



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

HoliDes

**D7.8 Tailored HF-RTP and Methodology Vs1.8 for the
 Aeronautics Domain**

Project Number:	332933
Classification:	Public
Work Package(s):	WP7
Milestone:	MS5
Document Version:	V2.0
Issue Date:	28.04.2016
Document Timescale:	Project Start Date: October 1, 2013
Start of the Document:	Month 29
Final version due:	Month 30
Deliverable Overview:	<p>Main document: D7.8 Tailored HF-RTP and Methodology Vs1.8 for the Aeronautics Domain</p> <p>Annex I: ResourceShape Task Model in RDF</p> <p>Annex II: ResourceShape Training Model in RDF</p> <p>Annex III: Training Syllabus V3 – Confidential</p>
Compiled by:	Maros Raucina (HON)
Authors:	Zdenek Moravek (HON) Sara Sillaurren (TEC) Marieke Thurlings (TWT) Adam Herout (BUT) Bohuslav Krena (BUT) Jan-Patrick Osterloh (OFF) Frank Rister (TRS)
Reviewers:	Gert Weller (TAK) Stefan Kaufmann (IAS)
Technical Approval:	Jens Gärtner, Airbus Group Innovations



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



RECORD OF REVISION

Date (DD.MM.JJ)	Status Description	Author
21.03.16	Draft structure of deliverable	Zdenek Moravek
18.04.16	EATT input	Jan-Patrick Osterloh
19.04.16	DivA input incl. inputs from partners	Zdenek Moravek
19.04.16	Draft ready for consortium review	Maros Raucina
20.04.16	Internal Review	Stefan Kaufmann
22.4.16	Internal Review	Gert Weller
27.4.16	Update after review	Jan-Patrick Osterloh
28.4.16	Final version	Maros Raucina



Table of Contents

- 1 Introduction8**
 - 1.1 Objective of the document..... 8
 - 1.2 Structure of the document..... 10
- 2 Tailoring Methodology for WP710**
- 3 Re-test of the Platform Builder12**
 - 3.1 Tests for Diversion assistant AdCoS 12
 - 3.2 Tests for Training AdCoS 13
- 4 HF-RTP Instance for Diversion assistant.....14**
 - 4.1 Tailoring Rules 16
 - 4.2 ANaCoNDA and AdCoS 16
 - 4.2.1 The Interfaces..... 17
 - 4.2.2 The implementation 20
 - 4.2.3 ANaCoNDA aspects on the development process 23
 - 4.3 Missed event detector (MED) and AdCoS 24
 - 4.3.1 The interfaces 24
 - 4.3.2 The implementation 25
 - 4.3.3 MED aspects on the development process..... 26
 - 4.4 Pattern classifier and AdCoS 27
 - 4.4.1 The interfaces 27
 - 4.4.2 The implementation 29
 - 4.4.3 Pattern classifier aspects on the development process..... 31
 - 4.5 Cognitive Distraction Classifier (CDC)..... 31
 - 4.5.1 The interfaces 32
 - 4.5.2 The implementation 33
 - 4.5.3 CDC aspects on the development process 34
 - 4.6 RTMaps and AdCoS 34
 - 4.6.1 The interfaces 35
 - 4.6.2 The implementation 37
 - 4.6.3 RTMaps aspects on the development process 38
 - 4.7 Experiment Data Archive (EDA) and RTMaps..... 38
 - 4.7.1 The interfaces 39



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



- 4.7.2 The implementation 47
- 4.7.3 EDA aspects on the development process 51
- 5 HF-RTP Instance for EATT..... 52**
- 5.1 Tailoring Rules 55
- 5.2 MagicPED and AdCoS 56
 - 5.2.1 The Interfaces..... 56
 - 5.2.2 The implementation 57
 - 5.2.3 MagicPED’s aspects on the development process 57
- 5.3 TrainingManager and AdCoS 58
 - 5.3.1 The Interfaces..... 58
 - 5.3.2 The implementation 60
 - 5.3.3 TrainingManager aspects on the development process 62
- 6 Conclusion and Summary..... 63**
- 7 Way forward and upcoming activities 65**
- 7.1 Diversion assistant use-case 65
- 7.2 EATT use-case..... 65
- 8 Annex 66**
- 8.1 Annex I 66
- 8.2 Annex II 67
- 8.3 Annex III 68

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

Executive Summary

The following document shows the latest version (1.8) of the HF-RTP for the aeronautics domain. It covers two instances of the HF-RTP one applied for the cockpit AdCoS – the diversion assistant, the other for the training AdCoS – EATT.

The development of an RTP is highly iterative and with many documents and lots of contents have already been submitted. As such, this document does not seek to repeat what has already been written instead only new content or elaboration on previous topics will be written, namely the integration status of tools from the tailored HF-RTP instance for each individual AdCoS, improvement of HF process supported by the use of the HF-RTP instance and overall assessment of benefits provided by the tailored HF-RTP instance. Existing work which supports what is documented here will be referenced rather than be repeated.



HoliDes

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

HoliDes

Glossary

AdCoS = Adaptive Cooperative Human-Machine Systems

AFLW = Annotated Facial Landmarks in the Wild dataset

ANaCoNDA = Adaptable NAtive-code CONcurrency-focused Dynamic Analysis Framework

API = Application Program Interface

BCI = Brain Computer Interface

CDC = Cognitive Distraction Classifier

CLI = Command Line Interface

CSV = Comma Separated Value File

EATT = European Air Transport Training

EDA = Experiment Data Archive Tool

EEG = Electro-encephalogram

EFB = Electronic Flight Bag

ETA = Experiment Tagging and Annotating Tool

HF = Human Factors

HF-RTP = Human Factors Reference Technology Platform

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

JSON = JavaScript Object Notation

KNN = k-Nearest Neighbors Algorithm

MED = Missed Event Detection Tool

OSLC = Open Services for Lifecycle Collaboration

ROI = Region of Interest

RTMaps = Real-Time, Multisensor Application Framework

RTP = Reference Technology Platform

SDK = Software Development Kit

UDP = User Datagram Protocol

WP = Work Package

XML = EXtensible Markup Language

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

1 Introduction

This deliverable describes how the HF-RTP methodology Vs1.8 and the HF-RTP, which are being developed in WP1, are applied and tailored in the Aeronautics Domain.

The tailoring of HF-RTP for a specific use-case is a process of selection, adaptation and integration of tools from HF-RTP to address specific requirements of the use-case. The tools are applied individually or in connection with other tools, i.e. tool chains, to improve the development process or support functions of a system being developed.

If tools are to be connected, HF-RTP provides means of communication so that the tools can automatically exchange data or services. To achieve this ideal state of interoperability, the tailoring considers building communication adapters wherever it is needed.

1.1 Objective of the document

Deliverable D7.8 describes the results of the HF-RTP tailoring methodology applied to the Aeronautics Domain for the second half of project cycle 2.



HoliDes

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

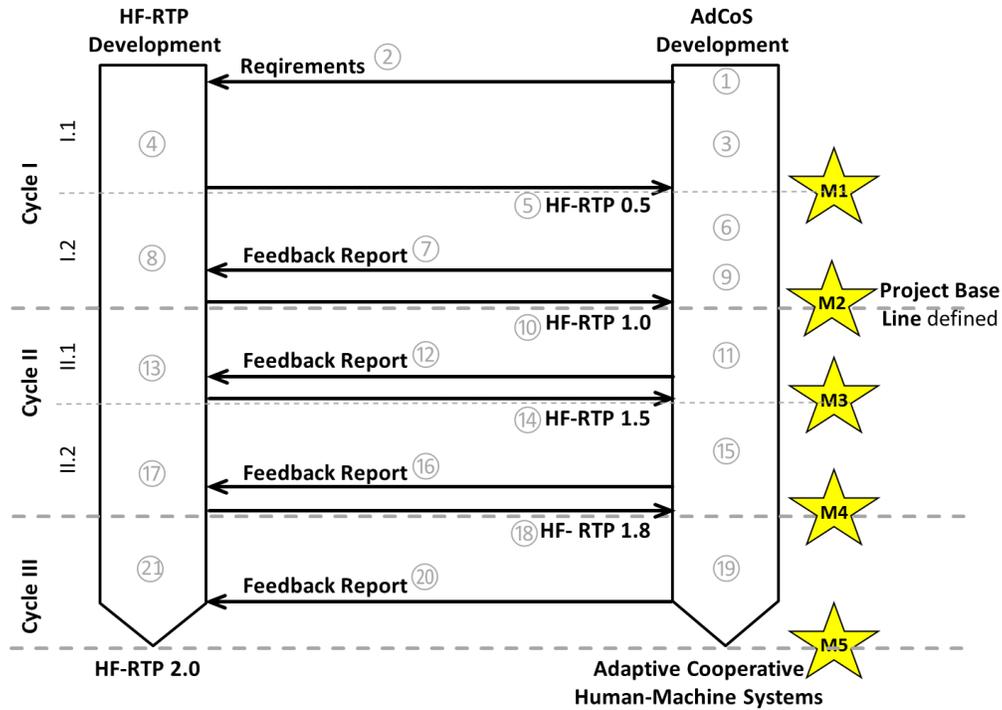


Figure 1: Overall workflow taken from the HoliDes proposal

The HF-RTP and the tailoring methodology (version 1.8) developed in WP1 and delivered in D1.6 are applied to the AdCoS of the Aeronautics Domain.

The previous version for tailoring of the HF-RTP in WP7 has already been provided in D7.6, which was based on the HF-RTP version 1.5 (M4).

1.2 Structure of the document

For each use-case the tailoring of HF-RTP is described with respect to progress from the previous version of the tailoring. Mainly, the document focuses to describing interfaces between tools and AdCoS and to the actual state of the implementation.

For each tool that makes specific HF-RTP instance, the following sections are provided

- Description of interfaces – input/output data and connectivity with AdCoS or other tools
- Status of implementation – current status, preliminary tests and verifications
- Aspects to development process – how tool changes the current practice especially from the HF perspective

Two use-cases are described for the Aeronautics Domain – the main use-case for Diversion assistant as a cockpit application, and the transition training use-case.

2 Tailoring Methodology for WP7

Work package 1 has defined a series of tailoring steps for the development of an HF-RTP instance. (The HF-RTP is the entire collection of HF MTTs in the HoliDes project and the HF-RTP instance being an instantiation of those tools in an integrated manner.)

A full description of the tailoring rules can be found in *Annex II of D1.6 HF-RTP Version 1.8 Inc. Methodology and Requirements Analysis update*.

Table 1 shows the tailoring steps defined in WP1. For brevity, we will omit a description of these steps.

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

Table 1 - The Tailoring steps as defined by Annex II of D1.8

Tailoring steps
1. Identification of issues in the existing development process
2. Selection of methods, techniques and tools (MTTs)
3. Describe what information needs to be exchanged between the existing life-cycle tools of each AdCoS' toolchain and the MTTs to be integrated into it
4. Implementation of MTT mappings

Two different categories of software have been developed as MTTs:

- **Lifecycle tools:** These tools are meant to be integrated into the tool chains of the AdCoS owners to improve their development process and deal with lifecycle data such as requirements, test cases, analysis results and behavioural models (e.g., Magic-PED or OFF for the task modelling)
- **Non-lifecycle tools:** They have been developed to be embedded into the AdCoS and share real-time data with the AdCoS to improve its functionalities (e.g., the Driver Distraction Classifier of TWT developed in RT-Maps).

For the lifecycle tools, the integration into the HF-RTP implies the development of specific piece of software (i.e., adapters) to implement the OSLC specification and allow the sharing of data with other OSLC-compliant tools.

On the other hand, the non-lifecycle tools are not OSLC compliant, but are embedded in the AdCoS they are planned for, because the linked data approach of the IOS is an unnecessary overhead (in terms of implementation effort as well as performance cost at runtime) when integration of two local applications running on the same machine is needed. Since tailoring steps 3 and 4 (see Table 1 - The Tailoring steps as defined by Annex II of D1.8) refers to lifecycle integration, a tailoring up to step 2 is enough for integrating non-lifecycle tools.



3 Re-test of the Platform Builder

3.1 Tests for Diversion assistant AdCoS

In the WP7 Diversion assistant use case, the following tools had been identified as potentially useful and, therefore, would expect to be available through the platform builder:

- Eye Gaze Recordings
- Cognitive Distraction Detection Tool
- Pattern Classifier
- EDA tool
- RTMaps
- ANaCoNDA

For more information on the MTT investigation, please refer to D1.5.

Table 2 - Platform Builder test Cases DivA

Platform Builder Test Cases <i>Diversion assistant</i>			
<i>Test No.</i>	<i>Description</i>	<i>Expected Result</i>	<i>Actual Result</i>
1.	Select the Aeronautics domain and under the HF Issue <i>Attention</i> locate the following MTT: Eye Gaze Recordings	Eye Gaze Recordings appears as an MTT on the screen.	
2.	Select the Aeronautics domain and under the HF Issue <i>Distraction</i> locate the following MTT: Eye Gaze Recordings	Eye Gaze Recordings appears as an MTT on the screen.	
3.	Select the Aeronautics domain and under the HF Issue <i>Distraction</i> locate the following MTT: Cognitive Distraction Detection Tool	Cognitive Distraction Detection Tool appears as an MTT on the screen.	
4.	Select the Aeronautics domain and under the HF Issue <i>Workload</i> locate the following MTT: Pattern Classifier	Pattern Classifier appears as an MTT on the screen.	

Platform Builder Test Cases Diversion assistant			
<i>Test No.</i>	<i>Description</i>	<i>Expected Result</i>	<i>Actual Result</i>
5.	Select the Aeronautics domain and under the Related activity <i>Evaluation</i> locate the following MTT: EDA tool	EDA tool appears as an MTT on the screen.	
6.	Select the Aeronautics domain and under the Related activity <i>Evaluation</i> locate the following MTT: RTMaps	RTMaps appears as an MTT on the screen.	
7.	Select the Aeronautics domain and under the Related activity <i>Evaluation</i> locate the following MTT: ANaCoNDA	ANaCoNDA appears as an MTT on the screen.	
8.	Select the Aeronautics domain and under the Related activity <i>System implementation</i> locate the following MTT: ANaCoNDA	ANaCoNDA appears as an MTT on the screen.	
9.	The user is to create a project which contains all of the selected MTTs.	A named container is made available which houses all of the selected MTTs.	

As the table above illustrates, all of the identified MTTs were present in the Platform Builder. However not all MTTs are present in categories, where they were expected to be. This is the case for Eye Gaze Recordings and ANaCoNDA which were expected to show up in one category but appeared in another.

This feedback will be sent to WP1 for improvements in the next version.

3.2 Tests for Training AdCoS

Table 3 - Platform Builder test Cases Training

Platform Builder Test Cases Training AdCoS			
<i>Test No.</i>	<i>Description</i>	<i>Expected Result</i>	<i>Actual Result</i>

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

Platform Builder Test Cases Training AdCoS			
<i>Test No.</i>	<i>Description</i>	<i>Expected Result</i>	<i>Actual Result</i>
1.	Select the Aeronautics domain and locate the following MTT: MagicPED	MagicPED appears as an MTT on the screen.	
2.	Select the Aeronautics domain and under the HF Issue <i>Training</i> locate the following MTT: TrainingManager	TrainingManager appears as an MTT on the screen.	HF issue Training is missing, TrainingManager is also missing

This feedback will be sent to WP1 for improvements in the next version.

4 HF-RTP Instance for Diversion assistant

The HF-RTP instance for Diversion assistant will support both, the functions of the system and the development process. It is to be said that the instance or its compounds are not restricted for the Diversion assistant, but are intended to be used for future cases with little modifications. The HF-RTP instance is supposed to bring the following benefits

- A. **Operator state models:** tools as MED, cognitive distraction monitor and pattern classifier provide comprehensive functions that can detect state of an operator, i.e. pilot, and trigger mitigation actions via the selected modes of adaption as described in D3.5 Techniques and Tools for Adaption Deliverable. In tailoring, the AdCoS is endowed with ability to read state related data from these tools. The tools can be used individually (deployment) or in connection with data management tools (validation), as described in item C.
- B. **System verification and validation:** ANaCoNDA tool supports the isolation of thread-race related SW bugs with increased propensity. The thread-race related bugs are difficult to resolve and they often persist until validations where they cause data loss and delays in testing.
- C. **Data management:** a toolchain of RTMaps, ETA and EDA supports validations in data collection, archiving and analysis. The tools were specifically selected to address problems with synchronization of data from different data sources, to reduce experimenter head-down time and to simplify data analysis. The toolchain can be extended to process

data from operator state models, i.e. item A, and from system verification tests, i.e. item B.

The assessment of benefits with respect to the baseline has been described in D1.6 - HF-RTP Version 1.8 incl. Methodology and Requirements Analysis Update & 2nd year Baseline assessment. Since then, the HF-RTP instance for Diversion assistant has been extended as is shown in the updated development process in Figure 2.

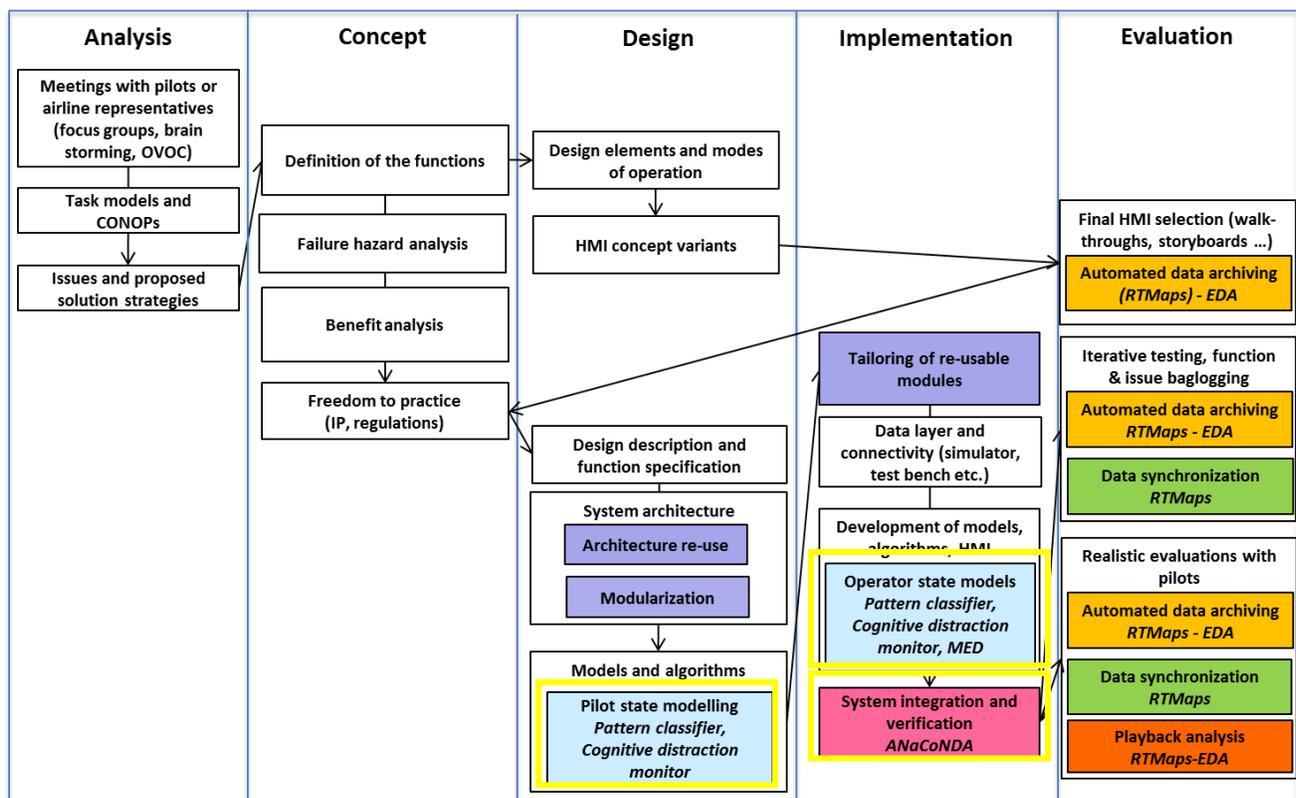


Figure 2: Updated development process for Diversion assistant using the HF-RTP instance. The updated aspects are highlighted with yellow frames.

The HF-RTP is supposed to impact preferentially HF aspects of the development process. The particular selection of HF-RTP instance made for Diversion assistant improves the HF aspects of the development process in the following way

- It integrates pilot state models and inference functions in the AdCoS for both – validation and deployment. The following states are addressed: workload, fatigue, and distraction/attention.
- It improves the performance of HF experts while executing an experiment by semi-autonomous notes taking that reduces the undesired head-down time of the experimenter.
- It simplifies post-experiment processing and normalization of HF data by automatic synchronization of various data sources, data categorization and archiving. These tasks that formerly were done by significant effort of HF experts, can now be made automatically and already in course of an experiment.
- It allows for novel cross-project data searches and interpretations, e.g. estimating previous exposure of a subject to objectives of an experiment.

Additionally, the HF-RTP instance support non-HF work, such as system design and verification.

Details will be given in the following sections in connection with particular tools.

4.1 Tailoring Rules

MTT	Lifecycle or Non-Lifecycle	Applied Tailoring Rules
ANaCoNDA	Lifecycle	1-4
MED	Lifecycle	1-4
PatternClassifier	Lifecycle	1-4
CDC	Lifecycle	1-4
RTMaps	Non-Lifecycle	1-2
EDA	Lifecycle	1-4

The following sections will describe the results of the tailoring rules 3 and 4, as well as the benefits gained by applying the tools.

4.2 ANaCoNDA and AdCoS

This section describes the integration between ANaCoNDA and the AdCoS. ANaCoNDA is a framework that simplifies the creation of dynamic analyzers for analyzing multi-threaded C/C++ programs on the binary level. The

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

framework provides a monitoring layer offering notification about important events, such as thread synchronization or memory accesses, so that developers of dynamic analyzers can focus solely on writing the analysis code. ANaCoNDAA also supports noise injection techniques to increase chances to find concurrency-related errors in testing runs.

4.2.1 The Interfaces

The ANaCoNDA framework is built of the following compounds

- the engine that connects to the application to be analysed. The engine handles interaction between the operating system, the framework and the application
- analyzers that interpret the code of the application and run-time data to detect thread-race related issues
- visualizers that interpret, filter and display acquired data.

The framework is modular so that various analysers and visualizers can be used and combined depending on the purpose of the task.

ANaCoNDA starts analysis based on input reference to already running application. In the current version the user starts both, the application and the framework, manually from the command line using the defined start-up interface. ANaCoNDA advises proprietary Intel PIN framework which events should be monitored (see Figure 3, step 2). An execution of application under test is instrumented, executed, and monitored in run-time by one of ANaCoNDAs analysers.

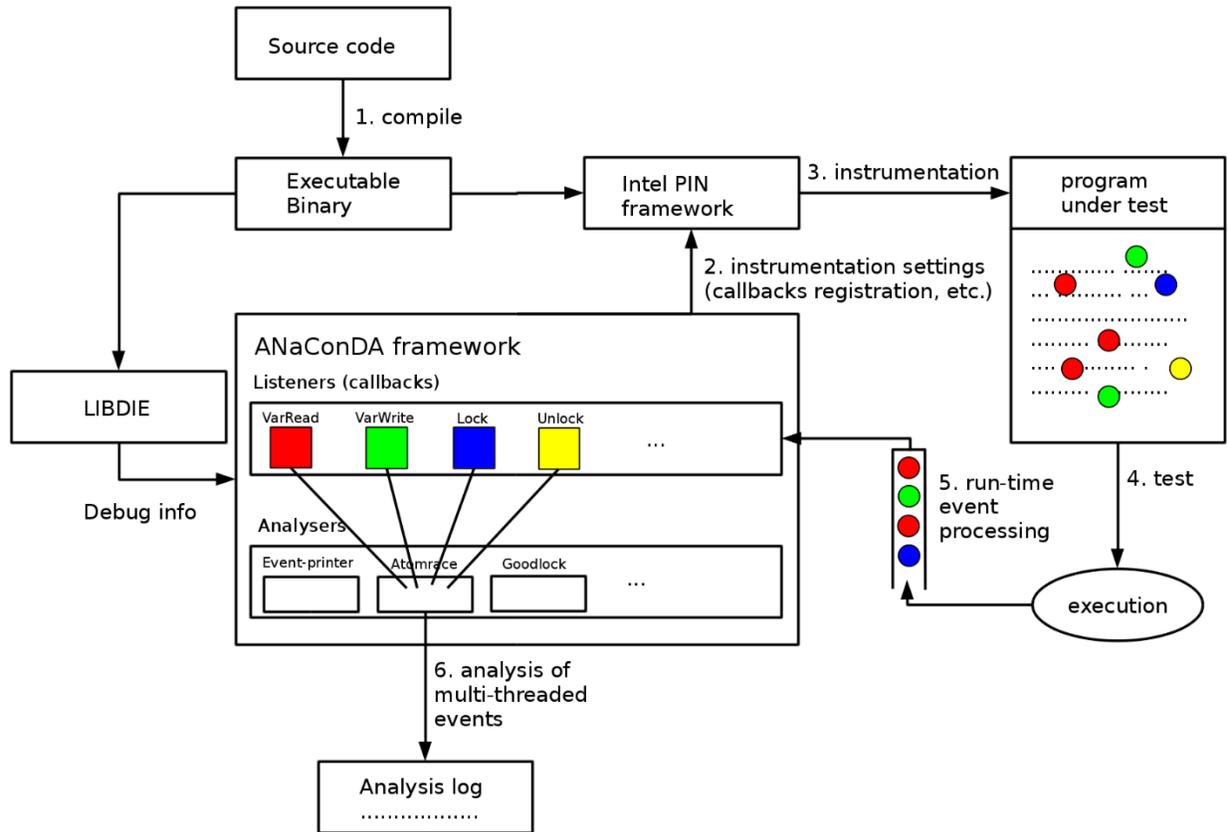


Figure 3: ANaCoNDA architectural model

ANaCoNDA is a CLI application for MS Windows and GNU/Linux systems. It is configured from command line and optional configuration files. The most common way of invoking ANaCoNDA is described in Figure 4.

```
usage:
tools/run.sh [--help] [--run-type { ANaCoNDA | pin | native }] [--config <dir>]
  [--time] [--threads <number>] [--verbose] [--profile]
  [--debug { framework | analyser | program }] [--debugger { gdb | eclipse }]
  <analyser> <program> [<program-parameters>]
```

required arguments:

- <analyser> A name of the analyser to be used.
- <program> A name of the program to be analysed. May be its alias or a path to the executable of the program. If a path to the executable is given, one may also specify the parameters given to it.



```
optional arguments:
--run-type { ANaCoNDA | pin | native }
    Execute the program in ANaCoNDA, PIN or no framework (native run). Default
    is to run the program in ANaCoNDA.
--config <dir>
    A path to a directory containing ANaCoNDA settings. Default is a directory
    containing the default ANaCoNDA settings (framework/conf).
--threads
    A number of threads the analysed program should utilize.
...
```

Figure 4: ANaCoNDA start-up arguments and configurations.

Since ANaCoNDA aims at hunting concurrency bugs, the framework also supports fine-grained combinations of noise. A noise is an external influence of the thread scheduler (i.e., context-switch timing). Noise injection therefore attempts to increase the chances to detect the rare executions that may lead to an error with bigger probability. ANaCoNDA supports specification of noise injection that might be used for memory accesses or lock operations (cf. Figure 5).

```
[noise]
type = yield
frequency = 100
strength = 4
[noise.read]
type = yield
frequency = 200
strength = 8
[noise.write]
type = sleep
frequency = 400
strength = 2
# Global noise settings
# Noise is realized by yield calls
# Noise is injected 100 times per 1000 cases
# Give up the CPU 4 times in a row
# Noise settings for read accesses
# Insert calls to yield
# Inject noise in 20 % of times
# Give up the CPU 8 times in a row
# Noise settings for write accesses
# Insert calls to sleep
# Inject noise in 40 % of times
# Sleep for 2 milliseconds
```

Figure 5: An example of different noise settings for read and writes.

If the output log is not to be visualized, it is stored as a file in the execution folder. As a future step, ANaCoNDA will implement OSLC interface to be able to connect and hand over data to an archiving tool for later analyses and verification tracking. An example of such archiving tool is EDA described in section 4.7.



Example of output file of run-time analysis is depicted in Figure 5. The highlighted lines claim that the analyser spotted a violation of synchronization in a sequence of events which should be executed automatically.

```

C: Thread 0 started
I: Mapping Thread 0 into Thread 0
C: Thread 1 started
I: Mapping Thread 1 into Thread 1
C: Thread 0 forked thread 1
...
C: Thread 5 forked thread 6
C: Before thread 5 joined with thread 6
Exec thread attr:
    policy=SCHED_FIFO, priority=87
Exec thread 42 ...
C: Thread 6: Instance of spoiler _tx_queue_send finished, start=[7,6,3,3,4,2,2],
end=[7,6,3,3,4,2,2]
C: Thread 4: Instance of target _tx_thread_create _tx_queue_create finished,
start=[7,3,2,2,4], end=[7,6,3,3,6]
C: Contract violation detected!
C:   Target [Thread 4]: _tx_thread_create _tx_queue_create
C:   Spoiler [Thread 6]: _tx_queue_send
C: Thread 0 finished
C: Thread 3 finished
C: Thread 4 finished
C: Thread 5 finished
C: Thread 6 finished
Command terminated by signal 2

```

Figure 6: Example of ANaCoNDA output log.

4.2.2 The implementation

The development process of tablet and EFB applications in Honeywell is based on proprietary development environment. The environment must support a platform independent development process and therefore one of the WP7 requirements for tools supporting the development process was the platform independence, WP7_HON_AERO_REQ57_v0.1 in D7.7 Implementation of Aeronautics AdCoS and HF-RTP Requirements Definition Update. The integration of ANaCoNDA in the development environment therefore required the tool to be ported on Windows operating system.

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

The ANaCoNDA framework is installed within the development environment and is associated with other run-time test and verification tools. When it is to be used, ANaCoNDA is started from the execution folder of application to be tested and it produces the output log in that folder. In the future, ANaCoNDA will connect via OSLC to a data archiving tool, such as EDA. The archiving tool will provide interface that will trigger a test session and it will collect testing data and annotate them in a database. The communication scheme is in Figure 7, and it is an analogy of communication between experiment devices and EDA, see Figure 17 and section 4.7.2.

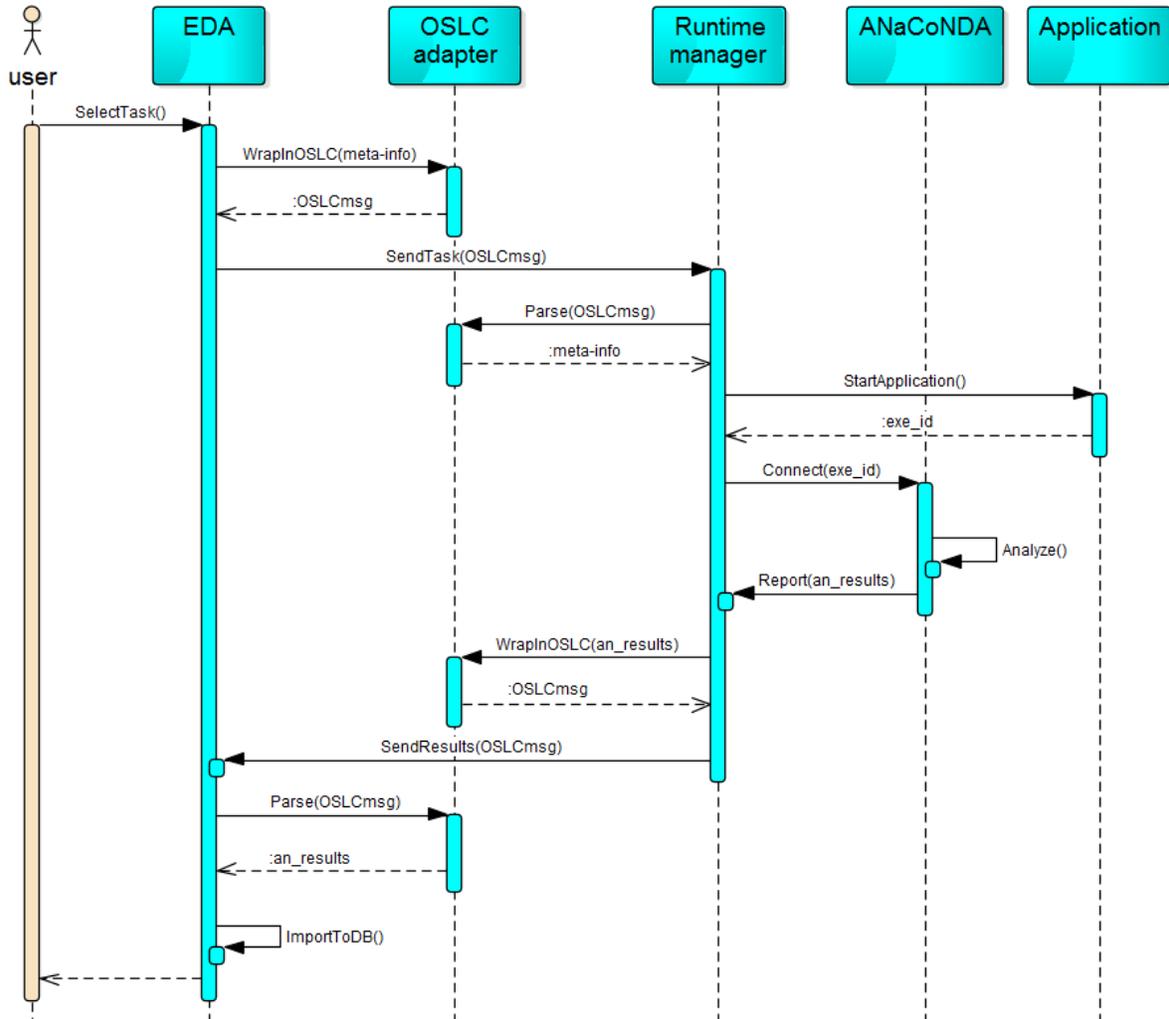


Figure 7: Sequence of actions in planned information exchange between ANaCoNDA and archiving tools such as EDA based on OSLC protocol

As part of the integration, a dedicated test-case was selected for ANaCoNDA to assess suitability for EFB applications. As the Diversion assistant was not ready for this time of test, another application, the HeloBumper, was used instead. HeloBumper is a collision avoidance system for helicopters, using several sensors (currently stereo-camera) to detect probable dangerous obstacles and then warn the crew by indicating the detected obstacle on the on-board EFB device.

HeloBumper had been flight-tested shortly before ANaCoNDA testing and a thread-race issue complicated and prolonged the flight-tests. Applying the ANaCoNDA analysis revealed the source of the issue in a two-hour test session, while in previous development it was discovered and resolved only during the critical phase of flight-tests with significant increase in cost.

The successful test showed that ANaCoNDA can be integrated in the verification phase of the development process.

4.2.3 ANaCoNDA aspects on the development process

Description	ANaCoNDA improves efficiency in software validation and verification with respect to increase the likelihood of occurrence of the thread-race related errors. It can be used from early state of development without previous experience of a problem and it can isolate problematic code in a short time.	
Process	Original <ol style="list-style-type: none"> 0. Assumption: a problem has appeared 1. Reproduction of problem – usually using an excessive number of software execution on multiple computers 2. Determination of conditions – the conditions are vaguely defined due to the nature of problem 3. Narrowing down problematic code area – in iterations repeating steps 1-3. 4. Problem resolution 5. Verification problem was solved by repeating step 1. 	HoliDes update <ol style="list-style-type: none"> 0. Not needed 1. Much lower number of executions needed to invoke a problem if there is one. 2. Not needed 3. Code area is defined exactly with additional information on nature of error. No iterations needed 4. Problem resolution 5. Verification problem was solved by repeating step 1.
HF aspects	<ul style="list-style-type: none"> • Stability of the system improve experience of HF experts during experiments and of users during deployment. • Reactive and adaptive systems are multi-thread applications, proper adaptation requires stable interaction of threads in application. 	

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

Benefits	<ol style="list-style-type: none"> 1. Testing of thread-safety can be done from early state of application and continuously to its finalization 2. Prior experience of a problem is not needed 3. Problematic area is determined narrowly 4. Likelihood of error occurrence is increased, thus reducing time and resources needed to determine the problem 5. Platform independent solution.
----------	---

4.3 Missed event detector (MED) and AdCoS

MED is a tool that analyses video recording to detect the operator's face and gaze direction. The tool comprises of hardware – the camera, and software – the face and gaze detection module and mitigation module.

The detection algorithms use a novel approach based on random forests. The random forests are trained on Annotated Facial Landmarks in the Wild (AFLW) dataset. In this dataset, faces are detected using frontal and profile face detector and the detections are crosschecked with the head position annotations present in the AFLW dataset. Gabor features are extracted from the face images and provided to the random forest training algorithm.

4.3.1 The interfaces

The tool does not require a data input, but it needs configuration for a specific environment. During the procedure, the algorithm is semi-automatically adjusted to pre-defined regions of interest (ROIs) – the operator is asked to look at a specific area, while algorithm connects the observed spatial orientation of head to a defined region. Alternatively, a geometry of the environment can be provided to automate the configuration process. Once configured, the tool can work autonomously as long as the geometry or ROIs do not change.

MED outputs the identifier of ROI that the operator is looking at. The identifier is broadcasted in real time in a preselected rate with minimum of 1Hz implied by time needed to reliable process the video data. The data format is based on a simple UDP protocol, but a future extension for an OSLC message is possible.

4.3.2 The implementation

For Diversion assistant use-case, the cockpit consists of three ROIs – EFB area aside the pilot, the primary flight display in front of the pilot and the outer world containing all other areas, see Figure 8.

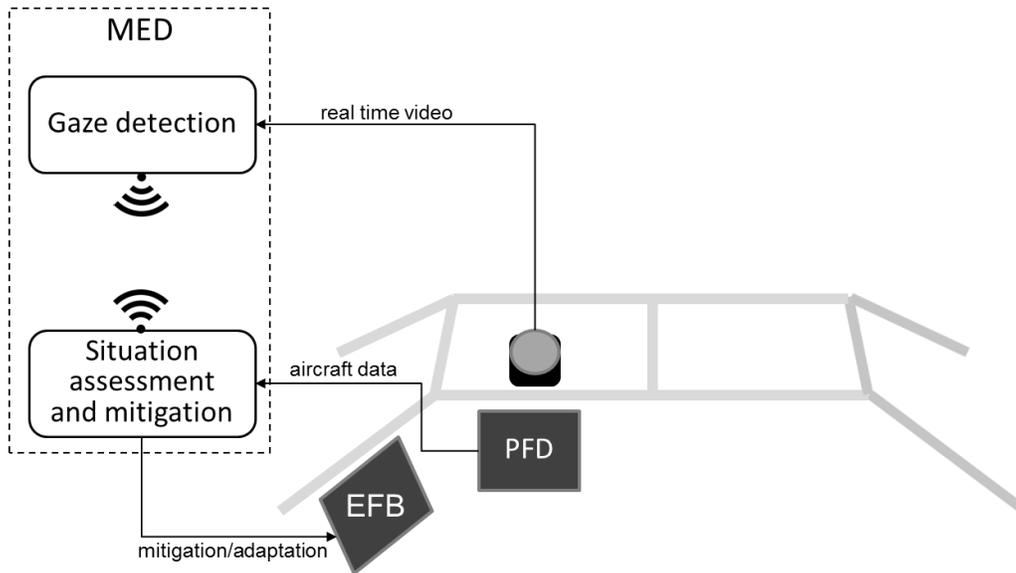


Figure 8: Definition of ROIs and data flow in application of MED to diversion assistant use-case

The diversion assistant is an EFB application that the pilot uses to create an accurate situation model about airports around the actual aircraft location. The pilot can use the application anytime during flight and in the event of diversion. At the same time he is supposed to aviate the aircraft as his primary task and to do so he is obliged to monitor avionics displays, especially the primary flight display.

As EFB applications always have low importance, MED provides means to detect inadequate use, i.e. time when the pilot is using the EFB device while his attention is required at the avionics display, see Figure 8.

In the current status, MED identifies ROI at which pilot looks and provides the information to the mitigation module. The mitigation module applies aircraft data to a list of predefined procedures, in which the pilot must monitor avionics displays. If MED detects pilot using EFB while his attention

is required at avionics, the diversion assistant informs pilot about the conflict. If pilot continues using EFB device, the diversion assistant escalates the warning by fading the screen of EFB, see Figure 9.



Figure 9: Escalation steps when MED detects conflict between required attention and real attention of a pilot. Left – no conflict, no interference. Center – conflict detected, yellow message informs pilot what requires his attention. Right – message ignored, EFB is temporarily disabled.

4.3.3 MED aspects on the development process

Description	MED provides information about the head orientation and thus about where the operator’s focus is directed. The information can be used to detect and mitigate potentially risky situations when pilot is unaware of an important information or event raised outside his primary field of view.	
Process	<p>Original</p> <p>This is a novel use-case. Currently pilots are trained to know when it is required to monitor avionics. Failing to the regular or monitoring is not detected and may lead to incidents/accidents, e.g. in Air Safety Group report Nr. 104.</p>	<p>HoliDes update</p> <ol style="list-style-type: none"> 0. Configure MED for a particular cockpit 1. MED acquires and analyses video stream to detect head orientation. 2. Mitigation module assess cockpit situation based on aircraft data and procedure models. 3. If conflict is detected, pilot is informed about the conflict. 4. MED interprets video stream to understand whether the pilot accepted or ignored the warning 5. If the pilot ignores the conflict,

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

		the warning is escalated eventually leading to EFB being temporarily disabled.
HF aspects	<ul style="list-style-type: none"> • Unobtrusive detection of pilot attention in the cockpit. • Enabler for meaningful adaptation of the system that is acceptable and comprehensive for pilots. • Video data can be used in the future to infer mental states of a pilot – sleep, drowsiness, stress etc. 	
Benefits	<ol style="list-style-type: none"> 1. Detection of missed event improves safety 2. Connection with procedure/ROI to be monitored improves situation awareness 3. Unobtrusiveness allows for long-term monitoring. 	

4.4 Pattern classifier and AdCoS

The goal of the tool is to wrap the pattern classifier model that interprets EEG and eye-tracking signal recorded on a human operator (pilot). The tool is an executable application that reads an input file from a specific folder and writes the output file with its results in the same folder. This output will be exposed to be used by an application that can use the state information about workload or fatigue to investigate the human operator state during an experimental scenario.

As future extension, the tool will read directly the acquired data and interpret them with respect to workload/fatigue in real time. The state information will be provided to a consumer (AdCoS) to trigger adaptation.

4.4.1 The interfaces

The **input** file with the EEG and eye-tracker data is stored in a CSV file, which has 31 columns (features for the model) for powers of alpha and theta bands recorded from a selection of standard channels, green in Figure 10, and for metrics derived from the eye-tracker data, such as percentage of eye closure. An extract from the data file is shown in Figure 11.



HoliDes

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

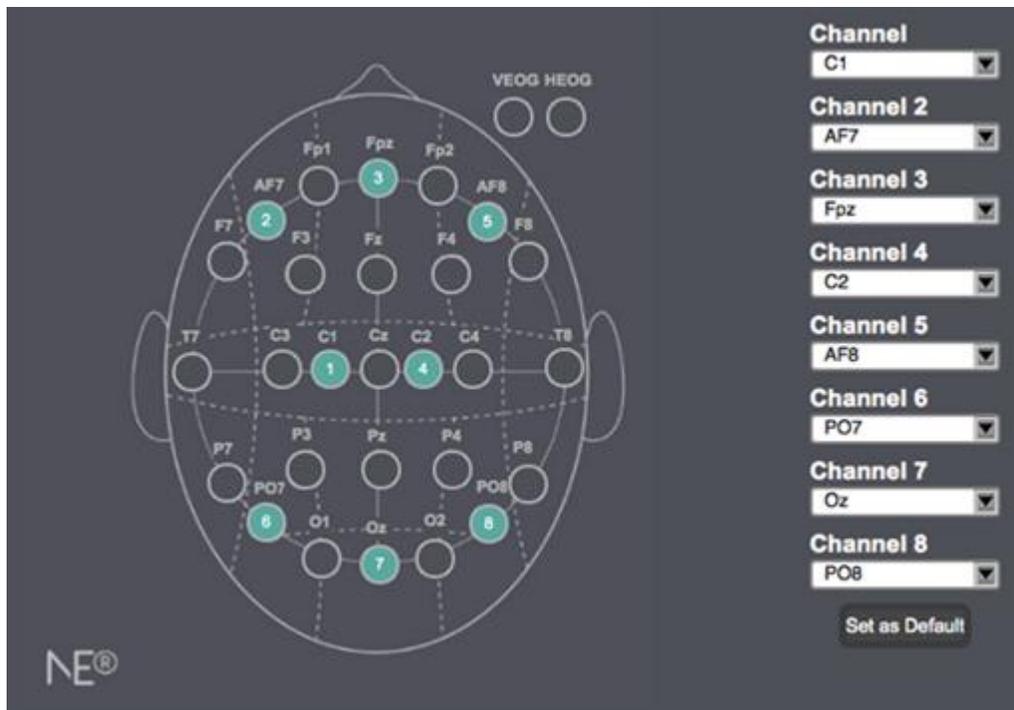


Figure 10: Selection of channels for workload/fatigue detection from EEG signal.



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



timestamp	alpha_P3	alpha_Fz	alpha_F4	theta_Pz	theta_Fz	theta_F4	r_eye_clos	r_pup_dmt	label
47983037	0.2218	0.0180	0.0190	0.2028	0.0166	0.0151	0.117001652718	0.00335993943736	LW
47999017	0.2218	0.0180	0.0190	0.2028	0.0166	0.0151	0.116870284081	0.00339571130462	LW
48015017	0.2218	0.0180	0.0190	0.2028	0.0166	0.0151	0.116870284081	0.00339571130462	LW
48031037	0.2218	0.0180	0.0190	0.2028	0.0166	0.0151	0.125756502151	0.0034096322488	LW
48046571	0.2218	0.0180	0.0190	0.2028	0.0166	0.0151	0.127158224583	0.00340204918757	LW
48062595	0.2218	0.0180	0.0190	0.2028	0.0166	0.0151	0.131465911865	0.00339202117175	LW
48078571	0.2218	0.0180	0.0190	0.2028	0.0166	0.0151	0.122940540314	0.00340295466594	LW
48094571	0.2218	0.0180	0.0190	0.2028	0.0166	0.0151	0.122829079628	0.00340377003886	LW
48111065	0.2218	0.0180	0.0190	0.2028	0.0166	0.0151	0.111488819122	0.00340534537099	LW
48126571	0.2028	0.0168	0.0188	0.1804	0.0151	0.0133	0.170476078987	0.00339816068299	LW
48142595	0.2028	0.0168	0.0188	0.1804	0.0151	0.0133	0.308652281761	0.00340801058337	LW
48158571	0.2028	0.0168	0.0188	0.1804	0.0151	0.0133	0.398672521114	0.00340803479776	LW
48174571	0.2028	0.0168	0.0188	0.1804	0.0151	0.0133	0.519093751907	0.0034053849522	LW
48190595	0.2028	0.0168	0.0188	0.1804	0.0151	0.0133	0.608943223953	0.00341672007926	LW
48206570	0.2028	0.0168	0.0188	0.1804	0.0151	0.0133	0.608943223953	0.00341672007926	LW
48222570	0.2028	0.0168	0.0188	0.1804	0.0151	0.0133	0.468381881714	0.00341986794956	LW
48238573	0.2028	0.0168	0.0188	0.1804	0.0151	0.0133	0.327937602997	0.00342665566131	LW
48254596	0.2028	0.0168	0.0188	0.1804	0.0151	0.0133	0.209520816803	0.00340537750162	LW
48270596	0.2028	0.0168	0.0188	0.1804	0.0151	0.0133	0.382279455662	0.00338092702441	LW
48286573	0.2028	0.0168	0.0188	0.1804	0.0151	0.0133	0.0000	0.0000	LW
48303021	0.2028	0.0168	0.0188	0.1804	0.0151	0.0133	0.704870283604	0.0000	LW
48319002	0.2028	0.0168	0.0188	0.1804	0.0151	0.0133	0.737061321735	0.0000	LW
48335003	0.2028	0.0168	0.0188	0.1804	0.0151	0.0133	0.836940526962	0.0000	LW
48351050	0.2028	0.0168	0.0188	0.1804	0.0151	0.0133	0.836923539639	0.0000	LW
48367012	0.2028	0.0168	0.0188	0.1804	0.0151	0.0133	0.0000	0.0000	LW
48383003	0.2028	0.0168	0.0188	0.1804	0.0151	0.0133	0.638828277588	0.0000	LW

Figure 11: Format of input data expected by pattern classifier (extract example).

The **output** state information is provided in form of an XML file containing the classification – workload/fatigue of a pilot and estimated accuracy of the prediction for each timestamp. In the near future (mid May 2016) the output format will be re-phrased to OSLC for smoother integration in HF-RTP.

4.4.2 The implementation

The tool is a simple executable file that can be placed in any location on the computer. The user will double-click on it, and the input and output file should be located in the same folder than the executable file. The main scheme of the integration and usage process is shown in Figure 12.

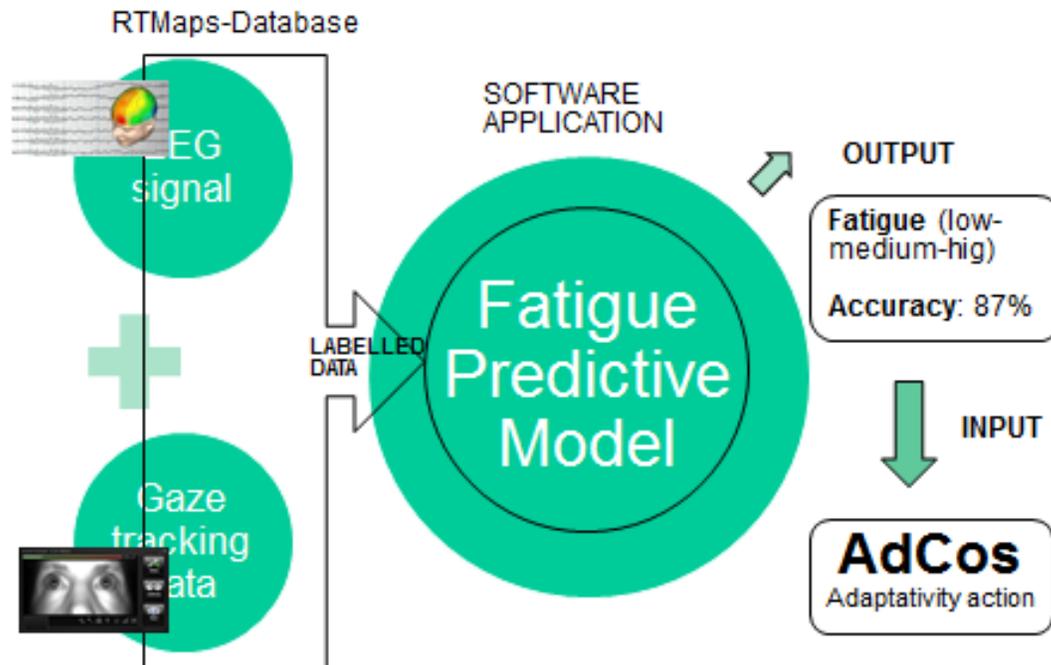


Figure 12: Implementation status for pattern classifier.

The statistical distribution of the brain-computer interfaces (BCIs) data varies across subjects as well as across sessions within individual subjects, limiting the transferability of trained models among them, and we have found this limitation in our particular case in HoliDes. This issue is known as **Transfer Learning**. In the context of BCI, transfer learning is of critical importance (it is long known that the EEG signal is **non-stationary**).

In order to deal with this issue, we have tried to find a shared structure in users' data to develop a global model that can be applied to all users.

To build a classification model, datasets from 21 users were used in supervised multiclass classification for classes of low workload (LW), medium workload (MW), and high workload (HW). The datasets were imbalanced, therefore specific metrics were applied: precision, recall and f1-score.

The model is based on the **KNN** technique and has been developed in **Python**, using some machine learning libraries such as **sklearn**, and some other scientific libraries such as **numpy**, **pandas**, **scipy**. There were 75% of

the data sets used for training and 25% for testing, using **stratified cross-validation**.

As a result, it was not possible to find a global model to be applied to any user. Instead, it is required to develop subject-specific models for each one in order to obtain good ratios. In the future it will be necessary to apply some special filters to the raw data, trying to find a shared structure that enables a model to be applied to any user out of the experiment with satisfactory ratios.

4.4.3 Pattern classifier aspects on the development process

Description	Pattern classifier interprets physiological data with respect to psychological state. The model can provide online information about operator's state that can trigger adaptation of the system or mitigation with respect to correct the operator's state.	
Process	<p>Original</p> <p>The state of the pilot is not monitored. The adaption is not supported.</p>	<p>HoliDes update – a new process</p> <ol style="list-style-type: none"> 1. Create adapter for particular recording devices to provide required data format 2. Connect adapter with pattern classifier 3. Define adaptation or mitigation in the system 4. Connect system with the pattern classifier
HF aspects	<ul style="list-style-type: none"> • Operator mental state determined • Enabler for state-driven adaptation and for state mitigation strategies 	
Benefits	<ol style="list-style-type: none"> 1. Autonomous state detection using commonly available recording devices 2. Complex classification models inside the tool 3. Simple data interface on input and output of the tool 	

4.5 Cognitive Distraction Classifier (CDC)

The Cognitive Distraction Classifier (CDC) is a tool that aims to detect cognitive distraction from an operator. Originally it is being developed for the automotive area, but transferability towards other domains, such as Aeronautics, have been discussed from the beginning.



The CDC should measure, process, and classify several types of data near-to-real-time during operation (e.g., driving). These types of data include, but are not limited to, facial video data and behavioral data. For first application of CDC in Aeronautics, the focus is on facial video data only, as the characteristics in this type is expected to be most similar between the two domains. Behavioral data, for example, is more complicated to compare directly, as the tasks involved in driving and flying vary greatly. Additionally, early experiments indicated that classification accuracy may mostly depend on facial video data. Here the CDC will further be described from the Aeronautics perspective.

4.5.1 The interfaces

Two standard web cameras are needed for recording the operator's face with a frame rate of 30 frames/s (RGB24 640x480 - 30 fps). The output is the level of distraction given in a numeric value (0-100). The theoretical maximum output rate equals to the input frame rate.

An RTMaps diagram, see Figure 13, has been created to connect web cameras, face tracking components (embedding the Intraface software²), and the computational model of the distraction classifier.

² Xiong, X & De la Torre, F., (2013). Supervised Descent Method and its Application to Face Alignment. IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 532-539. doi: 10.1109/CVPR.2013.75.



HoliDes

Holistic Human Factors Design of Adaptive Cooperative Human-Machine Systems

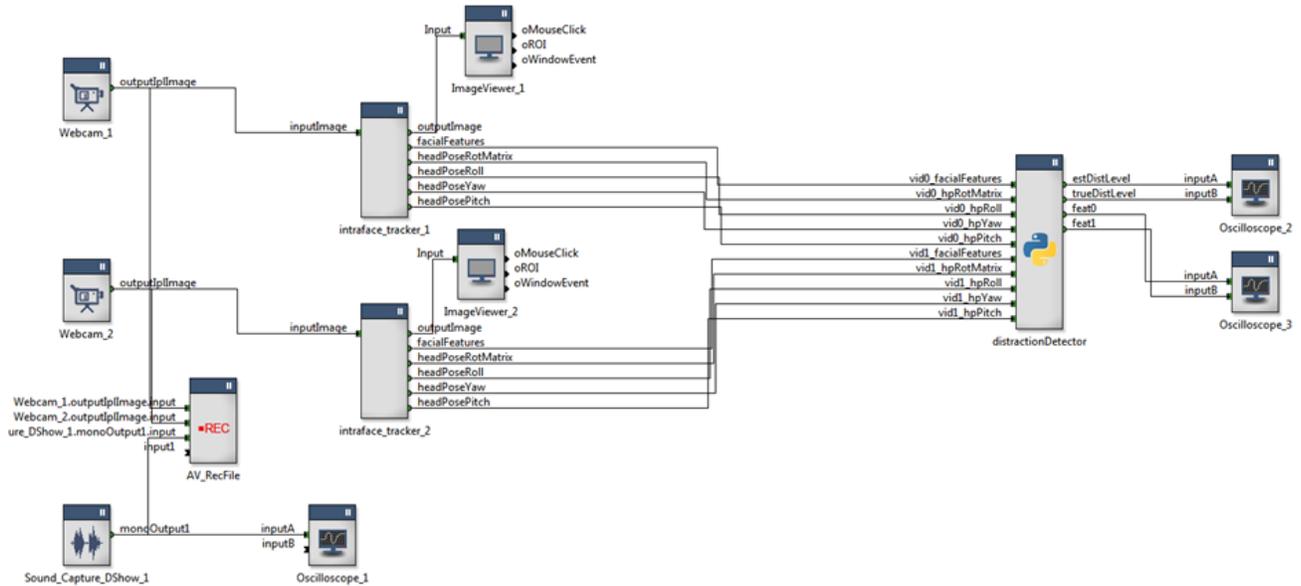


Figure 13: RTMaps diagram for processing video data into distraction features and distraction classification.

The distraction classifier output value (level of distraction, see above) can be sent to other RTMaps components simply by a new connection in the diagram or to other applications outside of RTMaps via TCP.

4.5.2 The implementation

To develop the basis of the CDC, scientific experiments have taken place, in which facial video- and behavioral data have been recorded from participants in a driving simulator. In different conditions the level of cognitive distraction was manipulated by a secondary cognitive task, such as mental calculation. Data were analyzed extensively offline, using a part of the data for training the classification model, and the rest of the data for testing using the trained classification model. The fundamentals for the applied cognitive and computational classification model have been developed. Currently almost all analysis scripts have been transformed for testing in near-to-real-time application.

First facial video data of an operator executing flying-like tasks and additional mental calculating tasks, were provided by Honeywell, and have been analyzed offline by TWT with the CDC V1. An experiment is planned to



collect more data in the cockpit, such that the CDC can be better adapted to the Aeronautics Domain, with its specific challenges, tasks, and characteristics.

4.5.3 CDC aspects on the development process

Description	Cognitive Distraction Classifier interprets video data of a pilot with respect to cognitive distraction. The information about distraction can trigger adaptation of the system or mitigation with respect to decrease need for cognitive resource in interpreting the display when the pilot is being distracted.	
Process	<p>Original</p> <p>The state of the pilot is not monitored. The adaption is not supported.</p>	<p>HoliDes update – a new process</p> <ol style="list-style-type: none"> 1. Arrange camera for optimal data acquisition given the task to be monitored 2. Define adaptation or mitigation in the system 3. Implement relevant adaptation/mitigation
HF aspects	<ul style="list-style-type: none"> • Operator mental state determined • Enabler for state-driven adaptation and for state mitigation strategies 	
Benefits	<ol style="list-style-type: none"> 1. Autonomous distraction detection using commonly available recording devices 2. Possible extension to use the system for detecting other mental states – drowsiness etc. 3. Complex classification models inside the tool 4. Simple data interface on input and output of the tool 	

4.6 RTMaps and AdCoS

RTMaps is a modular environment where data samples flow between functional blocks called components. While some sensors and devices are already supported by RTMaps, support for other devices has to be implemented. The RTMaps Software Development Kit (SDK), included in the RTMaps Developer Edition, enables C++ programmers to develop their own RTMaps components. The SDK was used to develop various components for e.g. flight simulator data accessory and metadata provider.

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

4.6.1 The interfaces

Means of communication between RTMaps and devices are broad. EDA interface is described in section 4.7. However, data from a flight simulator are not accessible via WEB interface, but rather via API. It is up to flight simulator access component to extract data from shared memory (fill up by the flight simulator) and offer them as output to downstream components. Data types and their meaning are deployed within library accessing shared memory in a separate configuration file.

Simplified version of a diagram implementing flight data access is shown in Figure 14.



HoliDes

Holistic Human Factors Design of Adaptive Cooperative Human-Machine Systems

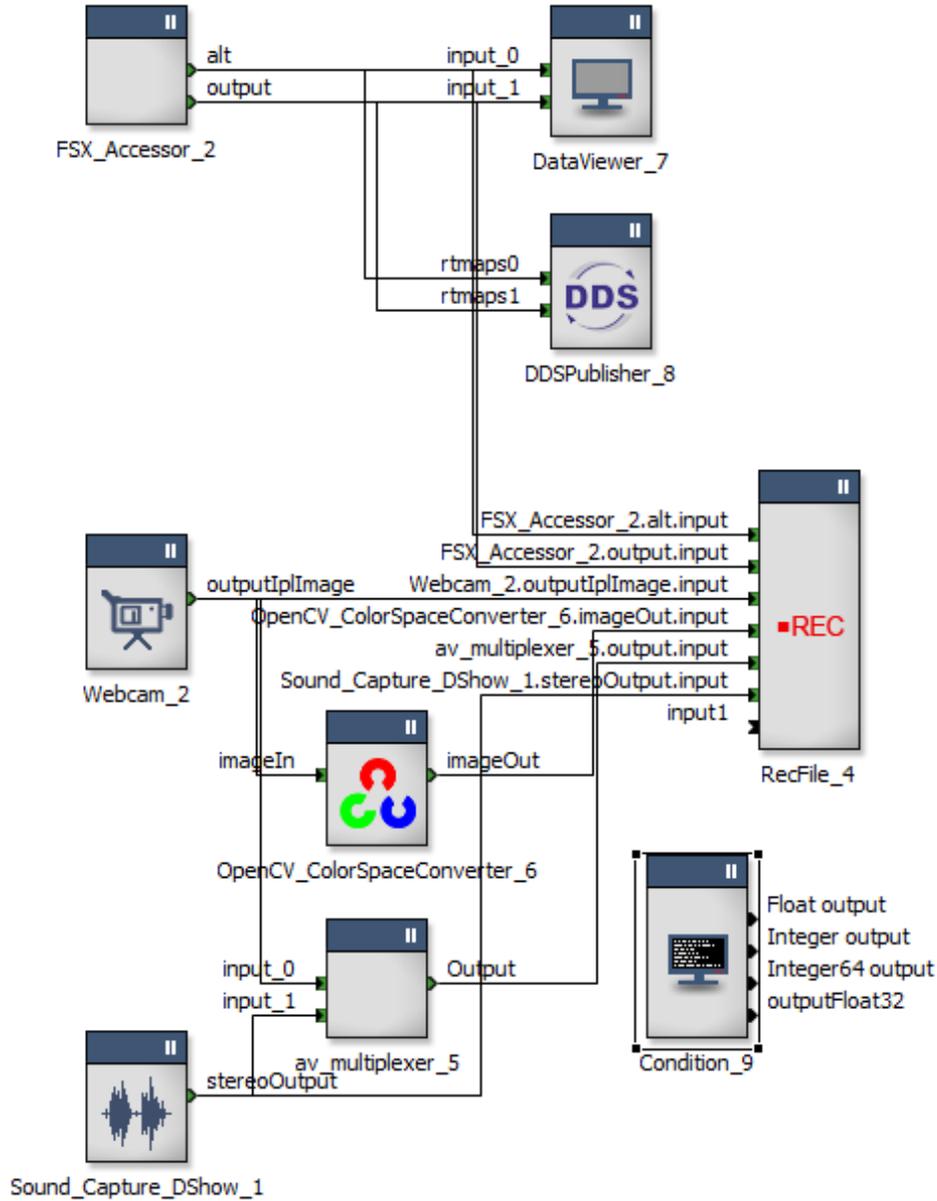


Figure 14: Component scheme for implementing data flow between simulator, video and sound devices and RTMaps.

4.6.2 The implementation

The RTMaps in connection with simulator and other typically used data recording devices were successfully tested to prove the required benefits of RTMaps: data synchronization across a distributed system.

Typically, a flight simulator consists of several computers since a single computer does not have enough computing and/or display resources. These computers are connected to a local area network creating a distributed system.

The following set-up was used to conduct necessary tests: the first computer is used to extract flight data from shared memory using data access component.

The second computer is used to acquire various data, e.g. video of the pilot's face, his/her voice, videos of altimeter, attitude indicator, air speed indicator, etc. Synchronizing the clocks of various RTMaps systems over the network is integrated in RTMaps.

The first computer serves as slave instance of RTMaps and serves two purposes: logging flight data via RTMaps recorder and also feeding data to master instance. The second computer serves as master instance of RTMaps, it is responsible for running and stopping both master and slave instances. It receives flight data from slave instance and also logs them. It separately stores video and audio from connected cameras. It combines audio and video to interleaved audio/video files. Flight simulator component has a special output "Altitude", which can serve as control variable for other components. It allows the user to specify when recording of selected output starts/stops. It can also trigger using a black-and-white filter on selected outputs via condition component.

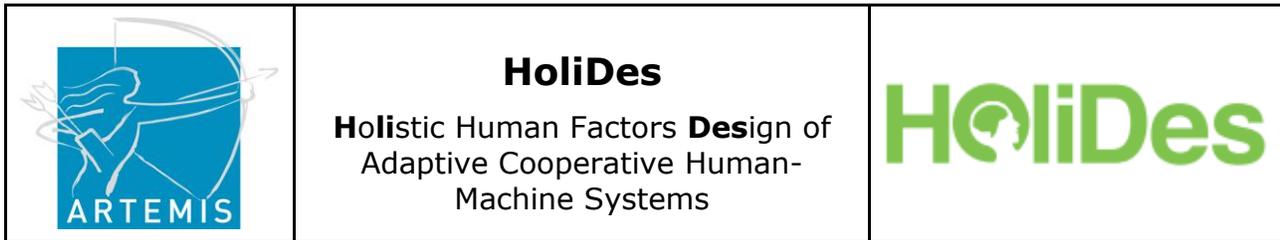
Most of the files from the dataset can be used directly by player component. Exception are e.g. multiplexed files, since while having two inputs, output is a single stream. Such files can be inspected by a third-party tool.

4.6.3 RTMaps aspects on the development process

Description	RTMaps provide a simulation environment that can simplify several HF tasks during the execution of an experiment. These are especially collection of data from a distributed system, synchronization of data from various data sources, data pre-processing or data play-back.	
Process	Original <ol style="list-style-type: none"> 1. Define synchronization event (not always possible, especially for distributed systems). 2. Connect device, start recording and invoke the synchronization event. 3. Verify data acquisition. 4. Conduct experiment. 5. Manually collect data from various computers and sources and place them at one location. 6. Verify or correct synchronization. 	HoliDes update <ol style="list-style-type: none"> 1. Not needed. 2. Connect devices and start recording. 3. Verify data acquisition. 4. Conduct experiment. 5. Not needed. 6. Not needed.
HF aspects	<ul style="list-style-type: none"> • HF experts do not need to spend time on solving synchronization (defining synchronization events or rescaling recorded data files), which is specific for each experiment. • HF experts do not need to spend time on non-expert manual work with collecting data from various computers. 	
Benefits	<ol style="list-style-type: none"> 1. Automatic data acquisition from various sources distributed on several computers. Automatic data organization to keep all data files together. 2. Automatic time synchronization of all data sets. 3. Simple inspection of data integrity during the course of experiment (all data can be visualize at one screen) 	

4.7 Experiment Data Archive (EDA) and RTMaps

EDA tool is a document management tool designed and developed in Python/Django to allow the definition and storage of multiple projects including their full data structure through a web interface. It provides a web



service and OSLC API. EDA is a centralized archive for HF data acquired in various development phases of a project. EDA stores acquired data together with meta-information that links the data to the project structure.

During the course of a project, there are a number of activities where data from subjects are collected. Based on the phase of the project, data is used for definition of a concept (voice of a customer) and requirements as well as for validation of prototypes.

The tool is designed to store the projects' structure and the data generated when confronting these projects with subjects in a centralized archive.

4.7.1 The interfaces

The EDA tool uses as **input** description data and meta-data about various projects and their HF related data collected in experiments, interviews etc. The tool provides a web interface to specify project data and JSON/OSLC based communication to simulation platform (such as RTMaps) to automatically collect experiment meta-data.

To define a project different general data should be entered:

- Description (plain text)
- Start date / End Date (date type)
- Team leader, HF Focal & SW leader (plain text)
- Description (plain text)

Each project executes a number of phases. Information about category (plain text) with different values as requirements, design, implementation and evaluation is entered. Within a phase there is a number of data collection sessions, which are associated to a project and a phase, and as data required have:

- Start (date)
- End (date)
- Observations (structure)
- Interviews (structure)
- Questionnaires (structure)
- Experiments (structure)

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

Each *session* has a start and end date and contains two files – the data collection plan describing what and how it will be done and the report/summary file describing the results of analysis of the acquired data. It also requires following information:

- Location where the session is being performed
- Who is responsible for the collection
- Who is responsible for the analysis

Other information stored in this EDA tool is about the subject doing the experiments. In this case, the stored information will be:

- Email (text with format email)
- Name (plain text)
- Last name (plain text)
- Year of birth (date)
- Gender (female/male)
- Rank (plain text)
- Flight hours (number)
- Aircraft flown most often (plain text)

The full EDA structure is described in Figure 15.



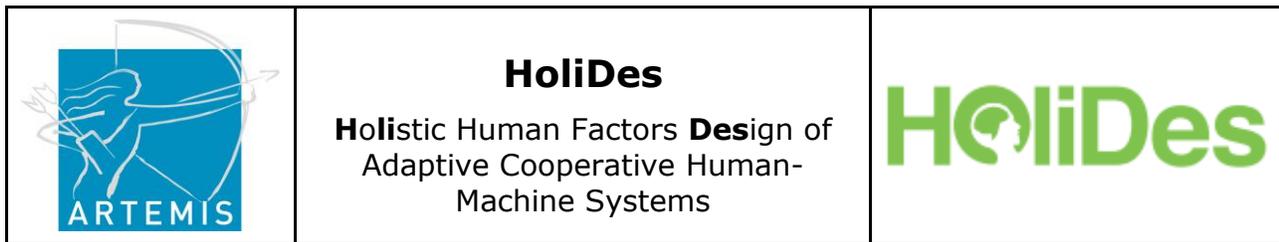
HoliDes

Holistic Human Factors Design of Adaptive Cooperative Human-Machine Systems



```
{
  "end": "31/10/2016",
  "hf_focal": "zdenek.moravek@honeywell.com",
  "id": 2,
  "keywords": [
    "GUI design",
    "Diversion"
  ],
  "name": "HoliDes",
  "phases": [
    {
      "artifacts": "",
      "category": "Requirements",
      "id": 2,
      "sessions": [
        {
          "analysis_responsible": "",
          "collection_responsible": "",
          "end": "25/11/2015",
          "experiment": "False",
          "id": 4,
          "interview": "False",
          "keywords": [
            "GUI design",
            "Diversion"
          ],
          "location": "None",
          "methods": [
            {
              "author": "sara.sillaurren@tecnalia.com",
              "id": 7,
              "keywords": [
                "GUI design",
                "Diversion"
              ],
              "location": "None",
              "phase": 2,
              "project": 2,
              "result": "",
              "session": 4,
              "title": "Interview to decide the GUI design",
              "type": "IN",
              "type_description": "Interview",
              "web": "False"
            }
          ],
          "observation": "False",
          "plan": "",
          "questionnaire": "False",
          "report": "",
          "start": "25/11/2015"
        }
      ]
    },
    {
      "artifacts": "",
      "category": "Design",
      "id": 3,
      "sessions": []
    },
    {
      "artifacts": "",
      "category": "Development",
      "id": 4,
      "sessions": []
    },
    {
      "artifacts": "",
      "category": "Evaluation",
      "id": 5,
      "sessions": []
    }
  ],
  "sap_id": "EG-1234",
  "start": "01/11/2013",
  "sw_lead": "filip.magula@gmail.com",
  "team_leader": "sara.sillaurren@tecnalia.com"
}
```

Figure 16: JSON message that mediates communication between EDA and a simulation platform.



EDA tool is connected and synchronized with a simulation platform to support automated data collection during experiments. As of HoliDes HF-RTP, the simulation platform is the RTMaps tool. This synchronization is done via web services and allow the following actions:

- Synchronization of the start and end of an experiment
- Saving in EDA of events and metadata of files generated during experiments.

The flow chart in Figure 17 describes the execution of an experiment. The steps involved are the following:

- User selects a task and request its execution through EDA
- EDA communicates via socket with the master instance of RTMaps
- The master instance of RTMaps starts the experiment and communicates with other potential slave instances of RTMaps.
- Optionally the user could finish the experiment from EDA. If so the EDA communicates via socket with the master instance of RTMaps.
- Finally, once the experiment is finished, the master instance of RTMaps sends back the metadata of the generated files and the events that took place during the experiment to EDA.

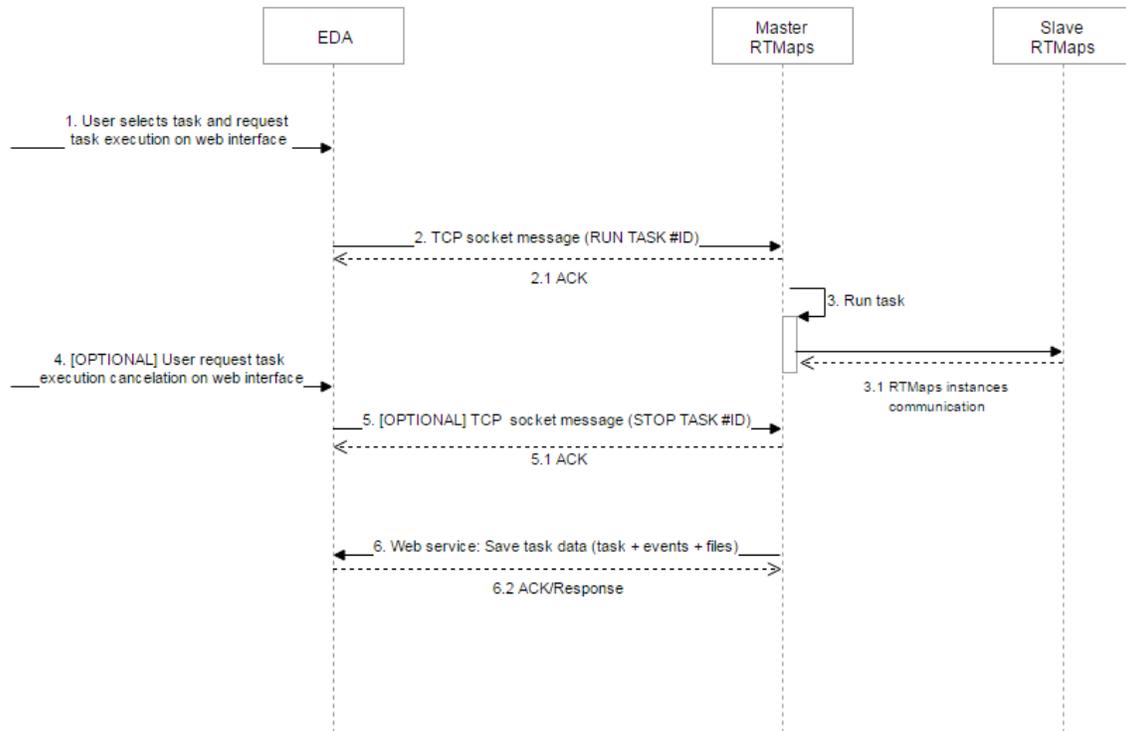


Figure 17: Communication scheme between EDA and RTMaps

OSLC integration

EDA also exposes its contents via OSLC specification (Open Services for Lifecycle Collaboration). Currently the EDA catalog can be accessed to obtain the different projects contained as OSLC service providers in RDF/XML syntax, see Figure 18.



HoliDes

Holistic Human Factors Design of
Adaptive Cooperative Human-
Machine Systems

HoliDes

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <oslc:ServiceProviderCatalog
3     xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
4     xmlns:dc="http://purl.org/dc/terms/"
5     xmlns:oslc="http://open-service.net/ns/core#"
6
7     rdf:about="http://127.0.0.1:8000/eda/oslc/catalog/">
8
9     <dc:title>EDA Service Provider Catalog</dc:title>
10    <dc:description>Public projects managed by the Experiment Data Archive tool</dc:description>
11    <dc:domain>None</dc:domain>
12
13
14    <oslc:entry>
15        <oslc:ServiceProvider>
16            <dc:title>Project 1</dc:title>
17            <oslc:service rdf:resource="http://127.0.0.1:8000/eda/oslc/catalog/project/5/" />
18        </oslc:ServiceProvider>
19    </oslc:entry>
20
21    <oslc:entry>
22        <oslc:ServiceProvider>
23            <dc:title>Project 2</dc:title>
24            <oslc:service rdf:resource="http://127.0.0.1:8000/eda/oslc/catalog/project/6/" />
25        </oslc:ServiceProvider>
26    </oslc:entry>
27
28 </oslc:ServiceProviderCatalog>
29
```

Figure 18: Structure of OSCL message used to mediate communication with EDA

OSLC also defines an alternative integration tool known as delegated dialog integration. Instead of the tool that is using the EDA contents build the interfaces, this way EDA would return the interfaces that allow the user to interact with the contents. This is expected to be developed in EDA in the following weeks to list and select a project from the OSLC catalog.

The diagram from the open-services.net website describes this integration technique.



HoliDes

Holistic Human Factors Design of Adaptive Cooperative Human-Machine Systems

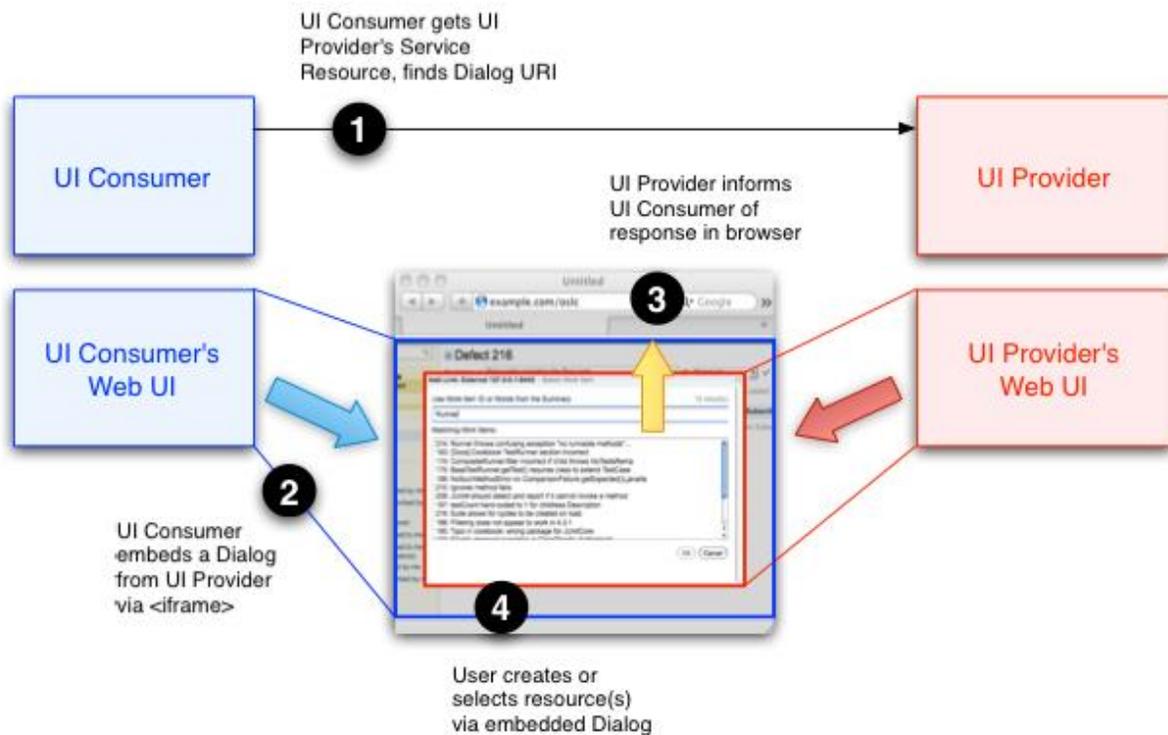


Figure 19: Scheme for integration of EDA

4.7.2 The implementation

First of all the user must authenticate himself in order to use the EDA tool. There has been developed a permissions management module, each user will only see the public projects and the ones assigned to him via users groups.

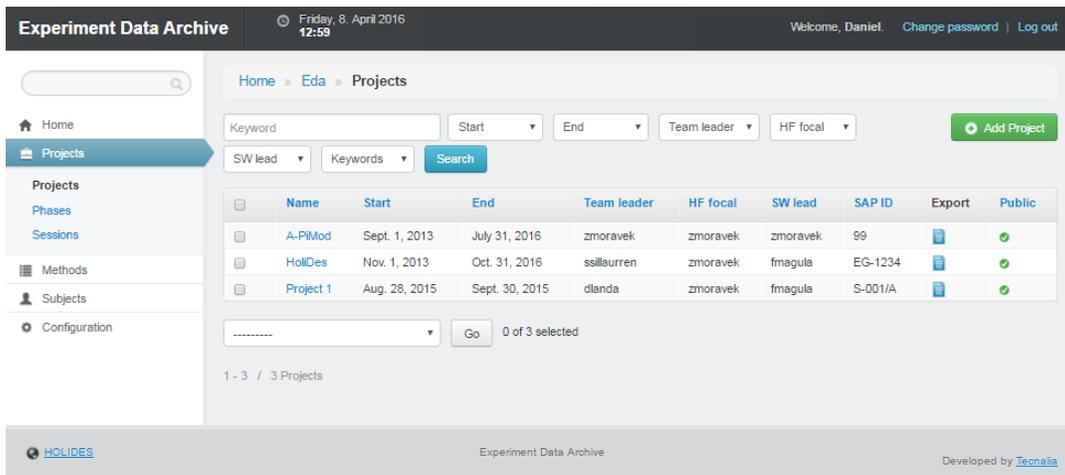


Figure 20: EDA - main window interface

Navigating from the left panel menu, Figure 20, the user can access the different modules that compose EDA. The main categories are:

- **Projects:** Projects defined in the tool, each of them with its structure of phases, sessions and methods. These subcategories can be accessed from within a project (seeing only the ones contained in the current project) or from direct links to all Phases or Sessions, see Figure 21.



HoliDes

Holistic Human Factors Design of Adaptive Cooperative Human-Machine Systems



Experiment Data Archive | Friday, 8. April 2016 13:00 | Welcome, Daniel. Change password | Log out

Home > Eda > Projects > Add Project

General | Methods | Keywords | Files | Missing fields

Permissions

ID: (None)

Name: *

Start: Today

End: Today

Team leader: [dropdown] [edit] [delete]

HF focal: [dropdown] [edit] [delete]

SW lead: [dropdown] [edit] [delete]

SAP ID:

Export: [icon]

Description:

Phases

Category	Artifacts	Delete?
Add another Phase		

Sessions

Edit	Phase	Start	End	Plan	Report	Obs.	Int.	Quest.	Exp.	Delete?
Add another Session										

HOLIDES | Experiment Data Archive | Developed by Tecnalia

Figure 21: Interface to define project properties

- **Methods:** The methods link allows the direct access to all defined methods belonging to any project. Within each of these methods (Observations, Interviews, Questionnaires and Experiments) the user can access and modify its properties and see all the subjects that have participated, see Figure 22.



HoliDes

Holistic Human Factors Design of Adaptive Cooperative Human-Machine Systems



Experiment Data Archive Wednesday, 13. April 2016 12:34 Welcome, Daniel. [Change password](#) | [Log out](#)

Home » Eda » Methods » Interview, Interview to decide the GUI design

General | **Keywords** | Files | Subjects

Project: HoliDes Inherited from the session

Phase: HoliDes, Requirements Inherited from the session

Session: * HoliDes, Requirements, 15/11/25 - [edit] [add]

Title: * Interview to decide the GUI design

Author: sllauren [edit] [add] [x]

Description:

Save

Save and continue editing

Save and add another

Delete

Tools

History

Add Method

Open questions

Edit	Text *	Delete?
Full edit	Which design do you find easier to use?	<input type="checkbox"/>
Full edit	Which design is better for you?	<input type="checkbox"/>

[Add another Open Question](#)

HOLIDES Experiment Data Archive Developed by [Tecnalia](#)

Figure 22: Interface to define a method

- **Subjects:** Access to all participant subjects in any method, and within each subject, access to all the method in which he/she has participated, see Figure 23.

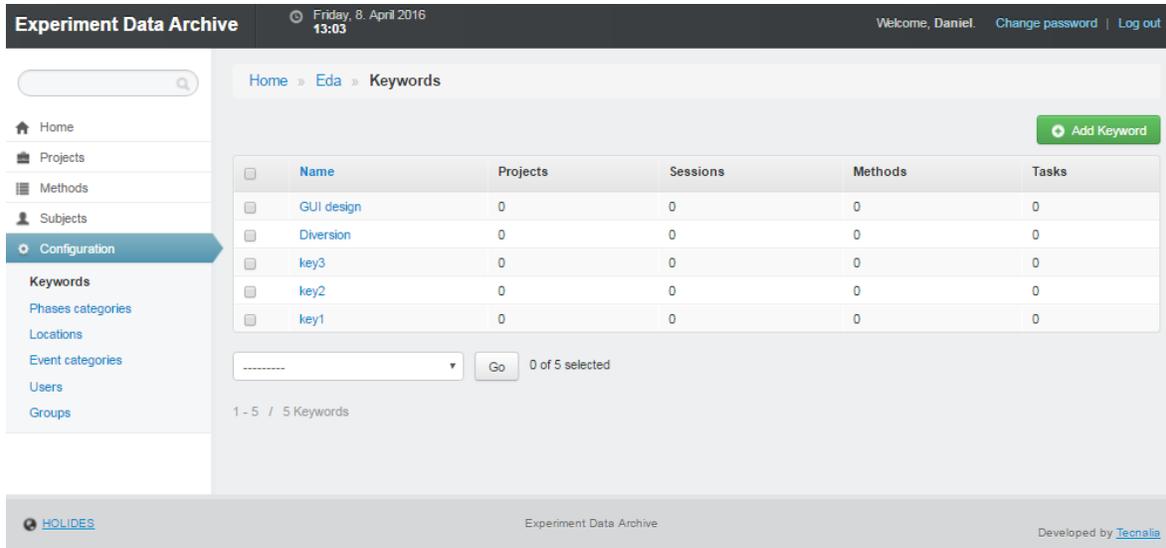
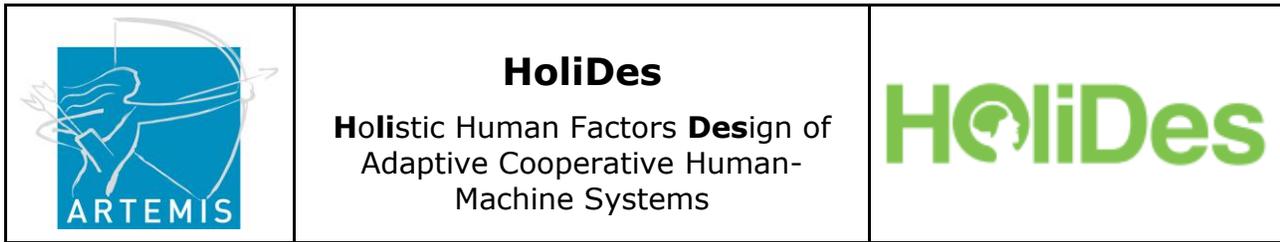


Figure 24: Interface to define meta-data, such as keyword classes, configurations etc.

The EDA tool is available in the following website:
<http://150.241.239.77/admin/eda/project/>

The user gathering information for an experiment, first of all should define a project, with its different metadata. After that, the project phases should be defined. Finally the number of collected sessions must be introduced in the EDA tool.

4.7.3 EDA aspects on the development process

Description	EDA connected to RTMaps improves the process of data acquisition and archiving. The tool chain assures correct links between meta-data and real data, it supports synchronization between various data sources and provides robust data structure.	
Process	Original <ol style="list-style-type: none"> 1. Set-up experiment and record data 2. Create experiment description including data to be collected and copy it to a predefined folder 3. Collect data from various 	HoliDes update <ol style="list-style-type: none"> 1. Set-up experiment 2. Define new data collection in EDA interface

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

	<p>computers/folders. Organize their names to correspond to the data collection and place them at one location</p> <p>4. Adjust timestamps of data files from different data sources – synchronize manually and rescale the data</p>	<p>3. Run RTMaps recorder to record and automatically synchronize data. EDA-RTMaps collect data and its meta-information and they store in at reserved location.</p>
HF aspects	<ul style="list-style-type: none"> • EDA collects all types of EDA data and keeps them in one place • EDA automates some tasks that HF experts had to do manually – data manipulation, data synchronization etc. • EDA enables advanced data analysis by providing data from multiple projects and experiments. • EDA enables creating online questionnaires and interviews. The data can be collected online. 	
Benefits	<ol style="list-style-type: none"> 1. Data collection is simplified allowing HF experts focus on expert work 2. Managed data with a simple web interface allows for data sharing within a team and among teams 3. Flexible meta-information allows to keep relations among various types of information 4. Look-up times for information related to evaluations are reduced, especially in project hand-over or when working with certification agencies, i.e. providing data recorded in past. 	

5 HF-RTP Instance for EATT

In the EATT use case, the AdCoS, i.e. the adapted “system”, is the training syllabus. The HF-RTP instance for EATT supports both, the functions of the system (i.e. adaptation of a syllabus to previous knowledge of pilots) and the development process of Training Syllabi itself. The HF-RTP instance is supposed to bring the following benefits

- A. **Syllabus Adaptation:** Instead of having a “one-size-fits-all” training for all trainees, which is currently standard in the training industry,

EATT sets the ground for individualized training. Individualized training overcomes several human factor issues. Each trainee has its individual previous experience, which either supports or hinders training. This experience start with individual methods of learning, up to knowledge on previously flown aircrafts, and is influenced by age, social- and cultural background, and the personal career of the trainee. For example, a 21 year old trainee will maybe learn new things faster, but a 50 year old trainee can compensate this with his previous experience. Objective of EATT is adaption to the previous knowledge, as a first step in the direction of individualized training.

- B. Syllabus Specification and Management:** Management and specification of training syllabi is a key success factor for a training company like LFT. Version management of syllabi is as important as specification of the training syllabi with regard to the customer's needs. This is especially true, when several trainers specify the training and other, locally distributed trainers, perform the training. Syllabi need to be updated, not only based on feedback during training, but also when standards or official regulations changes, and it has to be ensured that always the latest version of a syllabus is used during training. EATT is a first step to a tool-supported Syllabus Specification and Management.

The assessment of benefits with respect to the baseline has been described in D1.6 - HF-RTP Version 1.8 incl. Methodology and Requirements Analysis Update & 2nd year Baseline assessment. Figure 2 shows the HoliDes development process of a Syllabus.

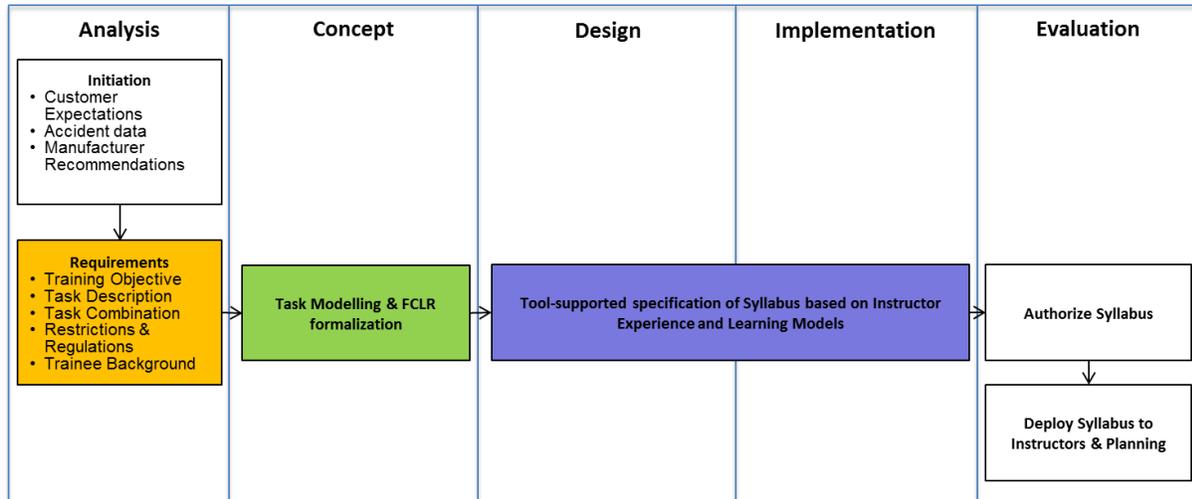


Figure 25: Development process for EATT using the HF-RTP instance.

The particular selection of HF-RTP instance made for EATT improves the development process in the following way

- It simplifies requirement capturing and monitoring for the trainers from customers as well as regulation.
- It provides standard procedure models and learning methods for the development of the AdCoS, allowing individualization (adaptation).
- It improves the performance of trainers while specifying a syllabus.

Additionally, the HF-RTP instance support HF work, by allowing individualization of the syllabus by supporting adaptation to the previous pilot knowledge.

Figure 26 shows the architecture of the HF-RTP instance for the EATT use-case. Central point of the HF-RTP instance is the TrainingManager. The TrainingManager:

- Reads the Standard Operating Procedures (①) for the target and the source Aircraft, compares them and calculates the category of difference.
- Allows specification of the syllabus, by providing some additional information (e.g. simulators, airports, their runways and departure and arrival routes) as well as the Flight Crew Licensing Requirements (FCLRs) (②) from the Database.

- The TrainingManager then saves the Syllabus in the database (③) and produces the Syllabus instance for the executing trainers in form of a Word Report (④).

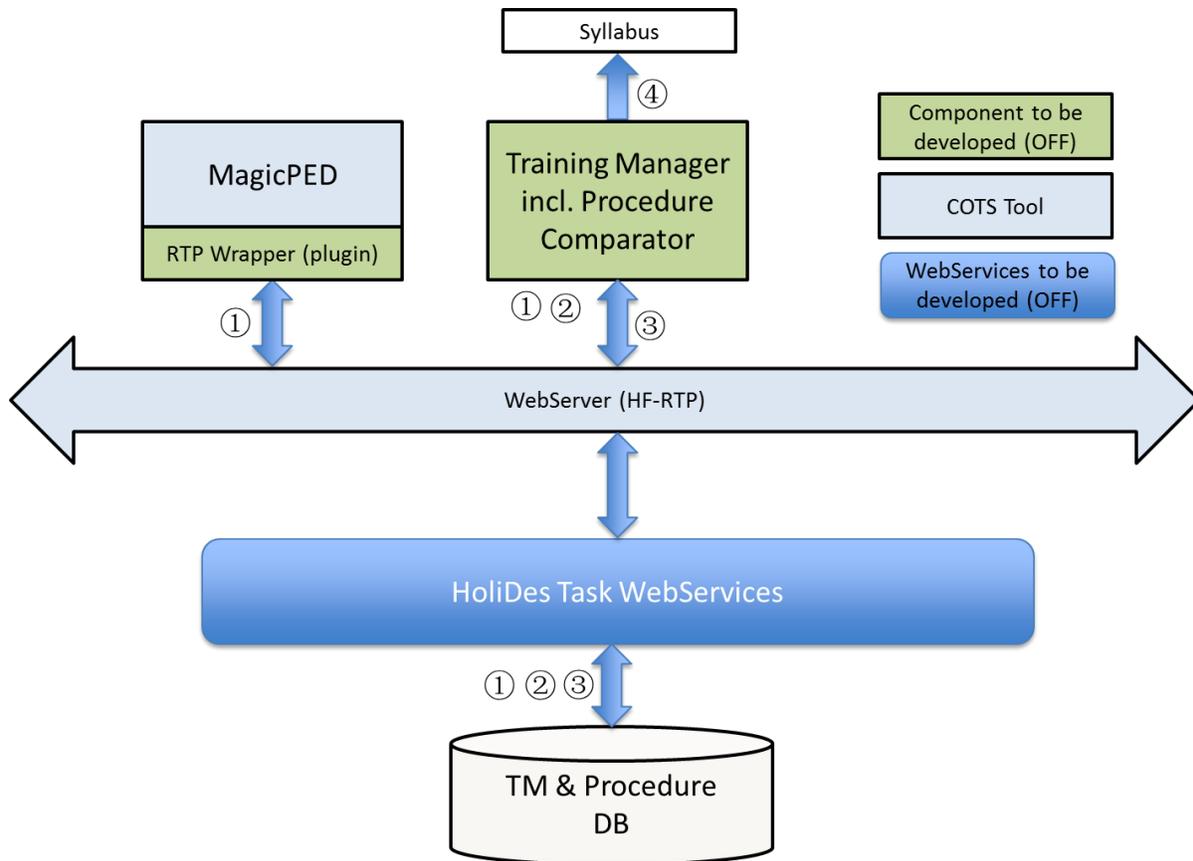


Figure 26: HF-RTP instance for EATT

Details will be given in the following sections with focus on the MTTs MagicPED and the TrainingManager.

5.1 Tailoring Rules

MTT	Lifecycle or Non- Lifecycle	Applied Rules	Tailoring
MagicPED	Lifecycle	1-4	
TrainingManager	Lifecycle	1-4	

The following sections will describe the results of the tailoring rules 3 and 4, as well as the benefits gained by applying the tools.

5.2 MagicPED and AdCoS

For MagicPED there is no direct connection to the AdCoS, but it is a pre-requisite for the TrainingManager. As shown in Figure 26, MagicPED stores the Standard Operating Procedures (SOPs) in the database.

5.2.1 The Interfaces

For MagicPED, an HF-RTP wrapper, in form of an OSLC client has been developed, which stores the SOPs in the database. For this, the OSLC Lyo library has been used. Lyo is a java-based Library that allows to generate the needed ServiceProviders and Services, and allows implementation of clients, too.

The following table shows the database columns specified for storing the Task Models. As the task models are still in revision, and for easier access of the model, for now we decided to store the model as XMI (from ecore) in the database (which is a serialisation of the Common Meta Model).

Column Name	Type	Comment
id	int(11)	Identifier of the task model
airline	int(11)	Primary key of the airline, to which this specific task model belongs
aircraft	int(11)	Primary key of the aircraft type, for which this task model belongs to.
version	timestamp	Timestamp of the current (mapped to "dcterms:modified" of the OSLC spec.
description	varchar(1000)	A textual description of the model, e.g. comment on latest change; mapped to "dcterms:description"
model	longtext	Serialised Task model (from ecore model of HoliDes task model)

Annex I shows the ResourceShape in RDF, created by Lyo, for the Task Models.

5.2.2 The implementation

MagicPED consists of the commercial UML tool MagicDraw and some plugins developed by OFFIS, providing a UML-Profile for a customized Language. This has been described in more detail in Deliverable D2.4, section 3.1. MagicPED has been used for the specification of the A320 and the B737 SOPs by HoliDes partner TRS, in cooperation with OFF. This activities have been concluded last year, and have been described in D7.5.

5.2.3 MagicPED's aspects on the development process

Description	MagicPED allows specification of task models, and has been used in HoliDes for specification of A320 and B737 SOPs. It is used in the concept phase of the syllabus development.	
Process	Original There is no task modelling in the original development process of LFT for a syllabus.	HoliDes update: <ol style="list-style-type: none"> 1. Formalisation of the Systems of an aircraft in MagicPED (Resource Model) 2. Formalisation of the SOPs of an aircraft in MagicPED (Task Model) 3. Storage of the SOP in the SOP database During Lifecycle of the aircraft: <ol style="list-style-type: none"> 4. Update of Resource and Task model due to <ol style="list-style-type: none"> a. Updates of Procedures required by law or by manufacturer b. Adaptation to Airline Specific Procedures 5. Update of the SOP in the database
HF aspects	<ul style="list-style-type: none"> • Adaption of training content to previous knowledge, and with this improving "workload" (learning effort) by not focussing on already known things. 	
Benefits	<ol style="list-style-type: none"> 1. Formalisation of SOPs supports standardisation within the training 2. Allows comparison of SOPs between different aircraft types. 3. Storage of the SOPs in a DataBase, in an IOS compliant matter, such that other tools beside the TrainingManager can understand the model, i.e. linking the models to requirements, or for using a 	

	Change Management tool for SOPs would be possible.
--	--

5.3 TrainingManager and AdCoS

This section describes the integration between the TrainingManager and the AdCoS (i.e. the adapted Syllabus). As shown in Figure 26, the TrainingManager retrieves the Standard Operating Procedures (SOPs) from the database, as well as the flight crew licensing requirements (FCLRs).

5.3.1 The Interfaces

Similar to MagicPED, also the TrainingManager implements a Lyo client to connect to the Webservice for the Training AdCoS. The Task Model is retrieved in the same format, as described in section 5.2.1. In addition to that, the TrainingModel as described in D2.6 section 2.1.6, is retrieved from the Webservice.

The following table shows the database information for the Training Model:

Column Name	Type	Comment
id	int(11)	Identifier of the training model
aircraft	int(11)	Primary key of the aircraft type, for which this training model belongs to.
airline	int(11)	Primary key of the airline, to which this specific training model belongs
version	timestamp	Timestamp of the current (mapped to "dcterms:modified" of the OSLC spec.
description	varchar(1000)	A textual description of the model, e.g. comment on latest change; mapped to "dcterms:description"
model	longtext	Serialized Training Model (from ecore model of HoliDes training model)

Similar to the task model, we decided (for now) to store the actual model in XMI format in the database, due to its changing format and for easier and

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

faster access via the WebService. Annex II shows the ResourceShape of the TrainingModel in RDF.

The other information retrieved by the TrainingManager are domain specific information (like airport information, runways, routes), and will not be described in this deliverable.

As output the TrainingManager produces a word report as syllabus. A syllabus contains for each lesson a) a graphical overview on the flight, b) a list of learning content in this lesson, and c) detailed time tables for the flights, partitioned into legs³. Annex III shows a Syllabus generated by the TrainingManager for the 3rd round of evaluation.

³ Each leg is associated with a trainee who is the pilot flying in this leg. This role of pilot flying is then changed to the other trainee in the next leg.



HoliDes

Holistic Human Factors Design of Adaptive Cooperative Human-Machine Systems



5.3.2 The implementation

The TrainingManager is a Stand-Along Java application, which can be installed in any location on the computer. For windows a JDK is included in the provided archive. A JDK (instead of an JRE) is needed, because the library used for creating the report needs a compiler for compiling java classes at runtime.

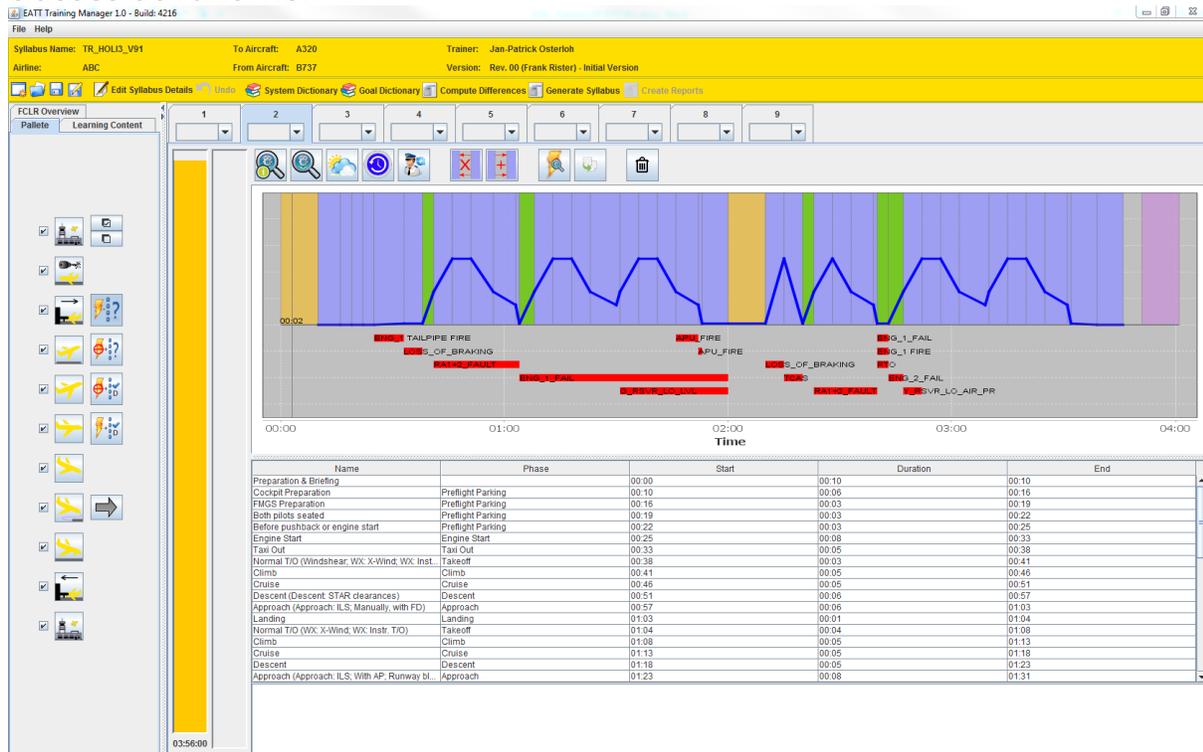


Figure 27: The TrainingManager's Main Window

Figure 27 shows the main window of the TrainingManager, with the syllabus in Annex III loaded. New in this version is (in comparison to the description in D2.6), that the two step approach (1. assignment of content to lesson, 2. detailed planning on timeline) has been reduced to one step, i.e. the content is directly added from the FCLR tree (Figure 28) to the timeline in one step. In addition to that, the palette, that allows an easier access to the content has been added, bringing a great improvement in the time needed for a lesson specification. Another new feature is the FCLR coverage overview as shown in Figure 29. The tree shows which FCLR is already covered by which trainee, i.e. the content has been at least trained in one lesson.



HoliDes

Holistic Human Factors Design of Adaptive Cooperative Human-Machine Systems



The FCLR tree (Figure 28) shows also the calculated category of differences, which can be used by the trainer when specifying the syllabus.

Element	FCLR	Use...	Rating	Learning Method
Complete Flight				
preflight_parking				
Cockpit Preparation				
PRELIMINARY_COCKPIT_PREPARATION	1.1.3	9	0.97	5 - Procedures ...
COCKPIT_PREPARATION	1.1.3	9	0.99	5 - Procedures ...
FMGS Preparation				
COCKPIT_PREPARATION_FMGS	1.1.3	9	1.00	6 - New Proced...
Both pilots seated				
BOTH_PILOTS_SEATED	1.1.4, 1.1.4, ...	9	0.99	5 - Procedures ...
Before pushback or engine start				
BEFORE_PUSHBACK_OR_START	1.1.4	9	0.93	6 - New Proced...
engine_start				
Engine Start				
STARTING_PROCEDURE	1.1.4	9	0.78	6 - New Proced...
AFTER_START	1.1.4	9	0.99	5 - Procedures ...
Engine Start Malfunctions (min: 3)				
taxi_out				
Taxi Out				
TAXI	1.1.5	14	0.96	6 - New Proced...
ATC_CLEARANCE_OBTAINED	1.1.6	14	0.98	6 - New Proced...
CABIN_READY_RECEIVED	1.1.6	14	1.00	6 - New Proced...
NAVIGATE	3.3.3	14	0.99	5 - Procedures ...
Taxi Out Malfunctions (min: 1)				
takeoff				
Normal T/O				
BEFORE_T/O	1.1.6	35	0.69	5 - Procedures ...
INITIAL_T/O	2.2.1	35	0.35	6 - New Proced...
T/O	2.2.1	48	0.42	6 - New Proced...
AVIATE	3.3.9.1, 3.3.9.1	48	0.30	5 - Procedures ...
MTOW (min: 0)				
T/O at MTOW				
Windshear, ENG Failure, RTO or Crew Men				
windshear				
Engine Failure (min: 0)				
T/O simulated engine failure shortly				
ENG_1_FIRE	2.2.5.1	5	0.15	5 - Procedures ...
ENG_1_FAIL	2.2.5.1	10	0.39	5 - Procedures ...
ENG_1_REVERSER_FAULT	2.2.5.1	0	0.27	5 - Procedures ...
ENG_1_REVERSE_UNLOCKED	2.2.5.1	0		
ENG_1_REVERSER_PRESSURI2	2.2.5.1	0		
ENG_1_FADEC_FAULT	2.2.5.1	0	0.10	5 - Procedures ...
ENG_1_LO_OIL_PR	2.2.5.1	0		
ENG_2_FIRE	2.2.5.1	6	0.17	5 - Procedures ...
ENG_2_FAIL	2.2.5.1	11	0.44	5 - Procedures ...
ENG_2_REVERSER_FAULT	2.2.5.1	0	0.27	5 - Procedures ...
ENG_2_REVERSE_UNLOCKED	2.2.5.1	0		
ENG_2_REVERSER_PRESSURI2	2.2.5.1	0		
ENG_2_FADEC_FAULT	2.2.5.1	0	0.10	5 - Procedures ...

Figure 28: FCLR Tree

Element	Trainee 1	Trainee 2
1.1 - FLIGHT PREPARATION		
1.1.3 - Cockpit Inspection		
1.1.4 - Use of Checklists prior Start		
1.1.4 - Starting Procedures		
1.1.4 - Radio/Nav Equipment Check		
1.1.4 - Self/setting of NAV/COM Frequencies		
1.1.4 - After Start Procedure		
1.1.5 - Taxiing in compliance with instruction from instructor		
1.1.6 - Before Takeoff Checks		
2.2 - TAKE OFFs		
2.2.1 - Normal Take-offs with different flap settings incl. expedite		
2.2.2 - Instrument T/O, transition to instrument flt required during		
2.2.3 - X-Wind T/O		
2.2.4 - T/O at MTOW		
2.2.5 - T/O with simulated engine failure		
2.2.5.1 - T/O with simulated engine failure shortly after reachi		
2.2.5.2 - T/O with simulated engine failure between V1 + V2		
2.2.6 & 6.6.1 - Instrument T/O with RTO+EVAC		
3.3.6.5 - Windshear T/O		
3.3 - FLIGHT MANOUVRES & PROCEDURES		
3.3.1 - Turns with and w/o spoilers		
3.3.3 - Normal Operation of Systems and controls engineer's par		
3.3.4 - Normal and Abnormal Operation of Systems		
3.3.4.0 - Engine		
3.3.4.1 - Pressurization and airconditioning		
3.3.4.2 - Pitot/Static System		
3.3.4.3 - Fuel System		
3.3.4.4 - Electrical System		
3.3.4.5 - Hydraulic System		
3.3.4.6 - Flight Control and Trim System		
3.3.4.7 - Anti-Ice/Glare Shield Heating		
3.3.4.8 - Autopilot/Flight Director		
3.3.4.9 - Stall warning or stall avoidance devices and stability		
3.3.4.10 - GPWS, WXR, Radio Altimeter, Transponder		
3.3.4.11 - Radios, Navigation Equipment		
3.3.4.12 - LDG Gear/Brakes		
3.3.4.13 - Slat and Flap System		
3.3.4.14 - APU		
3.3.6 - Emergency Procedures		
3.3.6.1 - Fire drills e.g. engine, APU, cabin, cargo, flight deck,		
3.3.6.2 - Smoke control and removal		
3.3.6.3 - Engine failures, shut down and inflight start		
3.3.6.5 - Windshear T/O and LDG		
3.3.6.6 - Simulated Cabin Pressure failure/Emergency desc		
3.3.6.7 - Incapacitation of flight crew members		
3.3.6.8 - Other Emergency Procedures		
3.3.6.9 - ACAS (TCAS) event		
3.3.7 - Steep turns 45° bank		

Figure 29: FCLR Coverage

The evaluation of the TrainingManager is currently ongoing, in parallel to the syllabi evaluation.

5.3.3 TrainingManager aspects on the development process

Description	The TrainingManager allows specification of syllabi. For transition trainings, differences between the source and the target aircraft can be calculated and visualized.	
Process	Original <ol style="list-style-type: none"> 1. Specification of the syllabus by the trainers in Word 2. Distribution of the syllabi via email, data file repository 	HoliDes update <ol style="list-style-type: none"> 1. Specification of the syllabus with the help of the TrainingManager 2. Distribution of the syllabi as Word file, but also storage of the syllabus in database. In future OSLC change management can be used to manage updates of the syllabi
HF aspects	<ul style="list-style-type: none"> • Adaption of the syllabus to the previous knowledge of pilots, as a preparation for further individualisation (i.e. online adaption based on training progress) • For the trainers, a significant reduction of time needed for the specification of syllabi can be reached by the use of the TrainingManager, because the TrainingManager allows focusing on the content of the training, rather than formatting a Word file. • The Trainers need to check the coverage of the FCLRs and customer needs. As the program knows the requirements, and can check the coverage, errors are reduced, thus customer and trainer satisfaction are higher. 	
Benefits	<ol style="list-style-type: none"> 1. Individualisation of syllabi ensures customer satisfaction and helps to maintain market share with up-to-date techniques and trends 2. Faster syllabus generation allows faster response to customer requests and feedback from other trainers. 3. The model-based pilot training fosters standardisation within and across aircraft types. 4. The (not yet implemented) storage of the syllabi and use of the RTP Change Management features will allow better management of syllabi updates and distribution 	

6 Conclusion and Summary

This document has taken inputs from WP1-WP5 to assist in the continuing development of the HF-RTP instances for the Aeronautics domain.

This document describes the latest status of the AdCoS and HF-RTP instances in WP7. In the Diversion assistant, video and bio-signal data are used to infer the state of a pilot and to select appropriate means of mitigation that are via adaptation communicated to pilots. Tools that support these functions were in part integrated in the AdCoS, in part there are new concepts to be tested in the last year of the HoliDes project.

Additionally, the HF-RTP instance for diversion assistant consists of tools that improve the development process in the verification and validation phase addressing several important issues related to HF activities. These tools are not integrated in the AdCoS, rather they are applied externally to improve the quality of the AdCoS.

The HF-RTP instance will be finished and applied in the aeronautics experiment to demonstrate the ability to support the process of conducting an experiment as well as to support adaptation of the AdCoS.

In the EATT training AdCoS, Syllabi has been specified based on the comparison and categorisation of the differences between the A320 and the B737. These syllabi have been specified in the TrainingManager, a WP2 MTT. As a pre-requisite, the WP2 MTT MagicPED has been used for the specification of the procedures (A320 and B737).

The TrainingManager has been presented to LFT Trainers in a workshop, and a lot of feedback was gathered for improvement of the TrainingManager, also the trainers were already very satisfied with the current version. Further tests will be conducted, when the feedback is implemented in the TrainingManager.

Regarding the AdCoS, a second cycle of evaluation has been conducted at LFT, where two new trainees have performed successfully a transition training from B737 to A320. In the near future, a 3rd round of training will be performed.

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

For both AdCoS an important benefit is that, the use of IOS compliant MTTs allows setting up a flexible toolchain, which is flexible, re-usable and can be extended with further MTTs later on, e.g. the models can be linked to requirements tools and/or change management tools. This wouldn't be possible without HoliDes HF-RTP.

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

7 Way forward and upcoming activities

7.1 Diversion assistant use-case

This document has shown the last version of the HF-RTP for the diversion assistant use-case. The work will still continue until the end of the project but no further major releases will be made. The continuation work will comprise

- Finalization of integration, especially for tools added later to the HF-RTP instance
- Testing of individual tools to verify requirements and asses benefits
- Application of HF-RTP instance as a whole to the final experiments.

7.2 EATT use-case

As described in the conclusion, we are currently in the evaluation phase of the syllabus. The 2nd round of evaluation has just been finished, and the 3rd and final round will start in May. In all three rounds of evaluations, B737 pilots are undergoing a transition training to A320, in order to acquire a Flight Licence for an A320. In the first two rounds, TRS has performed the training, but in the 3rd round the training will be performed by an LFT trainer, based on the HoliDes syllabus.

In parallel to that, the TrainingManager will be tested by LFT trainers, who are active in the writing of Syllabi. A first workshop has been performed, and follow up activities are planned. Currently the TrainingManager will be presented at the WATS 2016 conference, the “World Aviation Training Conference and Tradeshow” to a wider audience of training organisations.



HoliDes

Holistic Human Factors Design of
Adaptive Cooperative Human-
Machine Systems



8 Annex

8.1 Annex I

RDF description of the OSLC ResourceShape for the Task Models:

```
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#" xmlns:oslc="http://open-services.net/ns/core#"
xmlns:dcterms="http://purl.org/dc/terms/" xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
  <oslc:ResourceShape rdf:about="http://srvvwm03.offis.uni-oldenburg.de:8080/TaskAnalysisService-
0.0.1/resourceShapes/OSLCHierarchicalTaskModelEntry">
    <oslc:property>
      <oslc:Property>
        <oslc:readOnly rdf:datatype="http://www.w3.org/2001/XMLSchema#boolean">false</oslc:readOnly>
        <oslc:name>version</oslc:name>
        <oslc:valueType rdf:resource="http://www.w3.org/2001/XMLSchema#dateTime"/>
        <oslc:propertyDefinition rdf:resource="http://de.offis.hcd/taskanalysisversion"/>
        <oslc:occurs rdf:resource="http://open-services.net/ns/core#Exactly-one"/>
        <dcterms:title rdf:parseType="Literal">version</dcterms:title>
      </oslc:Property>
    </oslc:property>
    <oslc:property>
      <oslc:Property>
        <oslc:readOnly rdf:datatype="http://www.w3.org/2001/XMLSchema#boolean">false</oslc:readOnly>
        <oslc:name>model</oslc:name>
        <oslc:valueType rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
        <oslc:propertyDefinition rdf:resource="http://de.offis.hcd/taskanalysismodel"/>
        <oslc:occurs rdf:resource="http://open-services.net/ns/core#Exactly-one"/>
        <dcterms:title rdf:parseType="Literal">model</dcterms:title>
      </oslc:Property>
    </oslc:property>
    <oslc:property>
      <oslc:Property>
        <oslc:readOnly rdf:datatype="http://www.w3.org/2001/XMLSchema#boolean">false</oslc:readOnly>
        <oslc:name>id</oslc:name>
        <oslc:valueType rdf:resource="http://www.w3.org/2001/XMLSchema#integer"/>
        <oslc:propertyDefinition rdf:resource="http://de.offis.hcd/taskanalysisid"/>
        <oslc:occurs rdf:resource="http://open-services.net/ns/core#Exactly-one"/>
        <dcterms:title rdf:parseType="Literal">id</dcterms:title>
      </oslc:Property>
    </oslc:property>
    <oslc:property>
      <oslc:Property>
        <oslc:readOnly rdf:datatype="http://www.w3.org/2001/XMLSchema#boolean">false</oslc:readOnly>
        <oslc:name>description</oslc:name>
        <oslc:valueType rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
        <oslc:propertyDefinition rdf:resource="http://de.offis.hcd/taskanalysisdescription"/>
        <oslc:occurs rdf:resource="http://open-services.net/ns/core#Exactly-one"/>
        <dcterms:title rdf:parseType="Literal">description</dcterms:title>
      </oslc:Property>
    </oslc:property>
    <oslc:property>
      <oslc:Property>
        <oslc:readOnly rdf:datatype="http://www.w3.org/2001/XMLSchema#boolean">false</oslc:readOnly>
        <oslc:name>airline</oslc:name>
        <oslc:valueType rdf:resource="http://open-services.net/ns/core#Resource"/>
        <oslc:representation rdf:resource="http://open-services.net/ns/core#Reference"/>
        <oslc:range rdf:resource="http://de.offis.hcd/taskanalysisOSLCAirline"/>
        <oslc:propertyDefinition rdf:resource="http://de.offis.hcd/taskanalysisairline"/>
        <oslc:occurs rdf:resource="http://open-services.net/ns/core#Exactly-one"/>
        <dcterms:title rdf:parseType="Literal">airline</dcterms:title>
      </oslc:Property>
    </oslc:property>
    <oslc:property>
      <oslc:Property>
        <oslc:readOnly rdf:datatype="http://www.w3.org/2001/XMLSchema#boolean">false</oslc:readOnly>
        <oslc:name>aircraft</oslc:name>
```



HoliDes

Holistic Human Factors Design of Adaptive Cooperative Human-Machine Systems



```
<oslc:valueType rdf:resource="http://open-services.net/ns/core#Resource"/>
<oslc:representation rdf:resource="http://open-services.net/ns/core#Reference"/>
<oslc:range rdf:resource="http://de.offis.hcd/taskanalysisOSLCAircraft"/>
<oslc:propertyDefinition rdf:resource="http://de.offis.hcd/taskanalysisaircraft"/>
<oslc:occurs rdf:resource="http://open-services.net/ns/core#Exactly-one"/>
<dcterms:title rdf:parseType="Literal">aircraft</dcterms:title>
  </oslc:Property>
</oslc:property>
<oslc:describes rdf:resource="http://de.offis.hcd/taskanalysisOSLCHierarchicalTaskModelEntry"/>
<dcterms:title rdf:parseType="Literal">OSLCHierarchicalTaskModelEntry Resource Shape</dcterms:title>
</oslc:ResourceShape>
</rdf:RDF>
```

8.2 Annex II

```
<rdf:RDF xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#" xmlns:oslc="http://open-services.net/ns/core#"
xmlns:dcterms="http://purl.org/dc/terms/" xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
  <oslc:ResourceShape rdf:about="http://srvvmm03.offis.uni-oldenburg.de:8080/TaskAnalysisService-
0.0.1/resourceShapes/osLCTrainingModel">
  <oslc:property>
    <oslc:Property>
      <oslc:readOnly rdf:datatype="http://www.w3.org/2001/XMLSchema#boolean">false</oslc:readOnly>
      <oslc:name>version</oslc:name>
      <oslc:valueType rdf:resource="http://www.w3.org/2001/XMLSchema#dateTime"/>
      <oslc:propertyDefinition rdf:resource="http://de.offis.hcd/taskanalysisversion"/>
      <oslc:occurs rdf:resource="http://open-services.net/ns/core#Exactly-one"/>
      <dcterms:title rdf:parseType="Literal">version</dcterms:title>
    </oslc:Property>
  </oslc:property>
  <oslc:property>
    <oslc:Property>
      <oslc:readOnly rdf:datatype="http://www.w3.org/2001/XMLSchema#boolean">false</oslc:readOnly>
      <oslc:name>id</oslc:name>
      <oslc:valueType rdf:resource="http://www.w3.org/2001/XMLSchema#integer"/>
      <oslc:propertyDefinition rdf:resource="http://de.offis.hcd/taskanalysisid"/>
      <oslc:occurs rdf:resource="http://open-services.net/ns/core#Exactly-one"/>
      <dcterms:title rdf:parseType="Literal">id</dcterms:title>
    </oslc:Property>
  </oslc:property>
  <oslc:property>
    <oslc:Property>
      <oslc:readOnly rdf:datatype="http://www.w3.org/2001/XMLSchema#boolean">false</oslc:readOnly>
      <oslc:name>description</oslc:name>
      <oslc:valueType rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
      <oslc:propertyDefinition rdf:resource="http://de.offis.hcd/taskanalysisdescription"/>
      <oslc:occurs rdf:resource="http://open-services.net/ns/core#Exactly-one"/>
      <dcterms:title rdf:parseType="Literal">description</dcterms:title>
    </oslc:Property>
  </oslc:property>
  <oslc:property>
    <oslc:Property>
      <oslc:readOnly rdf:datatype="http://www.w3.org/2001/XMLSchema#boolean">false</oslc:readOnly>
      <oslc:name>atm</oslc:name>
      <oslc:valueType rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
      <oslc:propertyDefinition rdf:resource="http://de.offis.hcd/taskanalysisatm"/>
      <oslc:occurs rdf:resource="http://open-services.net/ns/core#Exactly-one"/>
      <dcterms:title rdf:parseType="Literal">atm</dcterms:title>
    </oslc:Property>
  </oslc:property>
  <oslc:property>
    <oslc:Property>
      <oslc:readOnly rdf:datatype="http://www.w3.org/2001/XMLSchema#boolean">false</oslc:readOnly>
      <oslc:name>airline</oslc:name>
      <oslc:valueType rdf:resource="http://open-services.net/ns/core#Resource"/>
      <oslc:representation rdf:resource="http://open-services.net/ns/core#Reference"/>
      <oslc:range rdf:resource="http://de.offis.hcd/taskanalysisOSLCAirline"/>
      <oslc:propertyDefinition rdf:resource="http://de.offis.hcd/taskanalysisairline"/>
```



HoliDes

Holistic Human Factors Design of Adaptive Cooperative Human-Machine Systems



```
<oslc:occurs rdf:resource="http://open-services.net/ns/core#Exactly-one"/>
<dcterms:title rdf:parseType="Literal">airline</dcterms:title>
</oslc:Property>
</oslc:property>
<oslc:property>
  <oslc:Property>
    <oslc:readOnly rdf:datatype="http://www.w3.org/2001/XMLSchema#boolean">false</oslc:readOnly>
    <oslc:name>aircraft</oslc:name>
    <oslc:valueType rdf:resource="http://open-services.net/ns/core#Resource"/>
    <oslc:representation rdf:resource="http://open-services.net/ns/core#Reference"/>
    <oslc:range rdf:resource="http://de.offis.hcd/taskanalysisOSLCAircraft"/>
    <oslc:propertyDefinition rdf:resource="http://de.offis.hcd/taskanalysisaircraft"/>
    <oslc:occurs rdf:resource="http://open-services.net/ns/core#Exactly-one"/>
    <dcterms:title rdf:parseType="Literal">aircraft</dcterms:title>
  </oslc:Property>
</oslc:property>
<oslc:describes rdf:resource="http://de.offis.hcd/taskanalysisOSLCTrainingModel"/>
<dcterms:title rdf:parseType="Literal">OSLCTrainingModel Resource Shape</dcterms:title>
</oslc:ResourceShape>
</rdf:RDF>
```

8.3 Annex III

Annex III is confidential and published as separate file *D7.8 - Annex III - TrainingSyllabus V3*.