necessary information is cluttered, the task execution time will automatically raise.

In the automotive domain OFFIS has developed a driver model which is already able to basically deal with the intended scenario of WP9. The model can simulate free-flow, car-following and lane changes right and left on a two-lane German Autobahn with medium traffic density. The model simulates gaze behaviour including a mirror view which automatically covers blind spot problems. The model heavily relies on peripheral and foveal vision which is necessary to interact with the highly dynamic traffic environment. Intention based top-down behaviour (model wants to do a lane change) as well as reactive bottom-up behaviour (model detects that a car has set its indicator) are important parts of the model. The existing driver model can be used in HoliDes and it can be extended by additional operational procedures to interact with an Adaptive Driver Assistant System (ADAS). The output is similar as for the aeronautic domain. Task interaction time and gaze behaviour can be analysed. Additionally, the impact of secondary tasks on, for example, distance keeping and lane changing could be analysed.

2.2.4 AdCoS Use-Cases

In the domain of driver modelling OFFIS and TAKATA have agreed on a use case where the model should be used to simulate the interaction between the driver and the HMI developed by TAKATA. The targeted Use-cases are WP9 TAK1-5.
In the aeronautics use case 1 “Airport diversion assistant”, CASCaS will be used together with the Human Efficiency Evaluator and CoSimECS to evaluate the system, and will be part of the AdCoS as rule-based evaluation of pilot state.

2.2.5 Output

For the previously described automotive use-case, it is expected that the output of the simulation directly addresses certain requirements, e.g. the driver model could be used to test requirements concerning the blind spot system, for example WP9_[TAK]_AUT: REQ01 / 03 / 05-10. Which requirements can be directly addressed is still to be discussed, but, as already mentioned, the driver model has specific capabilities to simulate mirror views in a virtual 3D simulation environment. Both partners use the same driving simulator software, thus the integration of the driver model is expected not to raise particular problems.

2.3 Human Efficiency Evaluator (OFF)

2.3.1 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?

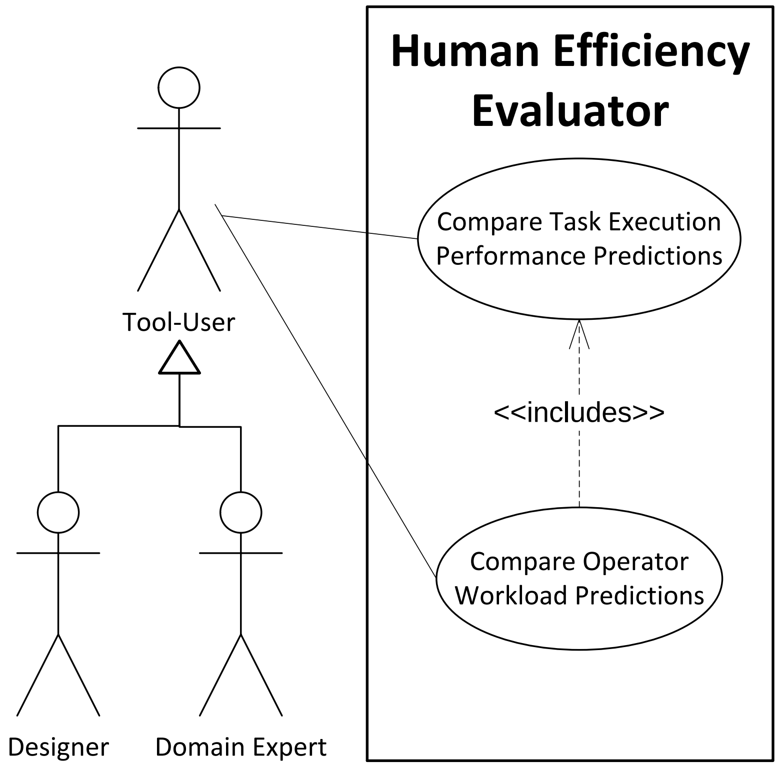


Figure 4: UML Use Case Diagram of the Human Efficiency Evaluator

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.

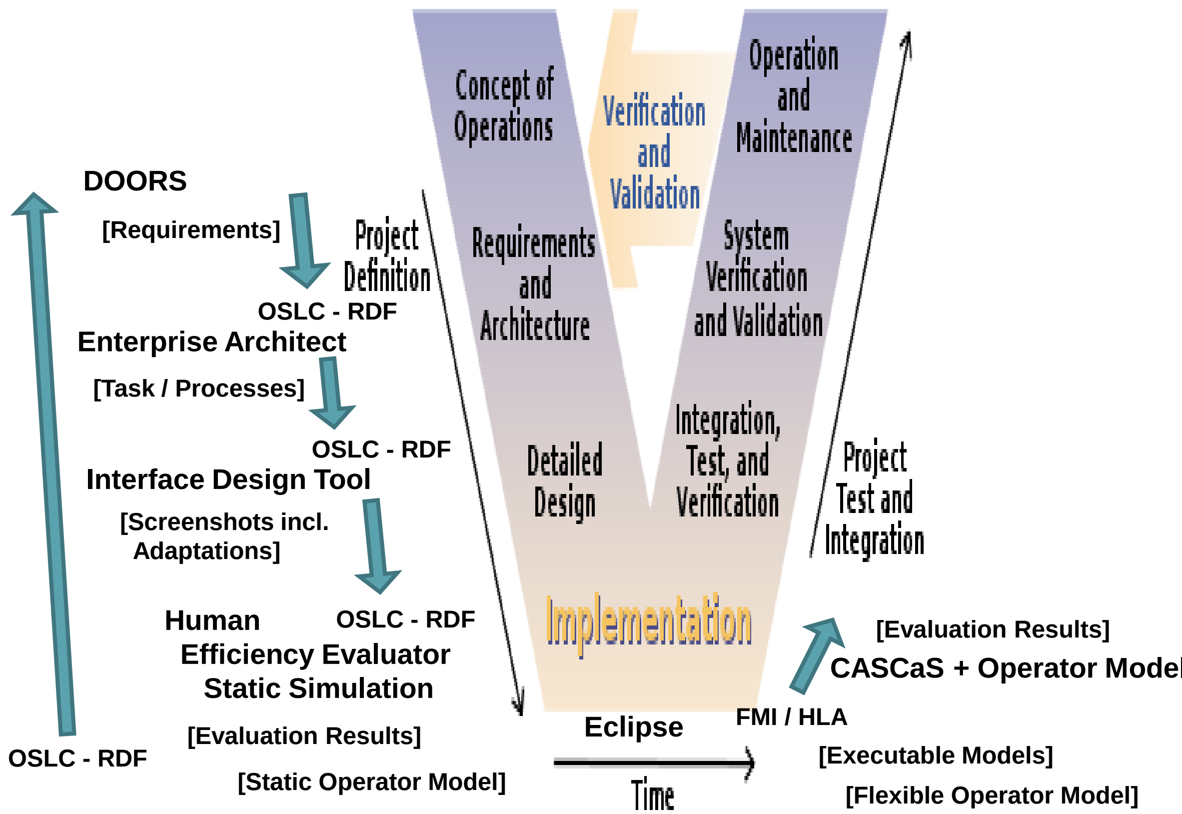


Figure 5: Human Efficiency Evaluator as part of a development process based on the V-Model.

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 5 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 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.

2.3.2 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 6 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 4, two different actors define the target audience of the tool: First, interface and instrument designers and second, domain experts (which are AdCoS users).

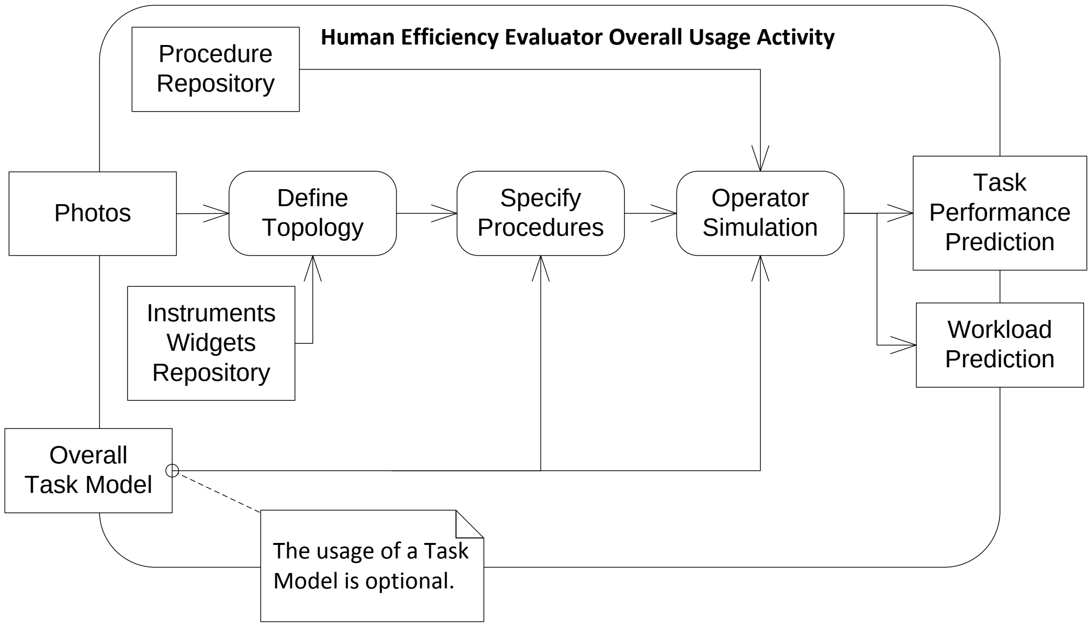


Figure 6: UML Activity Diagram of the Human Efficiency Evaluator overall activities.

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 6) 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).

2.3.2.1 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.

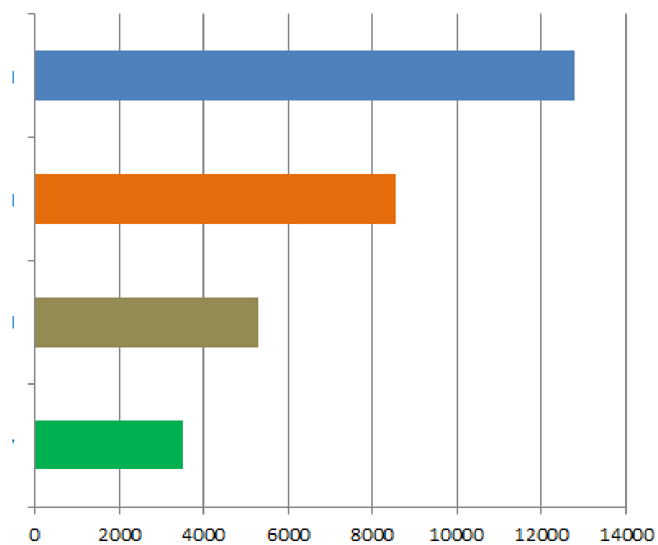


Figure 7: Exemplary performance comparison of different design or adaptation variants.

Figure 7 illustrates an exemplary task performance prediction of the HEE. Different design variants are compared on a millisecond scale.

2.3.2.2 Compare Operator Workload Predictions

After the task performance has been predicted for each design or adaptation variant, the operator workload can be predicted. Figure 8 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
.

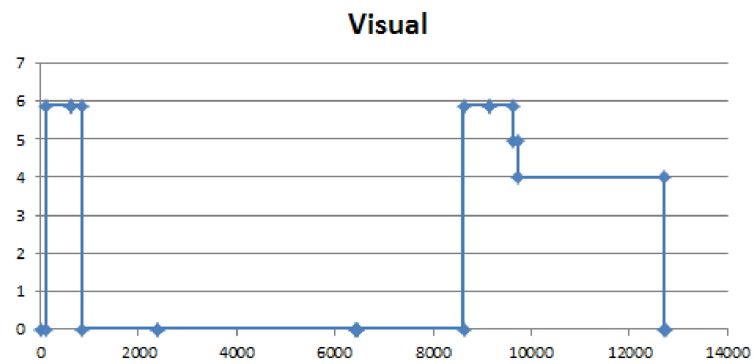


Figure 8: Exemplary workload prediction of the human operator's visual perception

2.3.3 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	
Activity Management Guidance	WP6_IGS_HEA_REQ06 WP6_IGS_HEA_REQ09
OpenEHR system	WP6_ATO_HEA_REQ08 WP6_ATO_HEA_REQ09 WP6_AWI_HEA_REQ03 WP6_AWI_HEA_REQ14
WP7, Aeronautical	
Adaptive diversion airport advisory	WP7_AER_UC1_HON_v01
WP8, Control rooms	
Operator Overload and Underload	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	
Support agent in LC manoeuvre	WP9_CRF_AUT_UC1

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

2.3.3.1 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.

2.3.3.2 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, 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-layouting 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 5 – 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 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).

2.3.3.3 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.

2.3.4 Input

Like illustrated in Figure 6 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).

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

2.3.5 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.

2.4 TrainingManager (OFF, TRS)

2.4.1 Purpose

The TrainingManager will be developed by OFF in cooperation with TRS within HoliDes. Objective is to develop a tool(-chain), which allows modelling of all aspects of a transition training (i.e. from one aircraft type to another), in terms of

- Procedures to be trained by the trainee (SOPs)
- Flight Crew Licensing Requirement (FCLRs; coming from Regulations)
- Training syllabi, including flight phases, scenarios, …

- Learning Knowledge

The TrainingManager will take the SOPs from two different aircrafts, and will create new training syllabi based on a scientific approach, by using the differences of the SOPs. It will also take into account latest knowledge on learning theory and practice.

2.4.2 Use Cases

The use case is to derive new training syllabi for transition from one aircraft to another, based on the SOPs and needed FCLRs.

2.4.3 AdCoS Use-Cases

The TrainingManager will be applied in the aeronautics domain in Use Case 2 "Adaptive Flight Crew Simulator Transition Training".

2.4.4 Input

Formal task descriptions, as specified with PED.

2.4.5 Output

A training syllabus in form of a report, containing different Simulator Lessons, as specified in the following figure:

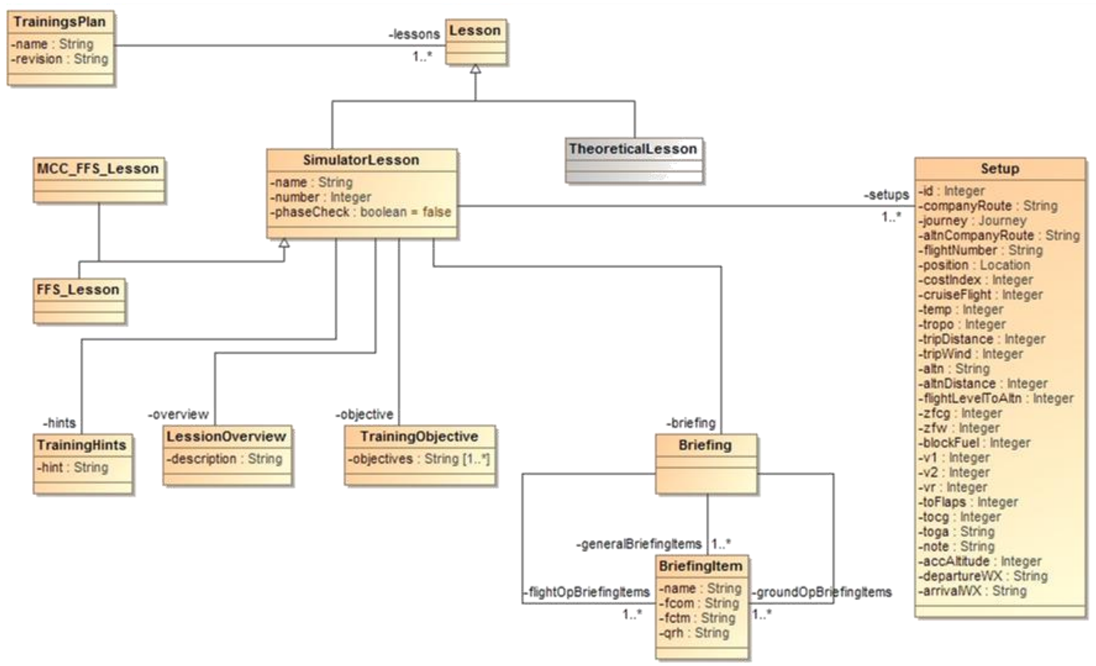


Figure 9: Meta-Modell Syllabus

2.5 GreatSPN (UTO)

2.5.1 Purpose

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

The framework supports the following functionalities:

- Structural analysis - place and transition invariants, deadlocks, boundedness, mutual exclusion, …
- Properties derivable with linear programming - such as upper and lower bounds for places and transition throughputs.
- State space generation, using advanced - techniques like symbolic data structures.
- Verification of logical and behavioural properties

- Numerical analysis of quantitative properties - such as average place distributions, expected transition throughputs, probability of exposing a specific behaviour defined using the CSLTA logic, etc…
- Simulation techniques available for very large model, where the construction of the reachability graph is impracticable.
- Optimization problem, described in the form of Markov Decision Processes (MDP).

2.5.2 Use Cases

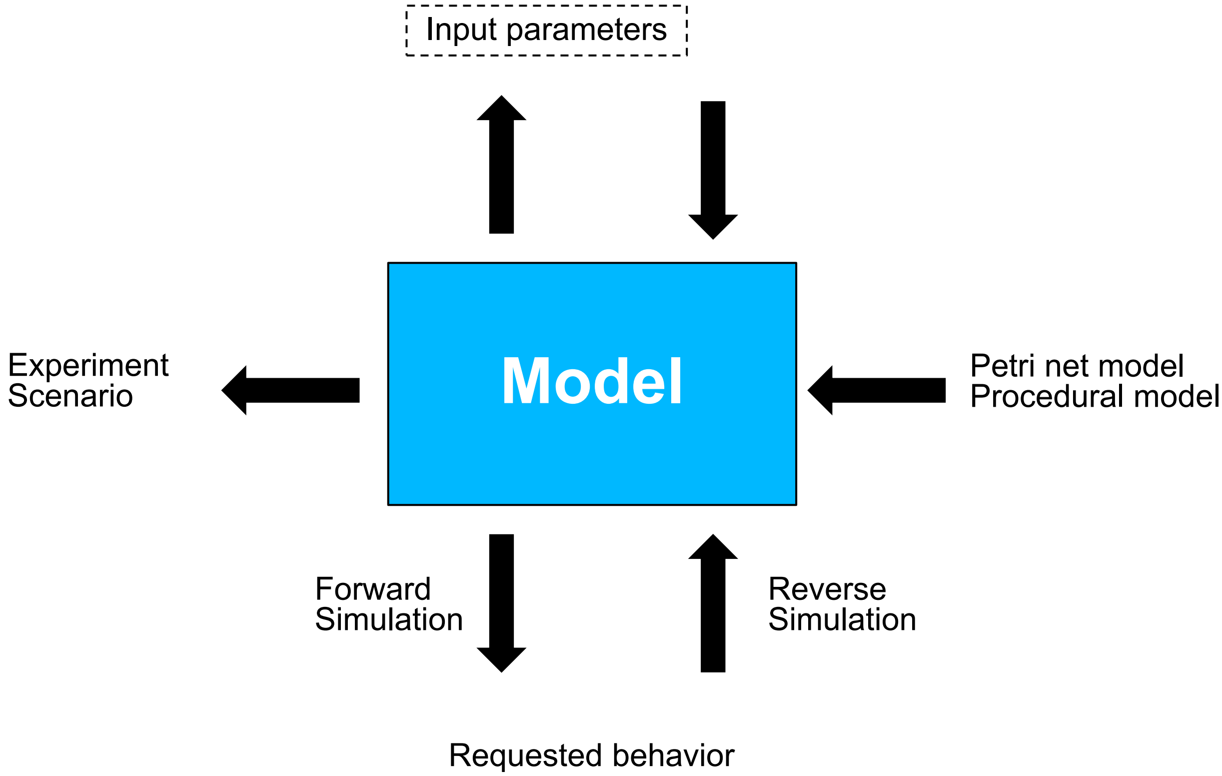
In general the tool can be used to model and validate any finite state machine. For aeronautics domain a special use-case of 'creating an experimental scenario' is proposed that should make use of the simulation/evaluation capability of the tool. If proved feasible, this use-case can be applied cross-domain.

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.

Using GreatSPN, it is believed that a model can be created easily and the tool can do

- Verification of formal logic consistence of the model (i.e. no dead-locks, place accessibility etc.)
- 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.



2.5.3 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. He can annotate building blocks with rational / explanation. The system verifies the validity of the model on the fly and warns the user if he tries to do invalid operation.
- 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 form, in which user can write values for specific variables. The user can change names of variables and

add new ones. He 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. Specifically it searches for dead-ends, infinite loops etc.
- Report - When the user is satisfied with the performance of the model, he lets the system create 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.

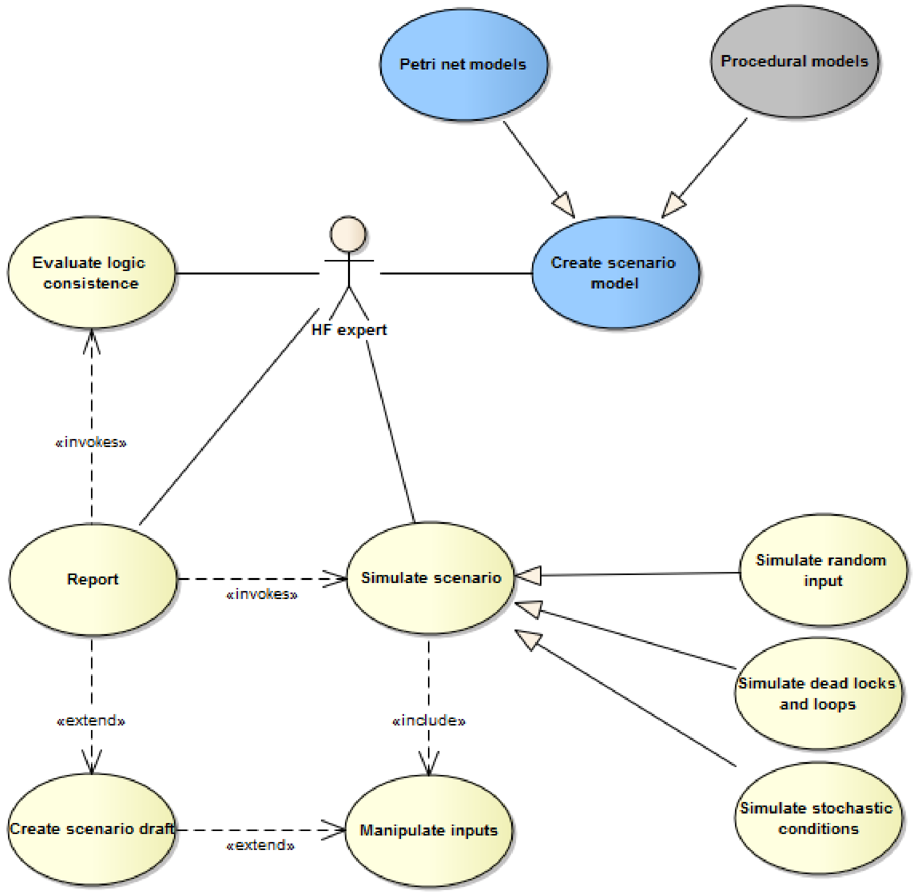


Figure 10: GreatSPN Use Case

2.5.4 Input

All input is provided by the tool together with the modelling and domain knowledge a user needs to have. If used in future, a database of models or re-usable components would be useful.

2.5.5 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.

2.6 Detection of driver distraction based on in-car measures (TWT)

This tool will be developed by TWT.

2.6.1 Purpose

Distraction during driving leads to a delay in recognition of information that is necessary to safely perform the driving task (Regan und Young 2003). Thus, distraction is one of the most frequent causes for car accidents (Artho et al., Horberry et al. 2006). Four different forms of distraction are distinguished, although not mutually exclusive: visual, auditory, bio-mechanical (physical), and cognitive. Human attention is selective and not all sensory information is processed (consciously). When people perform two complex tasks simultaneously, such as driving and having a demanding conversation, the brain shifts its focus. This kind of attention shifting might also occur unconsciously. Driving performance can thus be impaired when filtered information is not encoded into working memory and so critical warnings and safety hazards can be missed (Trick et al. 2004). Sources for distraction of the driver can be located within and outside of the car.
An acoustic analysis including the detection of the number of speakers, the degree of emotional content, information about the driver's involvement in the conversation (e.g., whether the driver himself is speaking), is used for the prediction of the driver's degree of distraction. In addition, eye-tracking signals, such as temporal measures of eye movements, and face movement information, such as mouth movements, can be exploited to increase the reliability of distraction prediction. A computational and empirical cognitive distraction model is used for analysing the different signals, with the aim of computing a "distraction degree" of the driver. The effect of cognitive distraction, based on different audio scenarios, on driving performance will be empirically tested in a parallel task in order to assess the impact of auditory stimuli on distraction.

2.6.2 Use Cases

Deriving knowledge about the human operator can be very valuable in the system validation phase. While interacting with a prototype or some modules of the AdCoS, the operator's degree of distraction can be evaluated. The tool provides feedback whether or not a new system (module) increases or decreases the operator's degree of distraction.

In addition, this tool bears the potential to be used online to classify the driver's distraction not only during testing of a prototype, but also during everyday interaction with the AdCoS. This online measure of distraction could in turn be used to adapt the degree of automation of the AdCoS to the driver's state.
A combination with the tools developed by BUT and UTO is possible to increase the tool's predictive power.

2.6.3 AdCoS Use-Cases

This method will be either applied to the frontal collision use case or to the overtaking use case of WP9.

2.6.4 Input

In-vehicle information is needed. This includes, but it is not limited to, in-car audio recordings and eye-tracking data from the driver. These data need to be stored in a way that enables linking them to certain system states, e.g., inputs from the user to the system. Thus multimodal data integration and synchronization is mandatory for the tool to produce meaningful results.

2.6.5 Output

The tool provides a temporal description of the driver's degree of distraction. The metrics used to quantify the driver's distraction based on in-car information are developed in T5.2. The different measurements will be integrated in RTMaps provided by INTEMPORA. Personal components of the cognitive model and computations are intended to be mobile, e.g., via a Smartphone App. The core of the App, the personal model, should be exchangeable between the mobile device and an on-board system.

2.7 COSMODRIVE and COSMO-SIVIC (IFS)

COSMODRIVE is a <u>CO</u>gnitive <u>S</u>imulation <u>MO</u>del of the car <u>DRIVE</u>r that will be developed by IFSTTAR. Then, in the frame of WP4, this driver model will be interfaced with 2 additional tools; Pro-SIVIC (provided by CIVITEC) and RT-MAPS (provided by 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.

2.7.1 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.

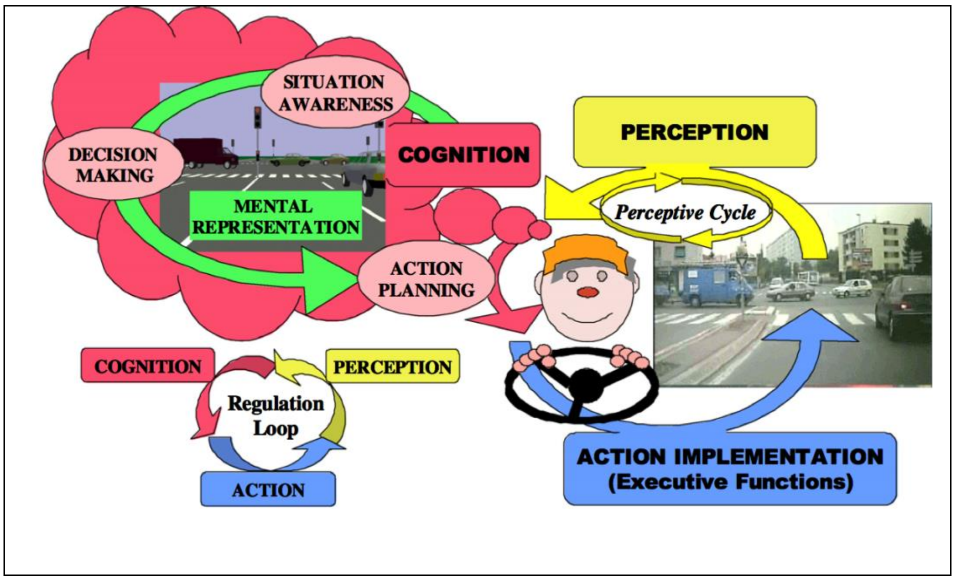


Figure 11: general architecture of COSMODRIVE Model

Synthetically, the general architecture of COSMODRIVE model (Figure 11) is made of 3 main modules (Bellet et al, 2012):

- A Perception Module: This module is in charge of simulating 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.
- A Cognition Module: This module is primarily in charge to 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)

- An Action Module: This module is in charge of implementing 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 models in HOLIDES 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 events, erroneous situational awareness, or inadequate behavioural performance, due to visual distractions (for instance caused by the execution of a secondary task performed while driving).

2.7.2 Use Cases

In 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:

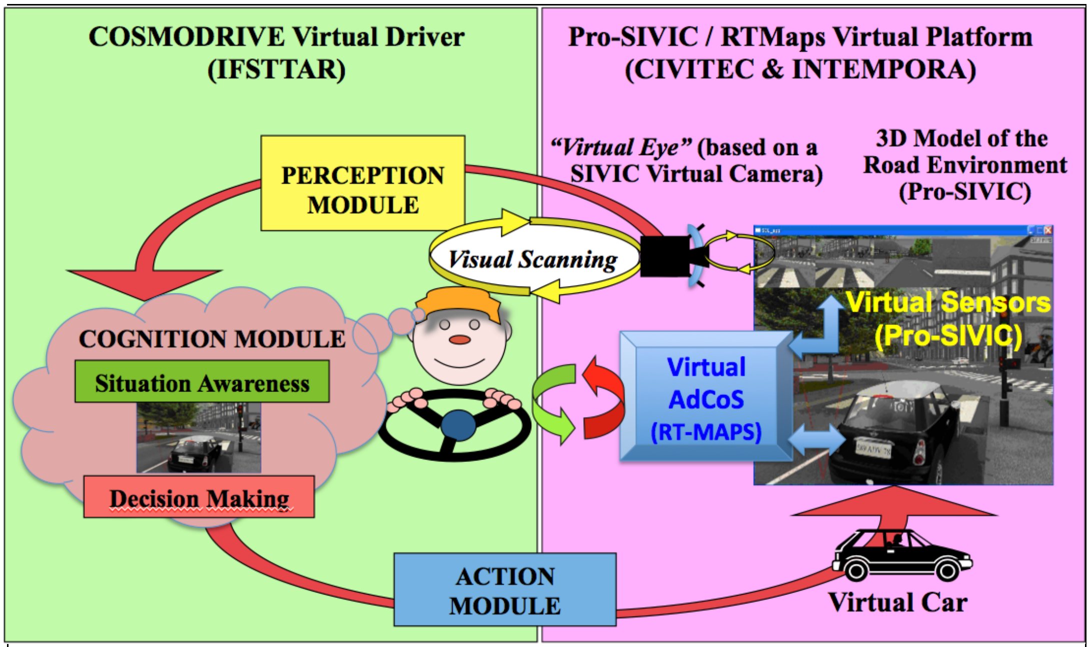


Figure 12: Overview of the COSMO-SIVIC platform (COSMODRIVE + Pro-SIVIC + RT-MAPS)

From the work to be done in WP2 and in WP4 - in partnership with CIVITEC and INTEMPORA - it is expected to have, at the end, 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).

2.7.3 AdCoS Use Cases

Pro-SIVIC and RT-MAPS will be also used 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 integrated COSMO-SIVIC platform presented in Figure 12 (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 driver model (CODMODRIVE) 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 steps, 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.

2.7.4 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 are already listed in the Holides requirements documents (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).

2.7.5 Output

At the COSMODRIVE level, the outputs generated by this driver model will primarily concerns driving behaviours in terms of both drivers' actions on 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 software; cf. WP9_IFS_AUT_REQ07: "Recording/using of eye-tracking data to assess driver' visual distraction").

With COSMO-SIVIC virtual Human Centred Design 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" and "new risks" liable to occur) before implementing them on a real car. For that, it will be necessary 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 the AdCoS work 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).

3 References

Artho, J.; Schneider, S.; Boss, C.: Unaufmerksamkeit und Ablenkung: Was macht der Mensch am Steuer? - Transport Research International Documentation - TRID. Online: http://trid.trb.org/view.aspx?id=1244037, last access 21.05.2014.

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.; Hjälmdahl, M.; Lüdtke 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.

González--Calleros, J. M., Osterloh, J. P., Feil, R., & Lüdtke, 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

Horberry, Tim; Anderson, Janet; Regan, Michael A.; Triggs, Thomas J.; Brown, John (2006): Driver distraction: the effects of concurrent in-vehicle tasks, road environment complexity and age on driving performance. In: *Accid Anal Prev* 38 (1), S. 185–191. DOI: 10.1016/j.aap.2005.09.007.

Lenk, Jan Charles; Droste, Rainer; Sobiech, Cilli; Lüdtke, Andreas; Hahn, Axel (2012): *Towards Cooperative Cognitive Models in Multi-Agent Systems*. In proceedings of COGNITIVE 2012, The Fourth International Conference on Advanced Cognitive Technologies and Applications; pp. 67-70, ISBN: 978-1-61208-218-9, Nice, France

Lüdtke, A., Frische, F., & Osterloh, J.-P. (2011). Validation of a Digital Human Model for Predicting Flight Crew- Aircraft Cockpit Interaction. In Tagungsband Berliner Werkstatt für Mensch-Maschine Systeme. Berlin.

Osterloh, J.--P., Bracker, H., Müller, H., Kelsch, J., Schneider, B., & Lüdtke, A. (2013). DCoS--XML: A Modelling Language for Dynamic Distributed Cooperative Systems. In Proceedings of the 2013 11th IEEE International Conference on Industrial Informatics (INDIN).

Regan, M. A.; Young, K. L. (2003): Driver distraction: a review of the literature and recommendations for countermeasure development. In: *Proc. Australas. Road Safety Res. Policing Educ. Conf.* 7 (v1), S. 220–227.

Trick, Lana M.; Enns, James T.; Mills, Jessica; Vavrik, John (2004): Paying attention behind the wheel: a framework for studying the role of attention in driving. In: *Theoretical Issues in Ergonomics Science* 5 (5), S. 385–424. DOI: 10.1080/14639220412331298938.