



## HoliDes

Holistic Human Factors Design  
of Adaptive Cooperative  
Human-Machine Systems

# HoliDes

## Update of the HoliDes HF-RTP Handbook

A user's guide from the theoretical background to actual application

<b>Project Number:</b>	332933
<b>Classification:</b>	Public
<b>Work Package(s):</b>	WP 1
<b>Milestone:</b>	M 5
<b>Issue Date:</b>	September 2016
<b>Document Timescale:</b>	Project Start Date: October 1, 2013 Written August 2016
<b>Deliverable Overview:</b>	<b>Main document:</b> D1.7 - HF-RTP Version 2.0 incl. Methodology and Requirements Analysis Update
<b>Compiled by:</b>	Sonia Bilbao, Sara Sillaurren, TEC
<b>Authors:</b>	Linda Onnasch, HFC Ian Giblett, AGI Nicolas Schneider, AGI
<b>Reviewers:</b>	Morten Larsen - AWI, Fabio Tango - CRF
<b>Technical Approval:</b>	Jens Gaertner (Airbus)
<b>Issue Authorisation:</b>	Sebastian Feuerstack, OFF

© All rights reserved by HoliDes consortium

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

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

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

---

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

## Table of Contents

<b>1</b>	<b>Project Background - Introduction to HoliDes</b>	<b>6</b>
<b>2</b>	<b>System Development in Short</b>	<b>7</b>
<b>3</b>	<b>Introducing Human Factors into System Development</b>	<b>9</b>
<b>4</b>	<b>The HF-RTP</b>	<b>10</b>
4.1	RTP Background	11
4.1.1	RTP Definitions	13
4.2	Approach for Human Factors Integration	14
<b>5</b>	<b>How to use the Platform Builder</b>	<b>19</b>
5.1	What is the Platform Builder?	19
5.2	Accessing the Platform Builder	19
5.3	Landing Page	20
5.4	Creating an HF-RTP Instance	21
5.4.1	Setting Criteria for MTT selection	21
5.4.2	Choosing the appropriate MTT	22
5.4.3	Creating a project	27
5.5	My Projects	29
5.6	The Adaptation Framework	30
5.6.1	Control Loops as the constituting part	30
5.6.2	How to use the Adaptation Framework	33
5.7	Explore or Manage MTTs: The Database	37
5.8	General Information and How-To	39
<b>6</b>	<b>Tailoring the HF-RTP to your purpose</b>	<b>40</b>
6.1	Tailoring Step 1	41
6.1.1	Example from the Control Room Domain	41
6.2	Tailoring Step 2	42
6.2.1	Continued Example from the Control Room Domain	43
6.3	Tailoring Step 3	44
6.4	Tailoring Step 4	45
6.4.1	Build in OSLC support natively	46
6.4.2	Extend tool to support OSLC with API	47
6.4.3	Extend tool to support OSLC without an API	48
6.4.4	Example: Connecting two tools	49

<b>7</b>	<b>Summary</b> .....	<b>51</b>
<b>8</b>	<b>References</b> .....	<b>52</b>

## List of Figures

Figure 1:	V-model of the system development lifecycle.....	8
Figure 2:	RTP tailoring for an RTP instance.....	12
Figure 3:	Human Factors cycle .....	15
Figure 4:	Alignment of HF activities to system development lifecycle ...	17
Figure 5:	Platform Builder Login Screen .....	19
Figure 6:	Pop-up window to create a new user account.....	20
Figure 7:	The landing page of the Platform Builder.....	21
Figure 8:	Search MTTs screen after user login .....	22
Figure 9:	Preliminary MTT list .....	23
Figure 10:	MTT Comparator .....	24
Figure 11:	Selecting an MTT from the Preliminary MTT List .....	27
Figure 12:	Creating a new project.....	28
Figure 13:	List of projects available by clicking on "Get My Projects". ..	29
Figure 14:	Viewing the project details.....	29
Figure 15:	Basic control loop of the adaptation framework .....	30
Figure 16:	Control loop to control a process .....	31
Figure 17:	Control loop used by an agent to control a process .....	31
Figure 18:	Control loop to control a process through a user interface ..	31
Figure 19:	Finite amount of primitive control loop .....	32
Figure 20:	Example of an AdCoS model with primitive control loop.....	33
Figure 21:	Adaptation framework integrated in Platform Builder .....	34
Figure 22:	Icons of the four Holides AdCoS .....	34
Figure 23:	Primitive control loop for Workload balancing AdCoS.....	35
Figure 24:	Interface for designing new AdCoS models .....	36
Figure 25:	Example list of HF issues and requirements. ....	36
Figure 26:	MTT Administration tool, upper screen part.....	37
Figure 27:	MTT Administration tool, lower screen part .....	38
Figure 28:	Youtube tutorial videos on the Platform Builder. ....	39
Figure 29:	Identification of issues in the development process.....	42
Figure 30:	Integration of MTTs into the development process .....	43
Figure 31:	OSLC Support built in natively.....	46
Figure 32:	Extending a 3rd party tool .....	47
Figure 33:	Extending a 3rd party tool without an API.....	48
Figure 34:	The Lyo Project .....	49
Figure 35:	The EA/HF Filer integration .....	50

## List of Tables

Table 1: RTP definitions .....	13
Table 2: Three approaches to implement MTT mappings .....	45

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

## 1 Project Background - Introduction to HoliDes

The aim of the HoliDes project is to address the development and qualification of Adaptive Cooperative Human-Machine Systems (AdCoS) where many humans and many machines act together, cooperatively, in a highly adaptive way in order to guarantee fluent and cooperative task achievement. So far, the objective is to develop processes, techniques and software tools that enable development and qualification AdCoS.

HoliDes is positioned within the Sub-Programme N°8 of the Call (“Human-centric design of embedded systems”). The lines of research can be described in along three main dimensions:

- 1) enhancing the safety and confidence of users and the public by considering human operator needs during Embedded System development of systems with many men and many machines. The HoliDes technology will support the entire engineering life-cycle, in particular in early phases. This will help minimizing the danger of human operator workload peaks, restricted situation awareness and conflicts between humans and machines and will thus increase safety
- 2) increasing the automation of tasks by developing techniques and tools for implementing adaptation in cooperative human-machine systems that will enable to build systems and tasks with increased levels of adaptive automation. This will allow to shift control from the humans to the technical systems and back progressively in real-time, taking into account contextual factors such as the capacity and load of human operators (e.g human operator behaviour, intentions and state)
- 3) to contribute to the fulfilment of the user-centered and technical objectives by providing open innovation environment. Therefore, the HoliDes partners will integrate their techniques and tools into a new **Human Factors Reference Technology Platform (HF-RTP)** that is closely linked to the CESAR RTP to enable interoperability.

The HF-RTP is complemented with a new methodology, which supports human-centered development of AdCoS. This enables holistic development of AdCoS both, from the human factors and technical perspectives. The HF-RTP and methodology are developed as an open innovation

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

environment to support continuous technology transfer from research to the industry. The AdCoS are developed in four domains: Health, Aeronautics, Control Rooms and Automotive.

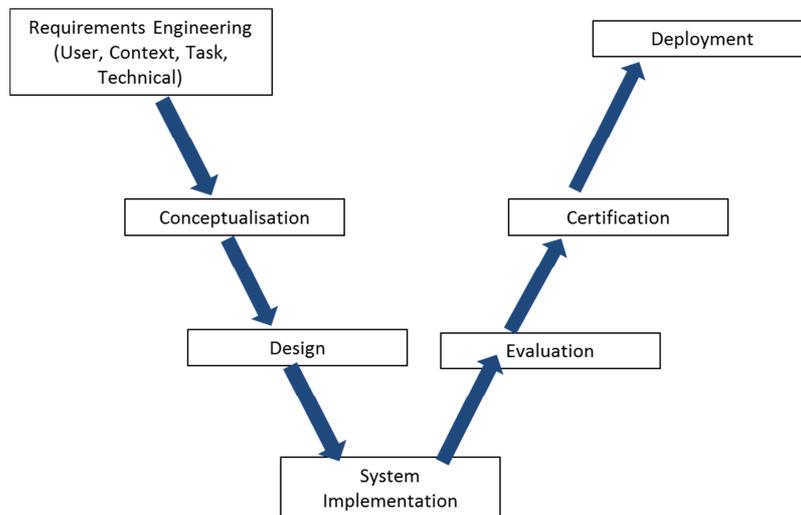
## 2 System Development in Short

Up to now, most system development processes are based on classical v-model approaches. A v-model is a graphical representation of the system development lifecycle to provide guidance for planning and realization of projects. The objectives of standardizing the development by applying this approach are:

- Minimizing projects risks by enabling a transparent and traceable project progress
- Improving quality by standardizing the development process
- Reducing total cost by introducing a transparent and straight forward process
- Improving the communication between all involved partners by introducing standards

In this regard, the v-model can also be understood as a project management tool. It summarizes the main steps to be taken and describes the activities to be performed for each step.

An example for such a model is presented in Figure 1 **Error! La autreferencia al marcador no es válida.:**



**Figure 1: V-model of the system development lifecycle**

The v-model in Figure 1 describes the system development lifecycle in seven distinct steps.

The left part of the model represents the **specification stream**. This includes the main steps of the actual system development and results in the first system implementation, e.g. a prototype of the system in question.

The first step deals with the requirements engineering as the starting point of system development. Requirements have to be identified regarding potential system users, the context, in which the system is planned to be implemented, the task that is going to be performed with the system as well as technical requirements that are mandatory to realize the system.

The second step represents the first ideas of the system to be developed. In this part, ideas regarding system functionality or interaction concepts have to be discussed and identified.

The third step constitutes the concrete system design. This includes the functional design of the system as well as the interface design.

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

The result of completing these steps is a first version of the system that can be implemented to start with an extensive testing (evaluation & certification). This is represented by the right part of the model, the **validation & deployment stream.**

One of the most important steps in this part is the evaluation of the system. This involves aspects like the operational or the performance qualification and can be realized with user tests, expert evaluations, simulations, etc.

Depending on the application domain, some systems might further have to pass certain certification processes. This is particularly important in safety-related domains like aviation.

If the evaluation and certification have provided acceptable results, the last step in the current v-model constitutes the actual deployment of the system.

However, anyone, who was ever part in a system development lifecycle, knows that system development is not that straight forward. What is needed are several iterative steps. I.e. most of the steps represented in the v-model are passed several times until the final system is ready for deployment.

### **3 Introducing Human Factors into System Development**

Even with the use of a v-model as a guide for project management, system development is often related to high cost and a prolonged development time compared to what was initially planned.

One reason for that is the lack of integrating Human Factors (HF) from the beginning into the development process. Today, a common approach is to consider HF only in the evaluation phase by conducting usability or user tests, if at all. This late introduction of HF eventually leads to higher development cost as HF-related problems like usability issues or

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

inappropriate function allocation between the human and the system are recognized only very late in the process, i.e. when the system has already been developed. To resolve these issues it is often necessary to start system development again at the design phase or sometimes even at the beginning, the requirements engineering. Cost and development time increase accordingly. By introducing Human Factors into the system development lifecycle from the beginning, some of these feedback loops could be prevented.

However, system developers often feel challenged by this approach in two ways:

- First, it is not always easy to find the right HF methods to be introduced to the according development phases. This is particularly true for non-HF experts.
- Second, the integration of HF data into the development process can be complicated as HF methods often rely on qualitative measures like interviews or observations. As these data are most often not interoperable with other tools, results often are not integrated into further development steps.

These challenges are addressed by the HF-RTP, which is explained in the following section.

## **4 The Human Factors Reference Technology Platform (HF-RTP)**

This chapter shall provide basic information about the HoliDes Human Factors Reference Technology Platform (HF-RTP), the Interoperability Specification (IOS) and other interoperability technologies that will be utilized to support human factors engineering. It will clarify some of the terminology that surrounds RTPs by giving succinct definitions.

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

## 4.1 RTP Background

A Reference Technology Platform (RTP) by itself is a set of tools or services which are IOS compliant to facilitate the exchange of project lifecycle data during design time. It provides a set of connectable tools needed to support collaboration between business processes of an enterprise, for example interactions with customers, suppliers and partners. The interoperability between the integrated tools is gaining importance for successful and efficient product engineering. An Interoperability Specification (IOS) addresses the needs for interoperability and collaboration between tools across the entire engineering lifecycle.

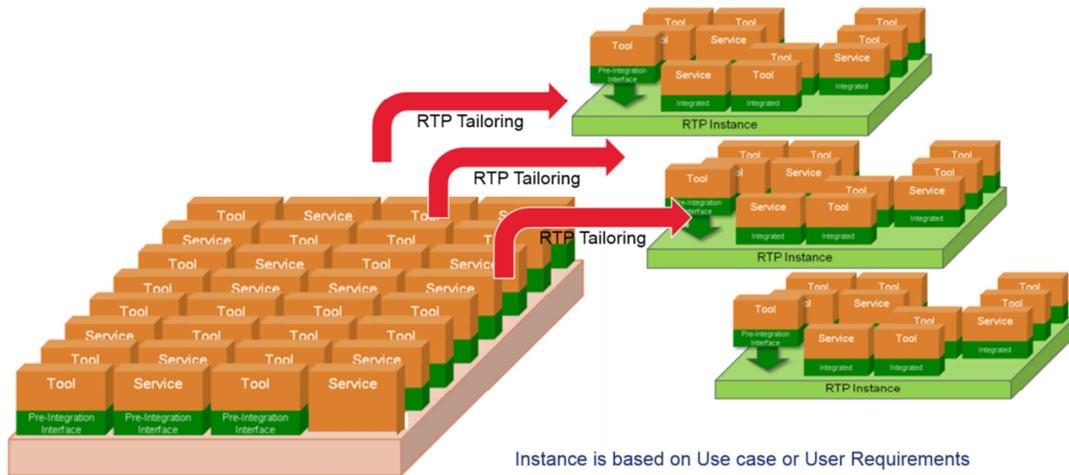
Data need to be exchanged in a common way using technologies that link them for full traceability throughout the product lifecycle. Requirements need to be traced to models they satisfy, bug reports need to be attached to the code modules with the error and test cases need to be linked to the components that they test. The internet is by far the most successful example of a scalable collaborative platform. The RTP concept aims to apply the principles which made the internet so successful and apply them to system engineering.

Hence, the main idea behind an RTP is to define an Interoperability Specification (IOS) to simplify the data exchange through unification / standardization of the underlying data models and data exchange format and to build a reference implementation of that standard. The OSLC (Open Services for Lifecycle Collaboration) standard provides such a specification of what information is necessary to define a requirement. Hence, if a database server offers an OSLC service for requirements, any client tool which is OSLC compliant can process the requirements database in a unified way. Thereby, the use of open standards for integration lowers the barriers to bringing new tools and processes into an already existing tool chain.

It is important to emphasize, that the RTP is not a piece of software or hardware that one could pick up and install. It is a library of tools and services called RTP Components which exist in a pre-integrated state. They are not connected directly and require instantiation before use in a

phase (tailoring, Figure 2). The instantiated RTP then allows the seamless development of different systems like the AdCoS developed in HoliDes.

Figure 2 illustrates an example which moves from the RTP to an RTP Instance.



**Figure 2: RTP tailoring for an RTP instance.**

### 4.1.1 RTP Definitions

Table 1 defines the following terms and definitions as used in HoliDes.

**Table 1: RTP definitions**

Term	Definition
<b>RTP</b>	(Reference Technology Platform) A set of pre-integrated tools and services that could be compiled and installed to satisfy the needs of a particular use case.
<b>RTP Instance</b>	A subset of RTP components that have been integrated through a stage known as tailoring.
<b>RTP Component</b>	A generic term to define an un-tailored ready to integrate component.
<b>IOS</b>	(Interoperability Specification) The specification which defines how the integration between RTP Components shall work.
<b>OSLC</b>	(Open Services for Lifecycle Collaboration) An open consortium which seeks to facilitate the exchange of data between engineering tools by using linked data. A key component of the IOS.
<b>Tool Chain</b>	A combination of tools or services which are used in a scenario to support a use case. The tool chain represents a particular flow of data through an engineering process.
<b>Tool</b>	The term tool and service are often used interchangeably. Strictly speaking, a tool represents software which an engineer will use to create data. For example to create requirements, models or documents.
<b>Service</b>	A service processes data and information that has already been produced by tools but is quite capable of creating new information. An example of a service could be a model transformation or an impact analysis.
<b>Interoperability</b>	Is defined as the ability of two or more tools or services to exchange information and then to work with that exchanged information.
<b>Traceability</b>	The ability to create and observe a link between information which can exist across different domains.

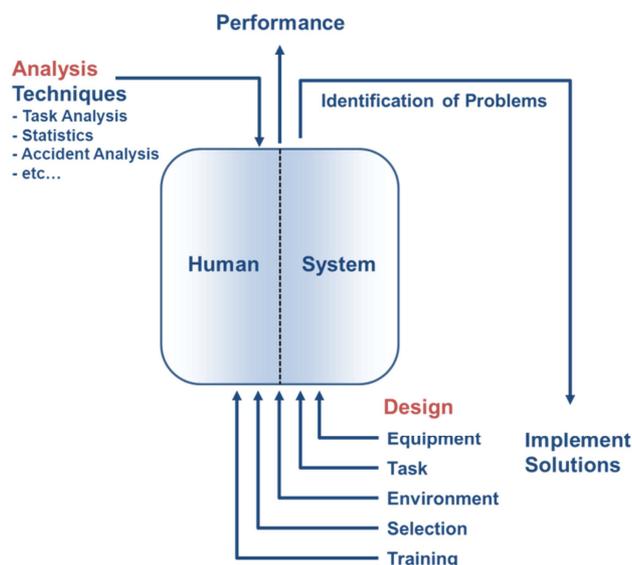
	<p style="text-align: center;"><b>HoliDes</b></p> <p style="text-align: center;"><b>H</b>olistic Human Factors <b>D</b>esign of Adaptive Cooperative Human- Machine Systems</p>	
---	---	---

## 4.2 Approach for Human Factors Integration

RTP related projects funded by ARTEMIS like CESAR [1], MBAT [2] or Crystal [3] focus on classical engineering processes, methods and tools; hence the definition of the RTP focuses on those aspects. One of the main goals of the HoliDes project is to extend the existing RTP concepts with an integration of human factors methods and tools into a *Human Factors Reference Technology Platform* (HF-RTP).

Human factors research and human centered engineering has become mandatory in the development and the design of modern technological systems. Human factors encompass the study of factors that influence the interaction of people with systems as well as using (and developing) tools and methods to support the development of systems. High quality in terms of reaching the human factors goals is not so much an attribute of the technical system developed but it is describing properties of the interaction of specified users performing a defined task in a particular environment or context using a specific system. Four main principles drive the human factors work in development projects: safety, performance, comfort and aesthetics [4] and are related to four human factors lines of investigation: physical, cognitive, social and emotional aspects of HMI.

In a very general sense, human centered design is considered as being a cyclic process, linked to the development process. We find examples of this view in [5] or [6] as well as in the norms and standards describing the usability engineering process (ISO 9241-Part 210 which replaced ISO 13407).



**Figure 3: Human Factors cycle (modified from Wickens et al. 2004 [7])**

The goals of human factors work are accomplished through a number of procedures in the human factors cycle (Figure 3). The starting point and first phase of the cycle are often analysis activities used for a diagnosis of deficiencies of an existing human—system interaction or of a projected but not (yet) implemented case of human-machine interaction. The detected deficiencies will be translated into so called *human factors issues* to be solved in the development process.

The scope of human factors issues was broadened during the last decades from an investigation of interaction and ergonomics in a mainly physical sense towards including cognitive ergonomics as well as team related and organizational factors influencing the human-system interaction at the sharp end [6].

During the second phase, the design phase human factors work includes the procedures supposed to implement solutions to the HF issues. These activities deal with a number of relevant elements of human-machine interaction like the task, the interfaces, the planned interaction or aspects of the environment as well as selection and training of personnel.

In the third phase of human-centered development, the evaluation phase, the resulting human-machine interaction is assessed, considering quality criteria like mental workload, situation awareness, usability or user experience, and human error, depending on the issues identified in the diagnosis phase.

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

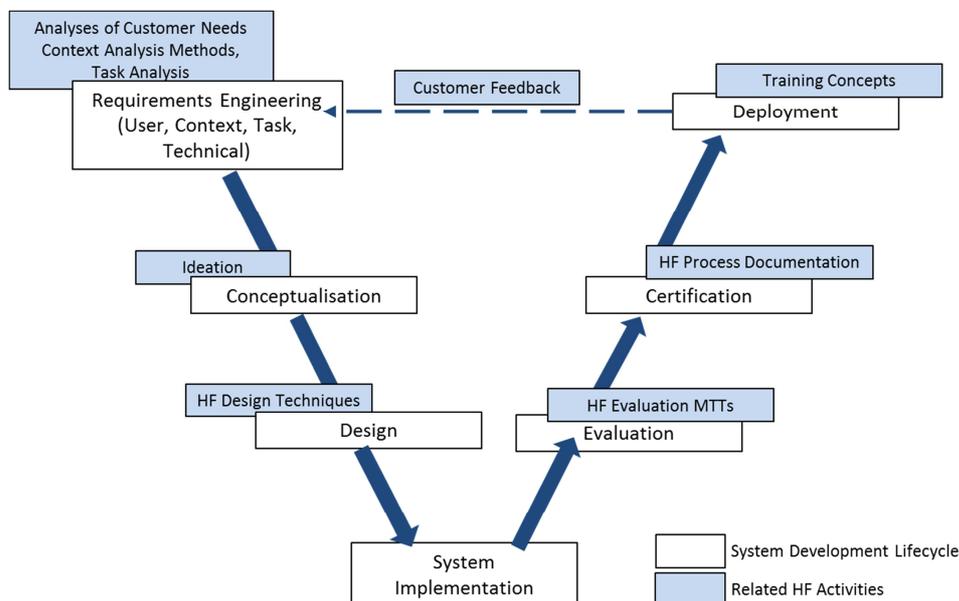
Analysis (1) and evaluation (3) activities often apply methods of empirical investigation while the outcome of such methods is often used to create models of the observed phenomena. The design activities (2) use formative techniques and tools to create new interfaces, task structures or interactions designs.

Typical examples for applied methods and tools during the three HF activities are:

- analysis
  - (cognitive) task analysis
  - critical incident analysis
  - accident and incident analysis
  - contextual inquiries using
  - interviews, questionnaires
  - user profiles, personas, scenarios
  
- design
  - interaction design
  - visual design, GUI
  - task design and task allocation
  - paper prototyping
  - wireframes
  - training design
  - selection procedures
  
- evaluation
  - hallway testing
  - heuristic evaluation
  - cognitive walk through
  - laboratory usability testing
  - usability questionnaires
  - simulator studies (e.g. driving or flight simulators)

From a generalized point of view, the three types of activities mentioned above can be performed in more than one of the lifecycle phases, e.g. error analysis is often done in the operations phase and could lead to

major redesign or even to the conception or the development of a new system generation. Many usability specialists advocate for evaluation starting early in the development process (a classic is [7]). Although we acknowledge that there is no perfect congruence between HF activities and development lifecycle, our goal is to achieve a good integration of HF activities into the standardized engineering approaches. Figure 4 shows the results of this effort.



**Figure 4: Alignment of HF activities to the V-Model of system development lifecycle**

The V-model in Figure 4 shows the alignment of HF activities to the seven phases of the system development lifecycle described earlier (page 7). The HF related activities represent superordinate terms which comprise various HF methods and techniques for each step (except system implementation). For example, typical methods summarized in the first step, analyses of customer needs, context and task, are GOMS Analysis & Prediction (Goals, Operators, Methods, and Selection rule; [8]), the Situation Awareness Requirements Analysis [9] or the ISONorm (Questionnaire to gather information on a system's compatibility with the DIN EN ISO 9241; [10]). For the design phase HF methods comprise methods like Subject Matter Expert Interviews or the Theatre Technique

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

for Acceptance Test [11]. The evaluation phase encompasses methods or techniques like the Nasa-TLX [12] or the Bedford Scale [13] for workload evaluation, SAGAT [14] and SART [15] for situation awareness, or the heuristic evaluation [16] and cognitive walkthrough [17] to assess the usability of a system.

These HF related methods and techniques are part of the HF-RTP. The integration of these activities into system development and the selection of the appropriate methods for a specific purpose are the objectives of the Platform Builder and of the tailoring process.

## 5 How to use the Platform Builder

### 5.1 What is the Platform Builder?

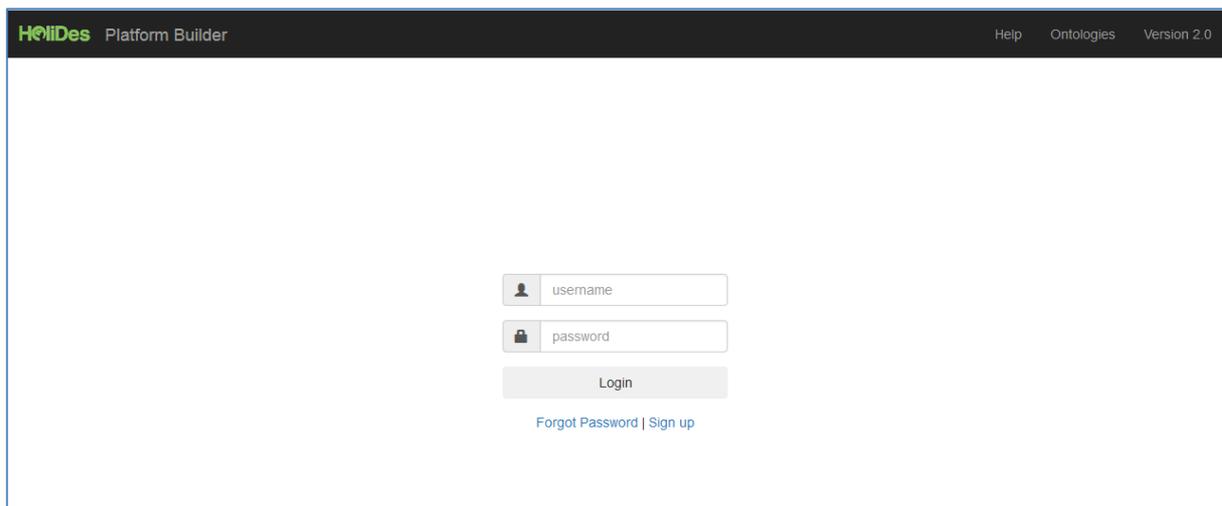
The Platform Builder is a web application and represents the front end of the HF-RTP, i.e. what becomes visible for the user. It is aimed at supporting the user to solve a concrete HF issue through an RTP instantiation. Based on an HF problem description, the Platform Builder aims to set up an instantiation of the HF-RTP for a specific domain project.

In doing so, the user is guided by certain selection criteria that have to be answered to find the appropriate methods, techniques or tools (MTTs) for the user's individual purpose.

### 5.2 Accessing the Platform Builder

The Platform Builder (PB) can be accessed through the following link:

<http://www.holidays.eu/PlatformBuilder/>



**Figure 5: Platform Builder Login Screen**

To login, the user has to enter his / her username and the according password (Figure 5).

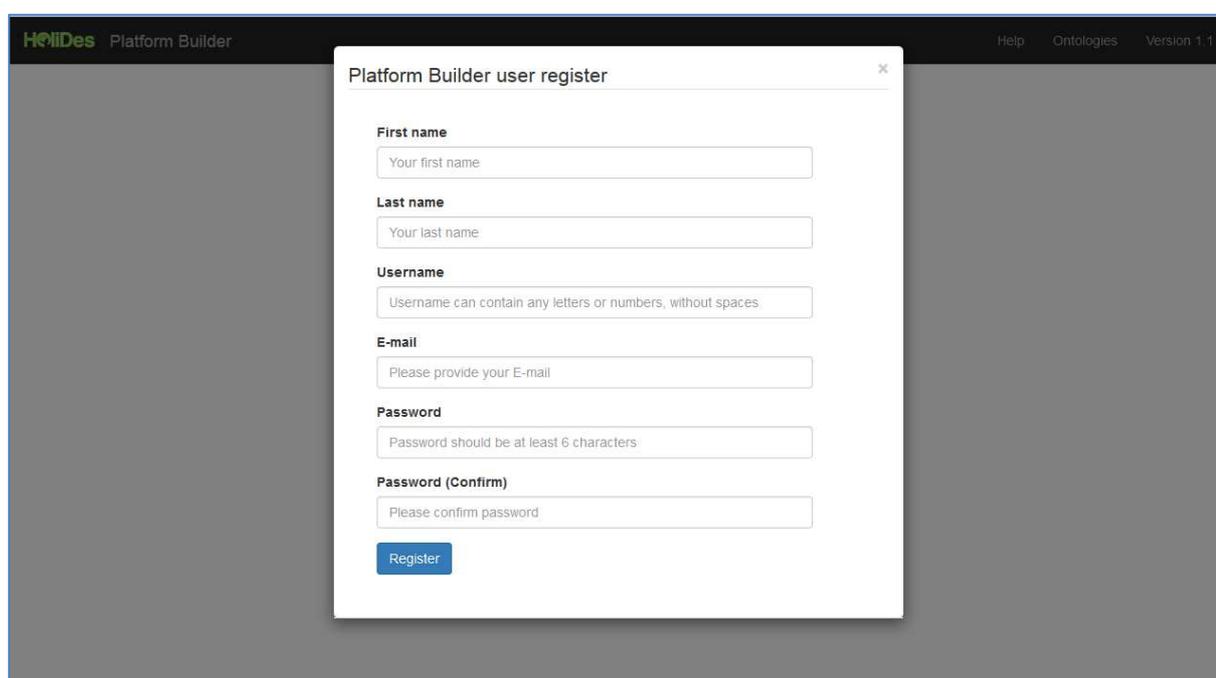


If a user has not created an account so far, he / she can do this by clicking on “Sign up”. A pop-up window appears, where the user then can fill in the required details to set up an account (see Figure 6).

If a user wants to test the PB first, he / she can also use the following generic login details:

Username: hfuser1

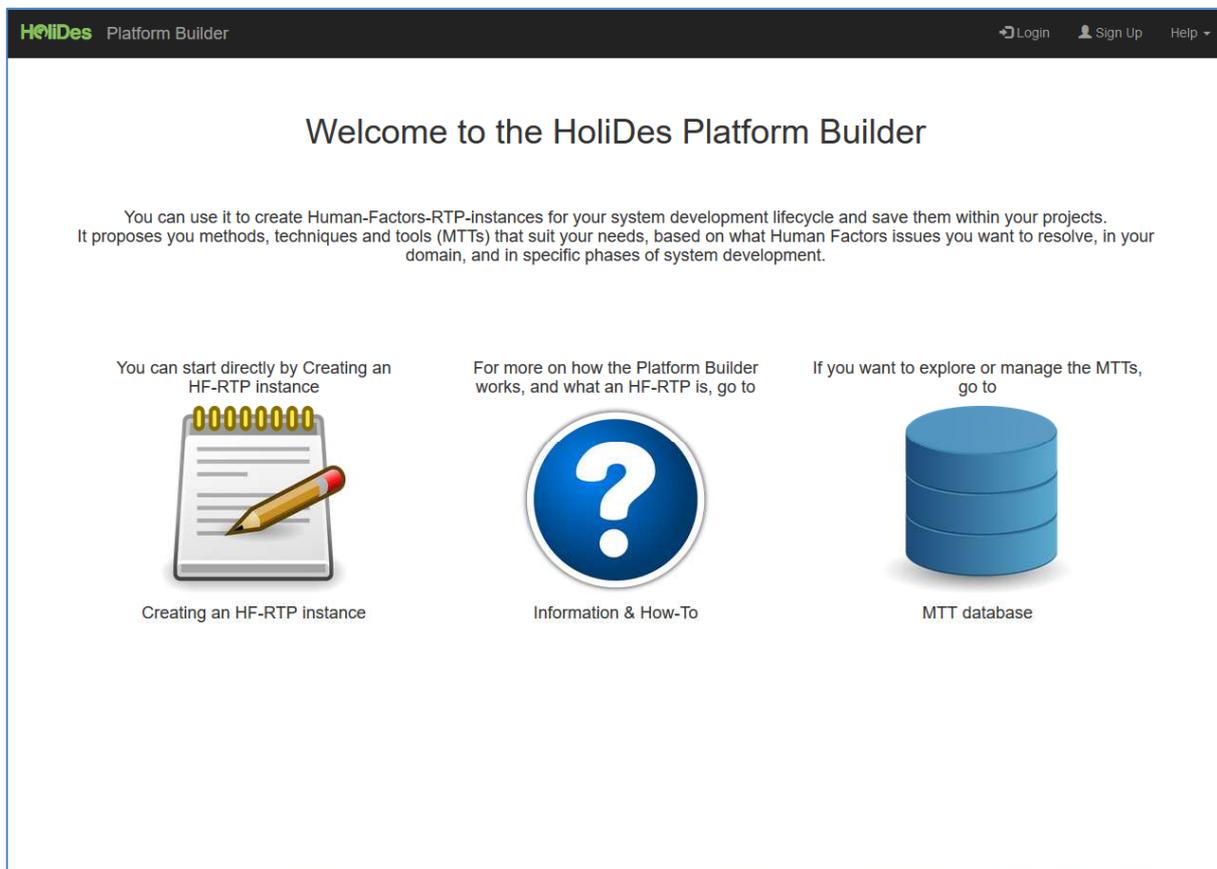
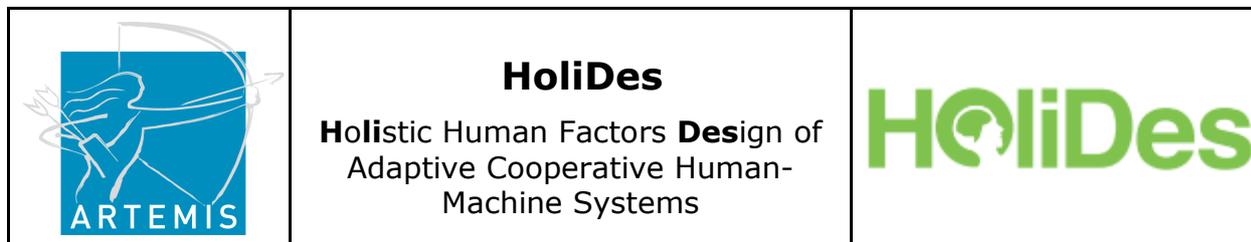
Password: hfuser1



**Figure 6: Pop-up window of the Platform Builder to create a new user account**

### 5.3 Landing Page

After the login the landing page of the Platform Builder appears where the user can select what he/ she wants to do first: create an HF-RTP instantiation, get information on the Platform Builder and the HF-RTP, or explore or manage MTTs (Figure 7).



**Figure 7: The landing page of the Platform Builder**

## 5.4 Creating an HF-RTP Instance

### 5.4.1 Setting Criteria for MTT selection

The user first has to choose between four different application domains the system is going to be developed for: Aeronautics, automotive, control room or health. This selection is mandatory because it represents a relevant input for the MTT search as several HF related MTTs are developed for a specific domain only and are therefore not appropriate or applicable in other domains.

After selecting one of the four domains, more specification options become selectable (these are optional).

The user now has the possibility to select the HF issue which should be considered in the system development process. Currently, MTTs for the



following HF issues are available: attention, behaviour, distraction, safety, situational awareness, task performance, usability and workload.

Moreover, the user can select regulations that have to be fulfilled by the developed system. These regulations are also highly domain specific. In the aeronautics domain, e.g., most regulations stem from the Federal Aviation Administration (FAA).

Last but not least, the user can choose the system development lifecycle phase in which the HF related MTT should be integrated.

These four selection criteria are represented on the first screen after the user login (see Figure 8).

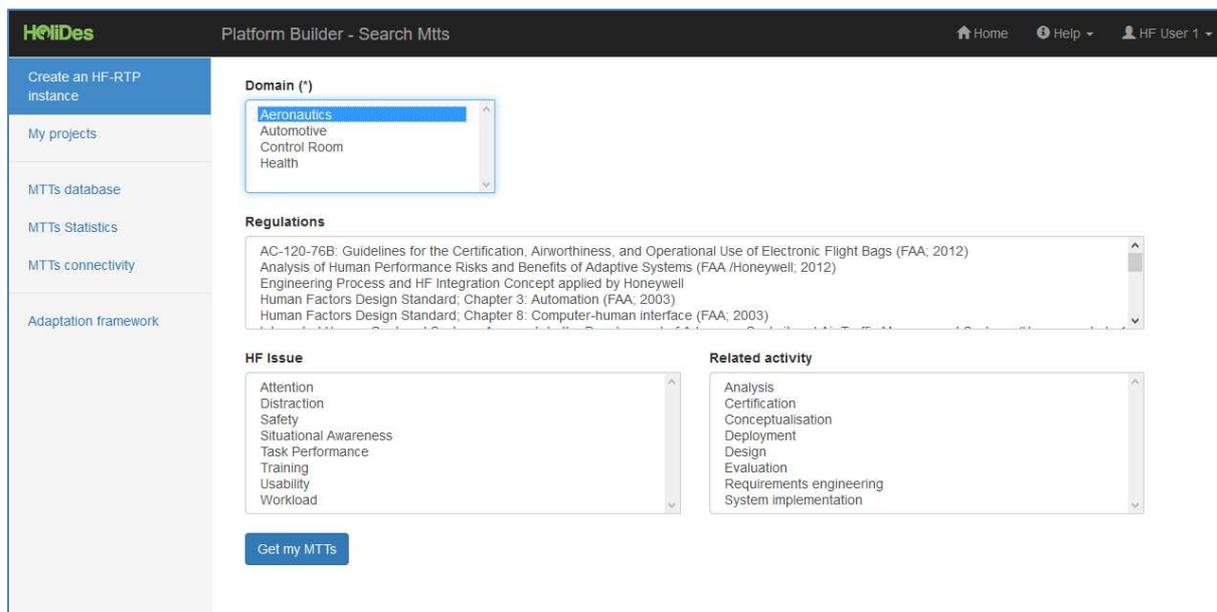


Figure 8: Search MTTs screen after user login

When the desired selections have been made, the user retrieves a list of appropriate MTTs by clicking on the button “Get my MTTs”.

### 5.4.2 Choosing the appropriate MTT

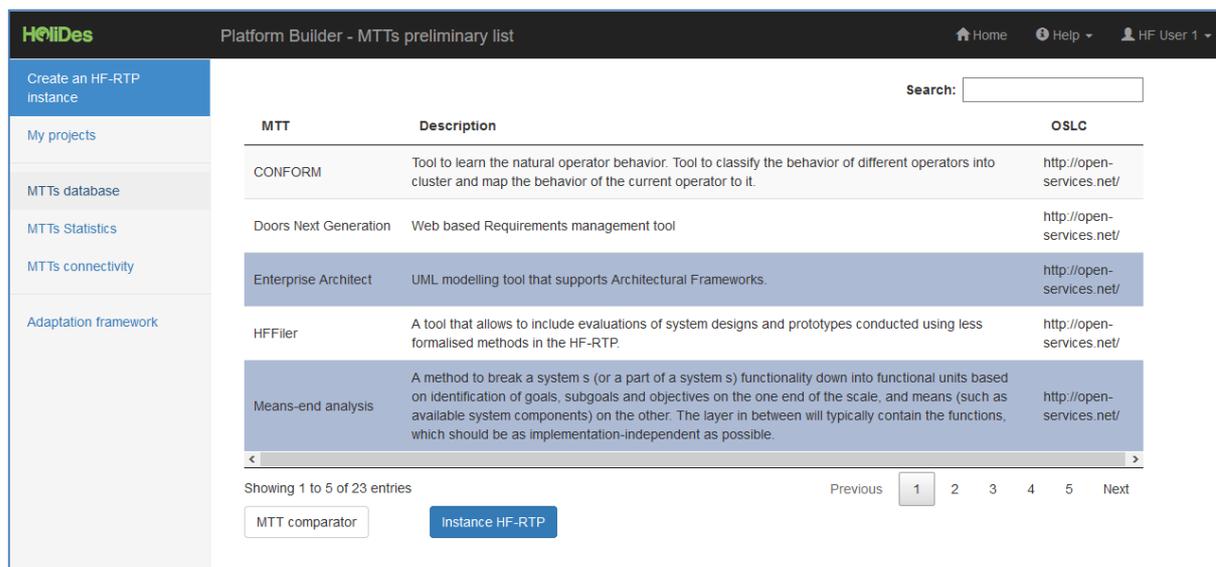
This screen proposes a list of preliminary MTTs that accomplish the criteria selection from the PB Input screen. This list contains per MTT

- the name of the MTT,
- a short description of the MTT,



- and the connector to the OSCL for each MTT.

Figure 9 shows an example of the “Preliminary MTT list” screen.



**Figure 9: Preliminary MTT list**

Moreover, the user has the option to select certain MTTs from the list for a detailed comparison. After the selection (just clicking on the MTTs that should be compared), the user clicks on the button “MTT comparator”. A pop-up appears in which the previously selected MTTs are now detailed (see Figure 10). Based on the detailed information on the proposed MTTs, the user can choose the MTT which fits best to his/ her purpose.



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

- Effort (time)

How much time is needed to apply the method?

This descriptor not only takes into account the actual time of method application, but also time that might be needed to adapt the method to the situation or domain it should be applied to and the time that is needed for analysis of results.

- Interpretation of Outcome

This descriptor specifies mandatory requirements for the interpretation of results.

- Measurement Source

What kind of data is recorded?

This descriptor differentiates whether data comes from subjective assessments or objective assessments.

*Subjective:* A user of a system (actor) fills out a questionnaire; an observer collects data by observing how a user performs a task.

*Objective:* Data is obtained by performance measures like reaction times or by psychophysiological measures like EEG, EDA or eye-tracking.

- Method applied by

Is expert knowledge mandatory to apply the method? One or two of three possible tags:

The descriptor determines what kind of knowledge level someone must have to apply the method and to appropriately interpret the gained data. Does the method require an *HF-expert*, a *domain expert* or can the method in principle be applied by everyone (*non-expert*).

- Resources

This descriptor contains a checklist defining the technical and human resources needed to apply the method.

- Time of Data Collection

When is the data obtained relative to the actual use of a system?

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

Methods can be either applied *prospective*, *real-time* or *retrospective*.

*Prospective* means that people never have interacted with the system they are going to evaluate, e.g. because the system does not exist right now or to ask for expectations people have of the system they are going to interact with.

*Real-time* data collection means that the data is obtained while someone is performing a task with the system in question.

*Retrospective* means that the data is collected after the user has interacted with the system. For example, questionnaires are often provided to users after they performed a task to evaluate workload or usability.

- Type of Empirical Method

In which way is the data collected?

Does the method require to conduct an *experiment*, does it need an expert to analyse the task (*expert inspection*), can data be collected by *observation*, in form of an *interview* or *questionnaires*?

It is also possible to find a specific MTT by filtering the preliminary list using the field "Search".

When the user has decided for an MTT, s/he has to select the MTT from the preliminary MTTs list (Figure 11). By clicking the button "Instance HF-RTP" the user then can instantiate the MTT for a specific project.



The screenshot shows the 'Platform Builder - MTTs preliminary list' interface. The top navigation bar includes 'Home', 'Help', and 'HF User 1'. A search bar is located at the top right. The main content area is a table with three columns: 'MTT', 'Description', and 'OSLC'. The table lists several MTTs, including ANaConDA, Data Race Detector, Search Bestie, HF-TA (Human Factors Task Analysis), and I-DEEP. The HF-TA row is highlighted in blue. Below the table, there is a pagination control showing 'Showing 6 to 10 of 23 entries' and a 'Previous' button. At the bottom, there are two buttons: 'MTT comparator' and 'Instance HF-RTP'.

MTT	Description	OSLC
ANaConDA	ANaConDA (Adaptable NATive-code CONcurrency-focused Dynamic Analysis) is a framework for creating dynamic analysers for multi-threaded C/C programs on the binary level. The framework provides a monitoring layer as well as noise injections techniques to increase chances to find concurrency-related errors in testing runs.	<a href="http://open-services.net/">http://open-services.net/</a>
Data Race Detector	The Race Detector	<a href="http://open-services.net/">http://open-services.net/</a>
Search Bestie	SearchBestie (Search-Based Testing Environment) is a generic infrastructure that is designed to provide environment for experimenting with applying search techniques in the field of program testing (e.g. to find optimal settings of injected noise to increase efficiency of AnaConDa and Race Detector	<a href="http://open-services.net/">http://open-services.net/</a>
HF-TA (Human Factors Task Analysis)	Toolkit that supports easy and (cost) efficient gathering of data for a hierarchical task analysis (HTA). This is enriched by interview techniques that can be used to elicit information regarding objective task characteristics, cognitive demands as well as human factors aspects associated with the task. For these purposes, it supports the following HF methods: Generic HTA, usability testing, card sorting technique, subject matter expert interviews, top level interviews (to gather knowledge of the experts bigger picture), Human Error Template Technique, and automatic time layering.	<a href="http://open-services.net/">http://open-services.net/</a>
I-DEEP	I-DEEP is a test automation manager on top of RTMaps. It provides a way to automate the execution of large sets of test cases based on RTMaps applications allowing to change parameters, or to go through a large list of recorded sessions.	<a href="http://open-services.net/">http://open-services.net/</a>

Figure 11: Selecting an MTT from the Preliminary MTT List

### 5.4.3 Creating a project

After the selection of an MTT the user can start to create a new project with the selected MTT as a first input or link the selected MTT to an already existing project (Figure 12).

If a new project is created, the user has to provide a project description, i.e., the name of the project. If the selected MTT is linked to an existing project, the user should leave this text field blank. In this case s/he has to click on the button "Link to project". A pop-up window appears to select the project from a list.

In both cases, it is mandatory to further provide a description of the phase in which the selected MTT should be applied (see Figure 12). After doing so, the user has to click on the button "Save Instance HF-RTP". A pop-up window informs about the successfully saved changes / project.

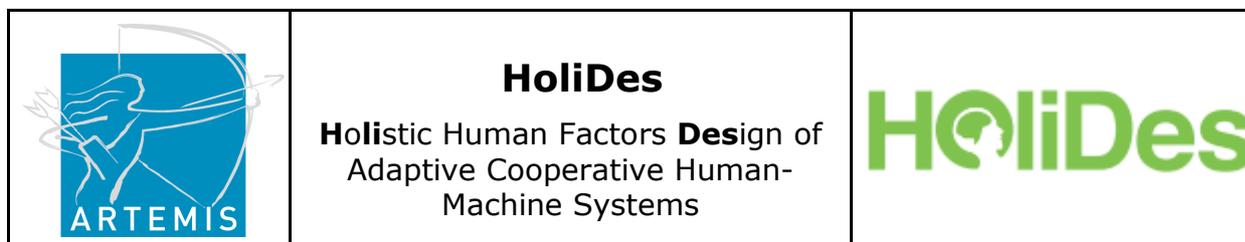


MTT	Description	OSLC
Doors Next Generation	Web based Requirements management tool	http://open-services.net/
HF-TA (Human Factors Task Analysis)	Toolkit that supports easy and (cost) efficient gathering of data for a hierarchical task analysis (HTA). This is enriched by interview techniques that can be used to elicit information regarding objective task characteristics, cognitive demands as well as human factors aspects associated with the task. For these purposes, it supports the following HF methods: Generic HTA, usability testing, card sorting technique, subject matter expert interviews, top level interviews (to gather knowledge of the experts bigger picture), Human Error Template Technique, and automatic time layering.	http://open-services.net/

**Figure 12: Creating a new project or linking a selected MTT to an existing project**

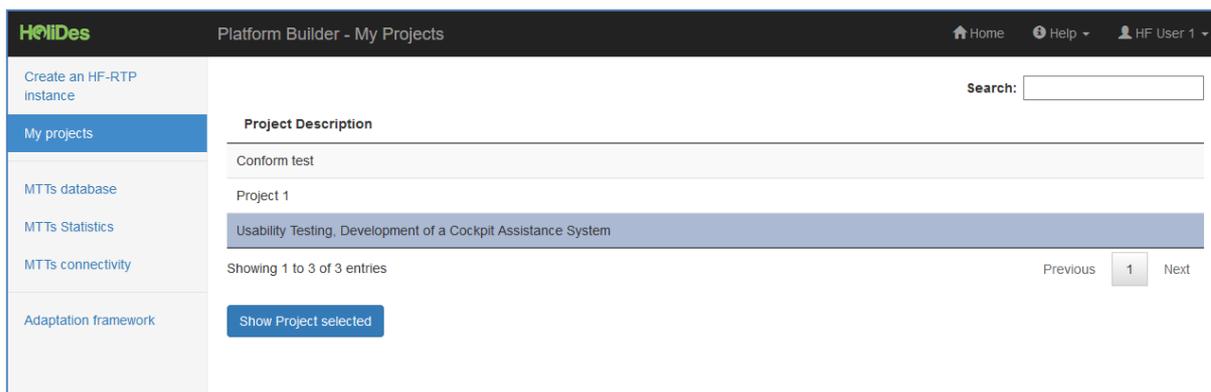
To select more MTTs for a project, the user has to click on “Create an HF-RTP instance” in the left main menu and then can start with the procedure from the beginning (see 5.4).

The left panel main menu also gives the possibility to access an existing project by clicking on “My Projects” explained in the next section.



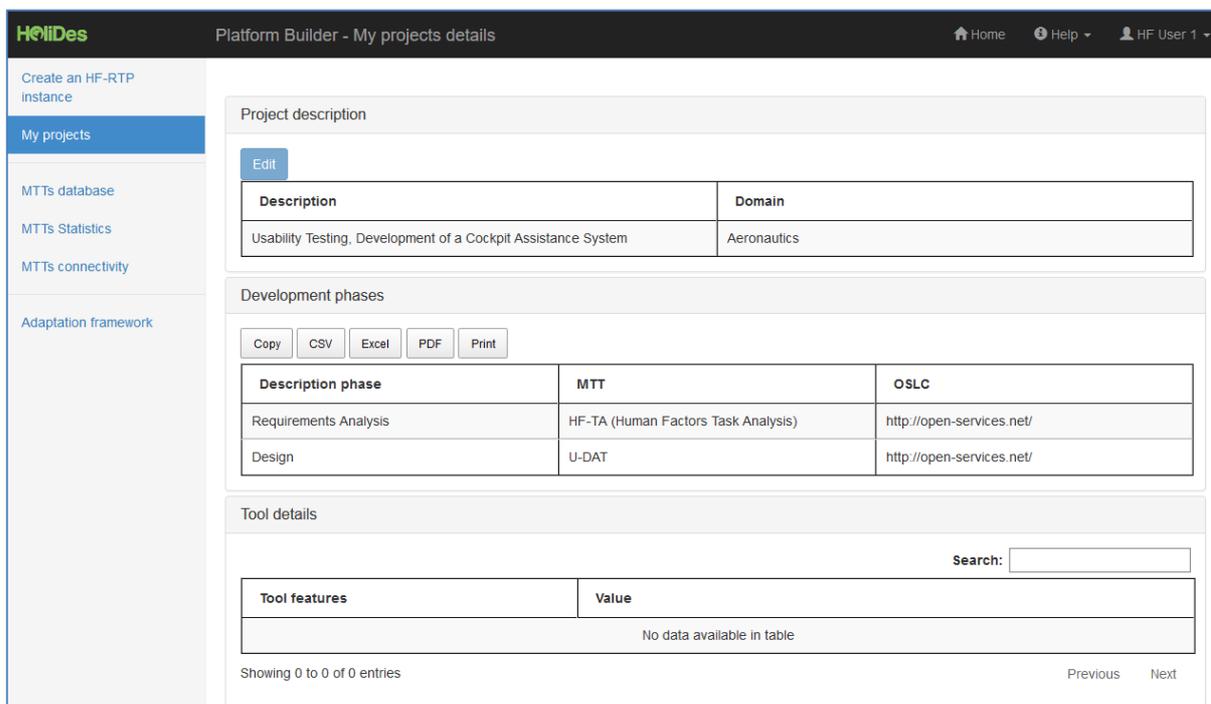
## 5.5 My Projects

This screen shows the projects saved by the logged-in user (Figure 13).



**Figure 13: List of projects available by clicking on "Get My Projects".**

A list of projects with the according descriptions is shown in a grid. Clicking on one of the projects and then clicking on "Show Project selected" redirects the user to the "My Project Details" screen for the selected project. This screen details the MTTs tool chain built (Figure 14).



**Figure 14: Viewing the project details**

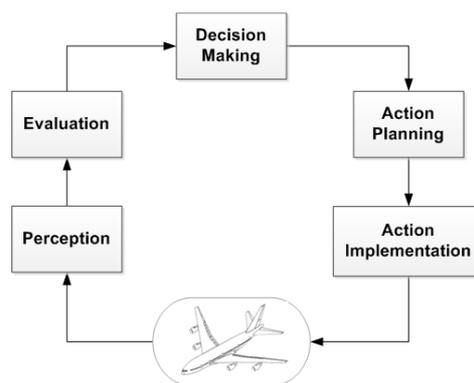
## 5.6 The Adaptation Framework

The adaptation framework is an extra functionality of the Platform Builder, which can be used to specify requirements of a system when adaptivity is a major issue. Therefore, the use of the adaptation framework is optional, but when in use, this should represent the first step in the MTT selection.

### 5.6.1 Control Loops as the constituting part of the Adaptation Framework

The adaptation framework (see D3.7 for full specification) is a generic method to design collaborative and adaptive systems. It is based on so-called primitive control loops that are combined with each other to build a complete AdCoS model.

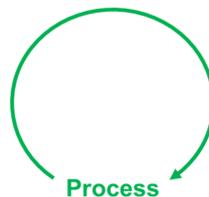
A primitive control loop is a sequence of operations aimed to control a process or an object. The following operations are executed every time the loop is started, to manipulate a certain object or process: Perception, evaluation, decision-making, action planning, and action implementation. These are also depicted in Figure 15.



**Figure 15: Basic control loop of the adaptation framework**

The objects, on which the loops are performed, represent “tasks”, “task distributions” or any kind of “process”.

Every control loop in the adaptation framework is represented as follows:



**Figure 16: Control loop to control a process**

To enhance this concept, we added the notion of “allocation”. Each control loop is managed by an “agent”. An agent could be human, a machine or the combination of several humans and machines. The control loop is then schematizing like that:



**Figure 17: Control loop used by an agent to control a process**

Each control loop acts in relation to a certain environment. Therefore, it has to be specified what exactly the loop controls. This is represented by the outcomes of the “action implementation” operation. This outcome may have several aspects, e.g. a display information on a screen, a sound, a vibration, human physical actions, ... . We define these outcomes as “User Interface”. The control loop is then depicted like this:

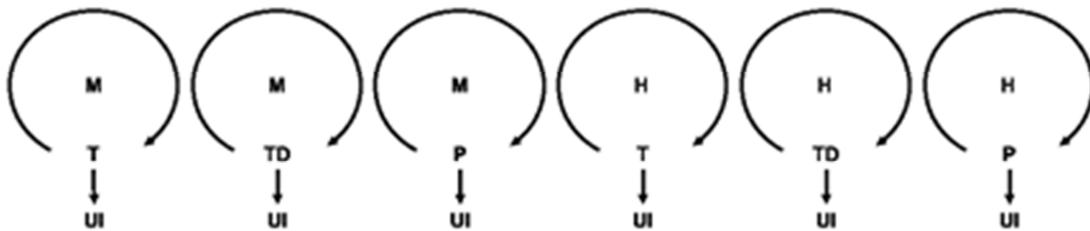


**Figure 18: Control loop used by an agent to control a process through a user interface**

Based on these attributes, a finite amount of primitive control loops results:

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

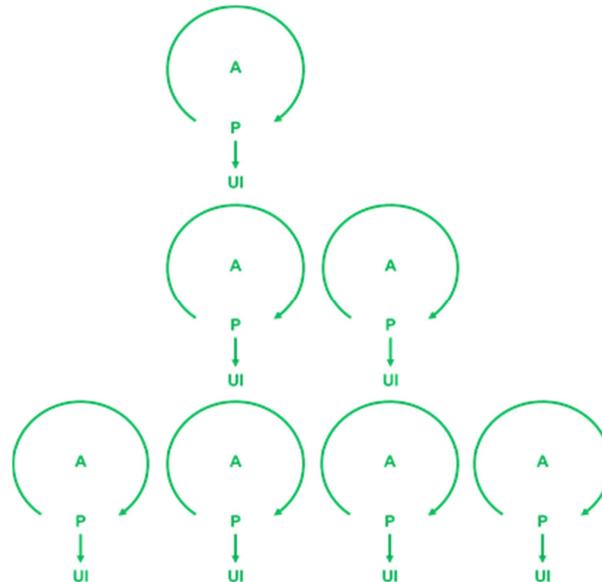
These are also depicted in Figure 19:



**Figure 19: Finite amount of primitive control loop (M: machine, H: human, T: task, TD, task distribution, P:process, UI: user interface)**

Based on this, it is possible to model any kind of AdCoS as a combination of primitive control loops.

An AdCoS is then defined as a tree like structure of primitive cognitive loops (Figure 20).



**Figure 20: Example of an AdCoS model with primitive control loop**

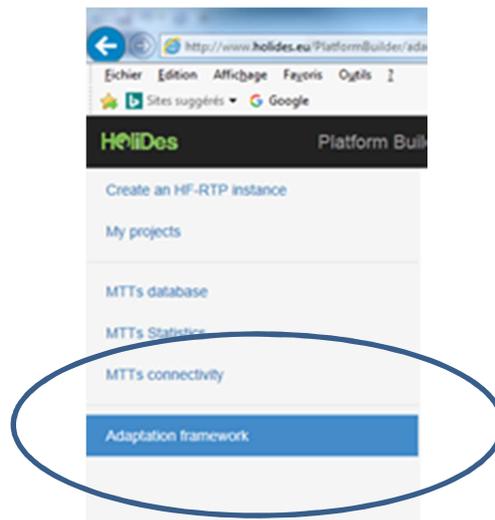
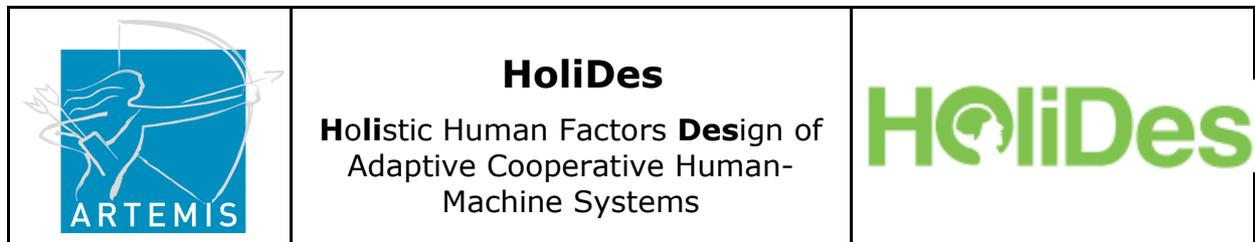
In sum, within the adaptation framework an AdCoS is represented as a combination of primitive control loops. Each control loop could be seen as a sub part of the AdCoS system and aims to manage a sub part of the global process. Formally, a control loop is then defined by:

- The **agent** that will control the loop
- The **process** on which the loop will be performed
- The **global process** of the AdCoS
- The **User Interface** on which the agent will act to manage the loop.

Depending on the nature of these variables (agent, process, global process, and user interface) the primitive loops come along with specific Human Factors requirements and Human Factor issues, which have to be considered in the design of an AdCoS. The detailed list of those requirements as a function of the aforementioned variables can be found in D3.7.

## 5.6.2 How to use the Adaptation Framework in the Platform Builder

In the Platform Builder, the adaptation framework can be visited by clicking on "Adaptation framework" in the left menu (Figure 21).



**Figure 21: Adaptation framework integrated in Platform Builder**

Once the user has clicked on the link s/he can between the following actions:

- Visualise some HoliDes AdCoS as example applications:
  - Patient positioning,
  - Diversion Airport,
  - Workload balancing,
  - Lane change,
- Design an own AdCoS using primitive control loops.

**Visualising existing AdCoS:**

To access example applications of the adaptation framework, the user only has to click on one of the following icons:



**Figure 22: Icons to run the adaptation framework on one of four Holides AdCoS**

After selecting one of the four AdCoS, the relative primitive control loops are shown together with accordingly pre-filled variable values (agent, process, global process, user interface). For the "Workload Balancing" use case, the example looks like this:

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



<input type="checkbox"/>	Loop	Agent	Process	Global Process		Interface
<input type="checkbox"/>	M-TD	"Display Advisor"	"Work Distribution"	"Border Control"	on	"Dashboard Interface"
<input type="checkbox"/>	H-TD	"Supervisor"	"Work Distribution"	"Border Control"	on	"Operators Display"
<input type="checkbox"/>	H-P	"Operators"	"Border Control"	"Border Control"	on	"Border Control Device"

**Figure 23: Example of primitive control loop for Workload balancing AdCoS**

With clicking on the button "Generate", the specific HF requirements and issues for this particular AdCoS are shown.

### Designing your own AdCoS:

Moreover, the user can model a new AdCoS by combining primitive control loops. In doing so, primitive loops are added and modified. First, the user has to select from a predefined set of options which primitives are going to be used:

- **H-T**: A **human** that closes a loop on a **task**
- **H-TD**: A **human** that closes a loop on a **task distribution**
- **H-P**: A **human** that closes a loop on a **process**
- **M-T**: A **machine** that closes a loop on a **task**
- **M-TD**: A **machine** that closes a loop on a **task distribution**
- **M-P**: A **machine** that closes a loop on a **process**

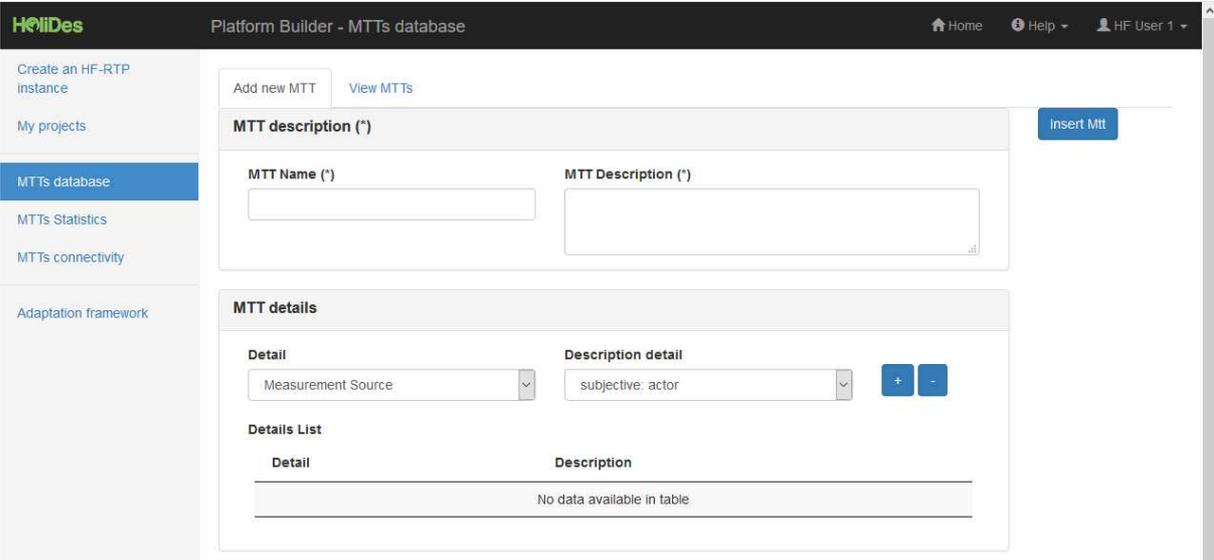
In a second step, each primitive control loop then has to be instantiated by defining the following three variables:

- Agent
- Object
- Process

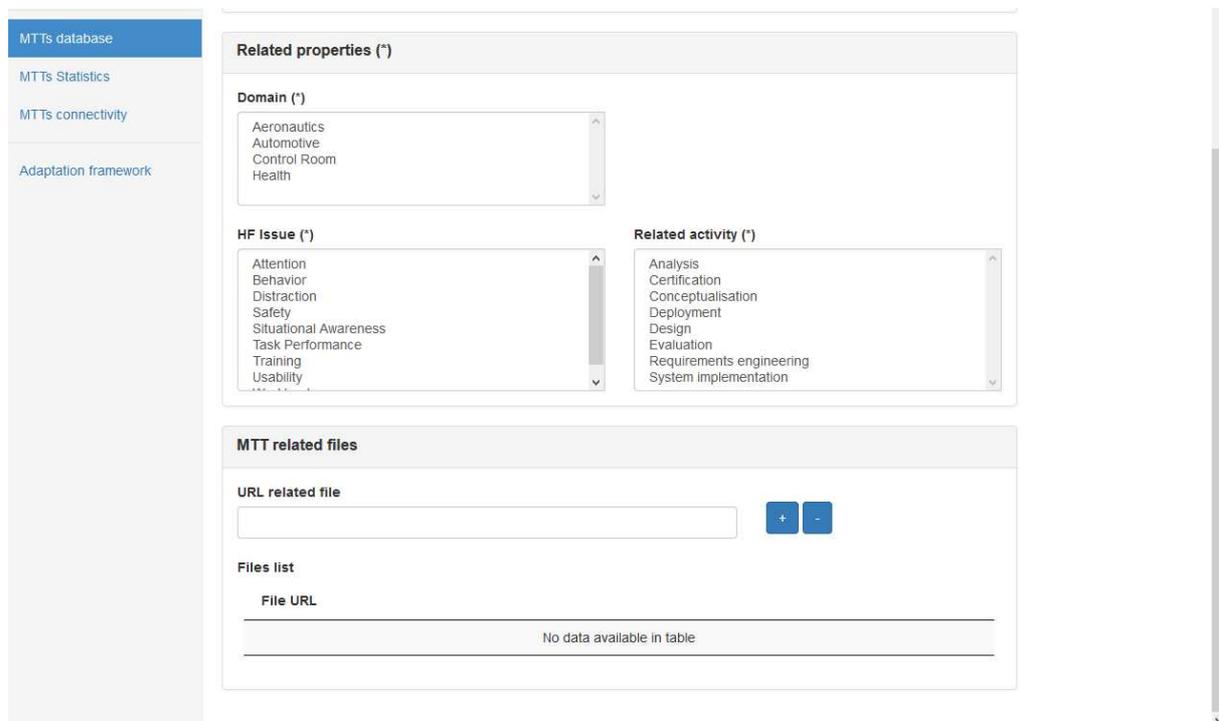
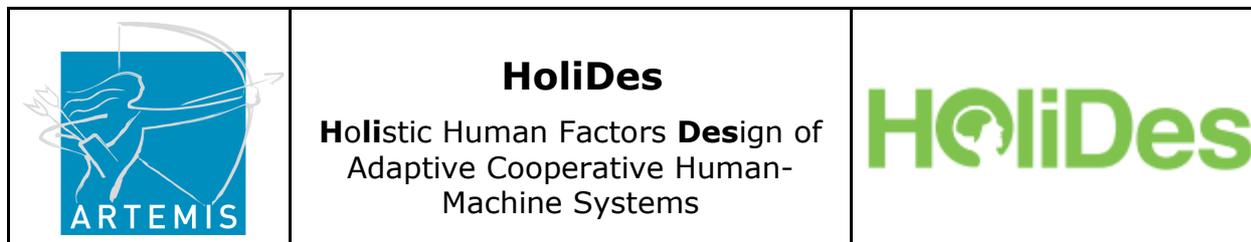


## 5.7 Explore or Manage MTTs: The Database

From the landing page as well as by selecting “MTTs database” from the navigation menu on the left side, the user can also explore or manage MTTs. The “MTT database” provides the possibility to include a new MTT into the Platform Builder (Figure 26, Figure 27). By so doing, the added MTT will also be available to the rest of PB app users.



**Figure 26: MTT Administration tool, upper part of the screen showing “MTT description” and “MTT details”**



**Figure 27: MTT Administration tool, lower part of the screen showing "related properties" and "MTT related files"**

The tab "Add new MTT" is divided into four sections:

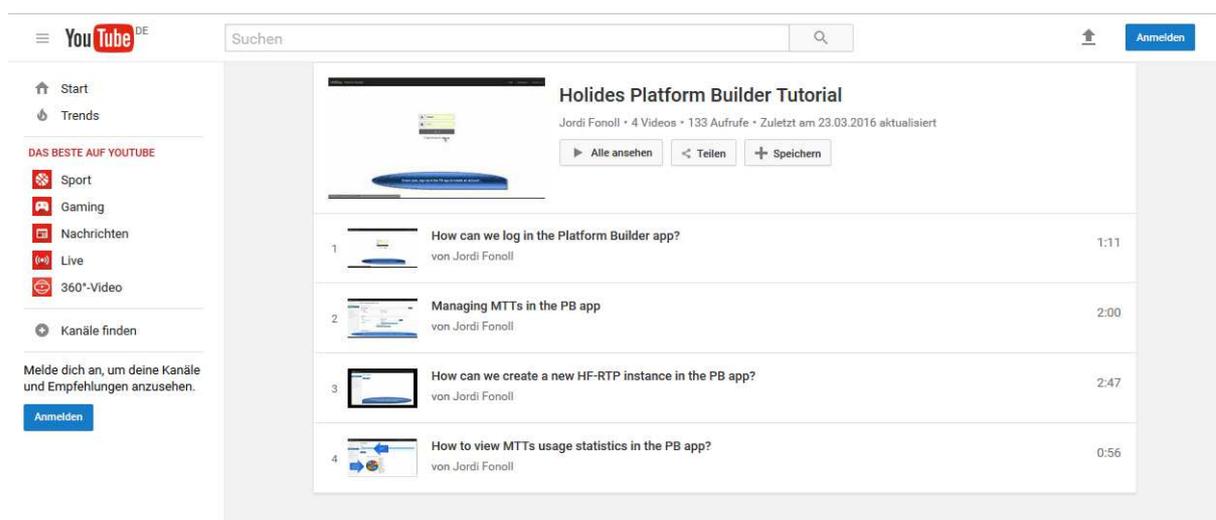
- MTT description: The user has to fill in the MTT name and a description of the MTT. This information is mandatory.
- MTT details: To describe the MTT comprehensively for other users, the MTT has to be linked to certain attributes. This can be done by selecting an attribute in the drop-down menu "Detail" and then specifying the attribute's characteristic from a predefined list using the second drop-down menu "Description detail".
- Related properties: The user has to select the criteria conditions (domain, HF Issue, extra requirement and related activity) to include in the MTT search criteria.
- MTT related files: In the last step, the user can link URL related files, like information on the MTT's author or links to companies which provide the tool.

Once all the required data has been provided, the user clicks on "Insert MTT" and all the information will be stored in the PB database.

The tab “View MTTs” provides the possibility to get all the MTTs included in the PB. To find a specific MTT, the user can use the “Search” field located in the top right button.

## 5.8 The Platform Builder and the HF-RTP: Information and How-to

For further information on the Platform Builder and the HF-RTP, the user can click on the according button on the landing page and is redirected to youtube where (s)he finds additional information and tutorials on the use of the Platform Builder (Figure 28).



**Figure 28: Youtube tutorial videos on the Platform Builder.**

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

## 6 Tailoring the HF-RTP to your purpose

There is a subtle but important difference between a tool being classed as part of the HF-RTP and tailored in the HF-RTP. To say that a tool is part of the HF-RTP represents a commitment to say that a tool should share resources in compliance with the HoliDes IOS. (I.e. tools will share data using RDF XML which is compliant to a HoliDes Meta Model and made available through a Restful interface and linked data).

To say that a tool is tailored in the HoliDes HF-RTP you are making a commitment to that two tools are exchanging information in accordance to the IOS. One tool is making its data available as RDF XML and is considered to be an OSLC provider. The second tool is asking for the data and is considered to be a consumer. In essence if a tool is tailored it 'understands' how to use that data. In this understanding, tailoring is about selecting tools and services and integrating them by providing suitable ways to exchange data between them. The result of a tailoring process is thus an instance of the HF-RTP addressing the specific needs of the project.

However the integration of Human Factors into the development processes as well as requirements related to adaptivity show a need for more adapted and more specific description of the tailoring steps. Human-Factors-related activities in many cases are not clearly formalized and might rely on informal methods with unclear specification how results should be presented. Activities connected to certification needs defined in application domains sometimes use "common practice" methods rather than specific tools to fulfil certification requirements. These practices may include, for example, the Code of Practice (automotive) or methods derived from User Centred Design (medical). Again, in these domains regulations are rather unspecific concerning HF activities. Many concepts and regulations provide guidelines on issues or requirements to be met. But most of the concepts and regulations are unclear about methods or tools that shall be used. Subsequently the level of formalization of results of HF activities is low. Data formats are left open.

To be able to integrate informal methods into the HF-RTP as well as with the supporting tool-chain, we have defined a Human Factors method

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

library describing common practices of used methods (see D1.5 for more information on that). This method library is the basis of the Platform Builder. The formalization of HF-related methods is done by using a predefined set of descriptors which are visible to the Platform Builder user as the selection criteria (Search MTTs screen and PB tool comparator).

Now, to tailor an MTT to your purpose, four successive tailoring steps should be accomplished. These are detailed in the following, always accompanied with a concrete example.

## **6.1 Tailoring Step 1: Identification of issues in the existing development process**

The first step is to provide a description of the overall purpose of the HF-RTP instance within the development process to be supported. This describes the specific workflow of the development process of the system to be developed as well as the issues faced by the developers, mainly in terms of Human Factors or adaptivity. Such issues might include a lack of traceability of engineering data or issues using information from one tool in another.

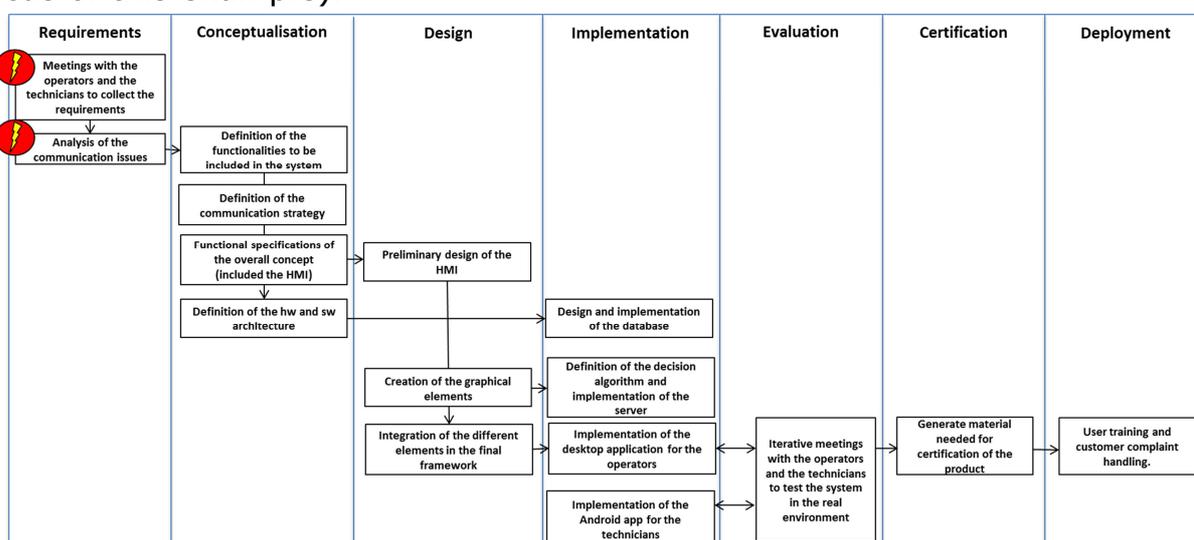
As far as the description of workflows is concerned, the tailoring process uses parts of the regular engineering process. We recommend to relate those workflows to the V-model phases described earlier in this handbook as those are also part of the tailoring process when using the Platform Builder.

### **6.1.1 Example from the Control Room Domain**

In this specific example, we focus on a control room that collects energy emergency requests and assigns the interventions to the technicians in the field.

At present, the communication between the Control Room operators and the operative teams takes place only via phone calls (very time-demanding) and the allocation of tasks and responsibilities is based on the senior experience of Control Room operators. The aim of a new system is to automatically allocate tasks and responsibilities to the operative teams.

However, previous systems (e.g. mobile apps) that had been introduced were always rejected by the operators and/or the technicians. So far, the reason for this rejection is not clear because no specific methodology has been applied in the requirements phase for the analysis of the communication between the operators and the technicians in the field. This problem is right at the beginning of the system development and is likely to affect the entire process. Figure 29 shows the traditional system development process and highlights the actions that have probably not been specific enough for a successful development (the later steps are not focus of this example).



**Figure 29: Identification of issues in the existing development process**

## 6.2 Tailoring Step 2: Selection of methods, techniques and tools

In the second step, related methods, techniques and tools are selected to fit the needs of the project purpose. A detailed rationale should be given for each of the selected MTTs which has to be included in the workflow. In doing so, the traceability of the project can be assured which can become an important issue, e.g. for later certification purposes of the system.

The MTT selection is supported by the Platform Builder. Based on the definition of the overall purpose of the HF-RTP instance and the specific

workflow of the development process you can now set the selection criteria accordingly and choose the MTT from the preliminary results list which fits best to your needs.

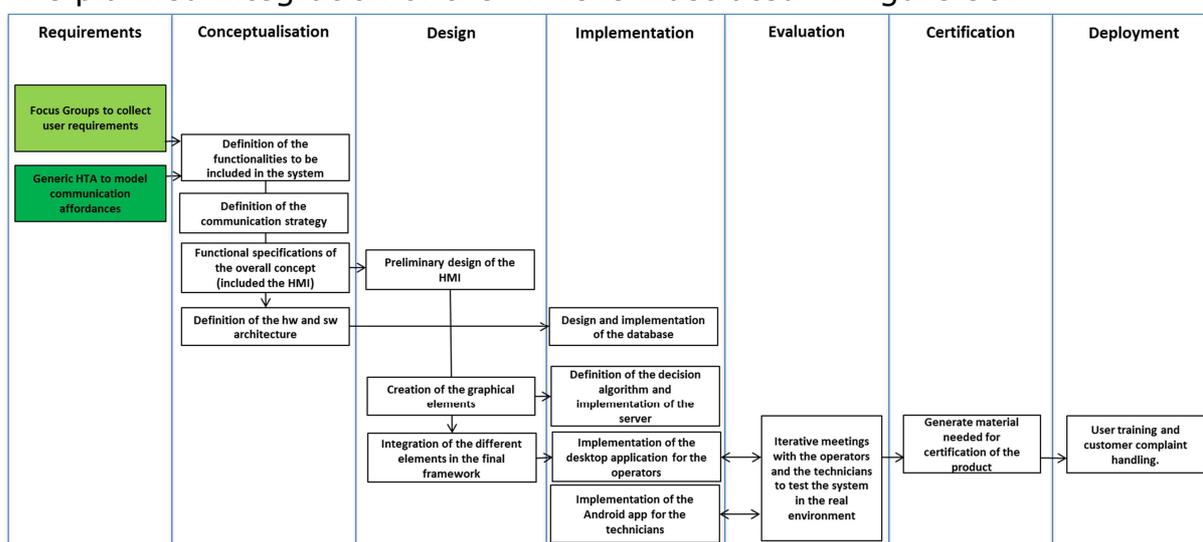
### 6.2.1 Continued Example from the Control Room Domain

By using the Platform Builder, the developers have decided to integrate the following MTTs in their system development process: Focus Groups and a generic Hierarchical Task Analysis (HTA).

The Focus Group offers a more structured approach to gather user requirements regarding the future system. The requirements should then guide the system design.

The application of a generic HTA will support the decomposition of the task itself in smaller hierarchically composed units like goals, sub-goals, plans. This will provide system developers with a more detailed understanding of the operators' task and will help to design a more appropriate system.

The planned integration of the MTTs is illustrated in Figure 30.



**Figure 30: Integration of MTTs into the development process**

### 6.3 Tailoring Step 3: Describe what information needs to be exchanged between MTTs

The next step is to describe the information exchange needed between the methods, techniques and tools in the actual tool chain. You need to decide which information (on a semantic level) should be available for other tools later in your system development process. For example, it could be relevant for designers to know the results from a requirements analysis, which was made at the beginning of the project to understand why the designers have to fulfil certain specifications in the system design. So again, the linking of information between data should support the traceability within the project and make the entire process more transparent. This, in turn, should prevent unintended redundant task execution or the loss of information during the project.

A common semantic approach allows the generalization of the information linkage across several methods and tools. If a tool is integrated to this common semantic representation it is integrated to all other methods, tools and services. In the HF-RTP,

this semantic approach is represented by the Common Meta Model together with the Human-Factors ontology.

#### Want more information?

##### *The Common Meta Model*

During the development process of a system, different tools and techniques can be applied. These tools and techniques have different underlying models. To allow data exchange between different tools, these aspects need to be synchronized. Therefore, equal concepts – which have the same semantic in different models but using a different wording – are identified and a common element in the Common Meta Model is created.

##### *The HF Ontology*

The HF Ontology defines the interoperability standard to link information related with Human Factors.

It is an OWL and more detailed formalization of the subset of concepts in the CMM related to Human Factors. The purpose of the ontology is twofold. On the one hand the HF ontology is used to define the common vocabulary to be used when **linking information** between MTTs related to HF issues. On the other hand, it contains information that is used by the Platform Builder **to assist the end user** in the process of selecting appropriate HF methods and tools to evaluate a given HF issue, or to select certifications that apply to a certain domain or HF issue.

## 6.4 Tailoring Step 4: Implementation of MTT mappings

Step 4 is the implementation of the mappings defined in step 3, the implementation of parsers to allow the tools to correctly interpret the information received according to a predefined communication protocol.

The actual implementation of the communication protocols and parsers for the sharing of data among tools and with the RTP relies on the definition of the information to be shared.

The Platform Builder supports this step by providing information if an MTT is IOS compliant and accordingly integrated into the HF-RTP. The information can be found in the Preliminary MTT List in the last column.

How two tools are actually implemented depends entirely on the specific tools, how they are set up to share data and how the implementer decides to perform the integration. If one decides to extend a tool by using its native API then you are constrained by the programming language of that API. For example, if your tool has a C++ API then it makes no sense to develop your extension in C#.

Whichever implementation you choose, you will end up with essentially the same thing, some sort of REST-ful interface which other REST-ful application can interact with. In the diagrams below, this is shown as an OSLC gateway. Explaining how to build a REST-ful interface is beyond the scope of this document but it's a topic which many tutorials can be found on the web.

In general, there are three approaches to implement the MTT mappings, shown in Table 2:

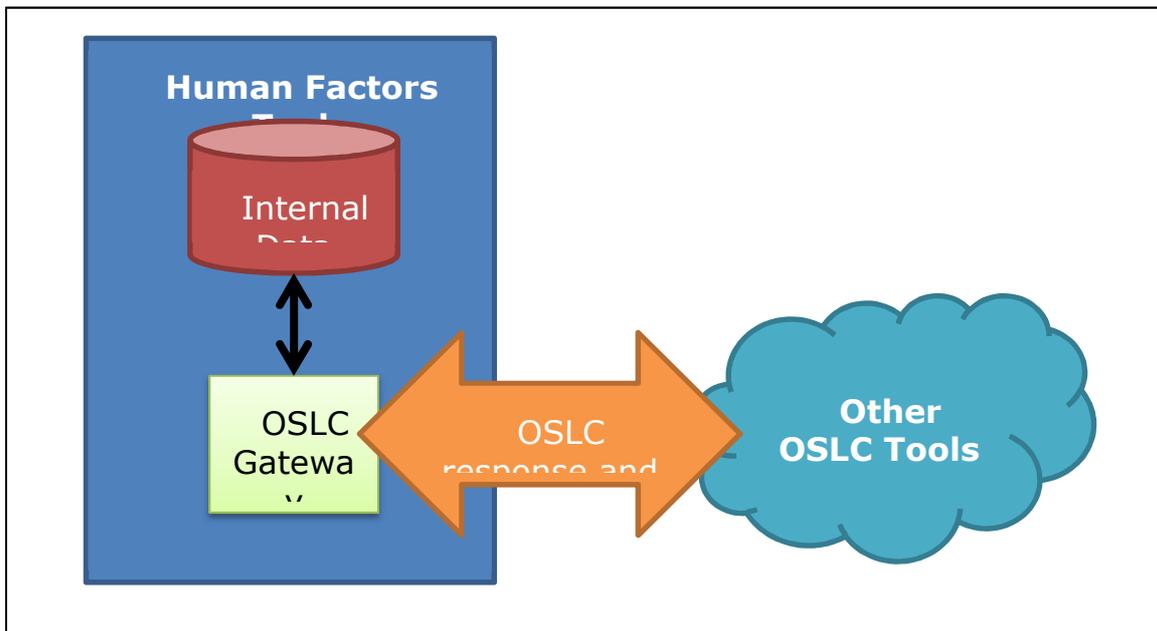
**Table 2: Tailoring Step 4 - three approaches to implement MTT mappings**

	<b>Description</b>	<b>When to use.</b>
1	Build in OSLC support natively	When you are the tool owner and have full ability to modify the application.
2	Extend tool to support OSLC with API	When you don't own the tool, but have access to the API to extend it.
3	Build a wrapper application which supports OSLC	When you don't own the tool and don't have an API to work with.

The different approaches are explained in the following.

### 6.4.1 Build in OSLC support natively

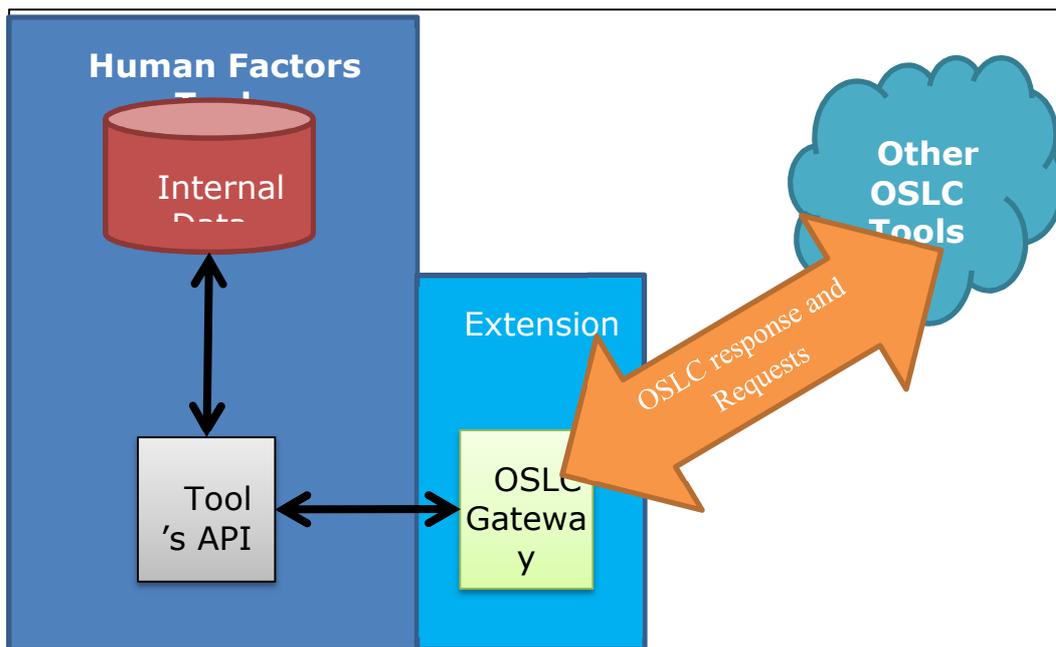
This is the best option when you have access to the tool's source code. (See Figure 31) This route assumes you are the tool owner and have total visibility to extend the tool from the inside. Whichever programming language you use to develop the tool is probably the one you should use to create your OSLC gateway.



**Figure 31: OSLC Support built in natively**

### 6.4.2 Extend tool to support OSLC with API

Figure 32 shows a diagram which links a 3<sup>rd</sup> party tool with other tools in the tool chain. This scenario assumes that the MTT in question is a black box with the inner workings accessible through a provided application programming interface (API). In this scenario, the API takes care of accessing the internal workings of the tool's data and the extension makes those resources available through OSLC. Once again, if your API is written in Java, for example, it makes no sense for you to pursue an extension in C++ or some other language. Once the gateway is working, other OSLC tools can send and receives commands, to and from the extension.



**Figure 32: Extending a 3rd party tool**

### 6.4.3 Extend tool to support OSLC without an API

There might be a situation where using an API to extend a tool's functionality is not possible. This could be due to extra licensing costs to access to the API or that the API does not exist. In such cases, some sort of 'wrapper' application would need to be written in order to make the internal data of the tool available as OSLC resources. This could be implemented via a file export which is then picked up by the wrapper. The wrapper application implementation then makes the resources available in the file export to the network via the gateway (Figure 33).

**Note:** this is the least desirable implementation for many reasons, in particular because, as soon as the data is changed from inside the tool, your file export is out of date. It should be recognised that an implementation of this sort is only for demonstration purposes. In an ideal world, all tool vendors would implement option 1.

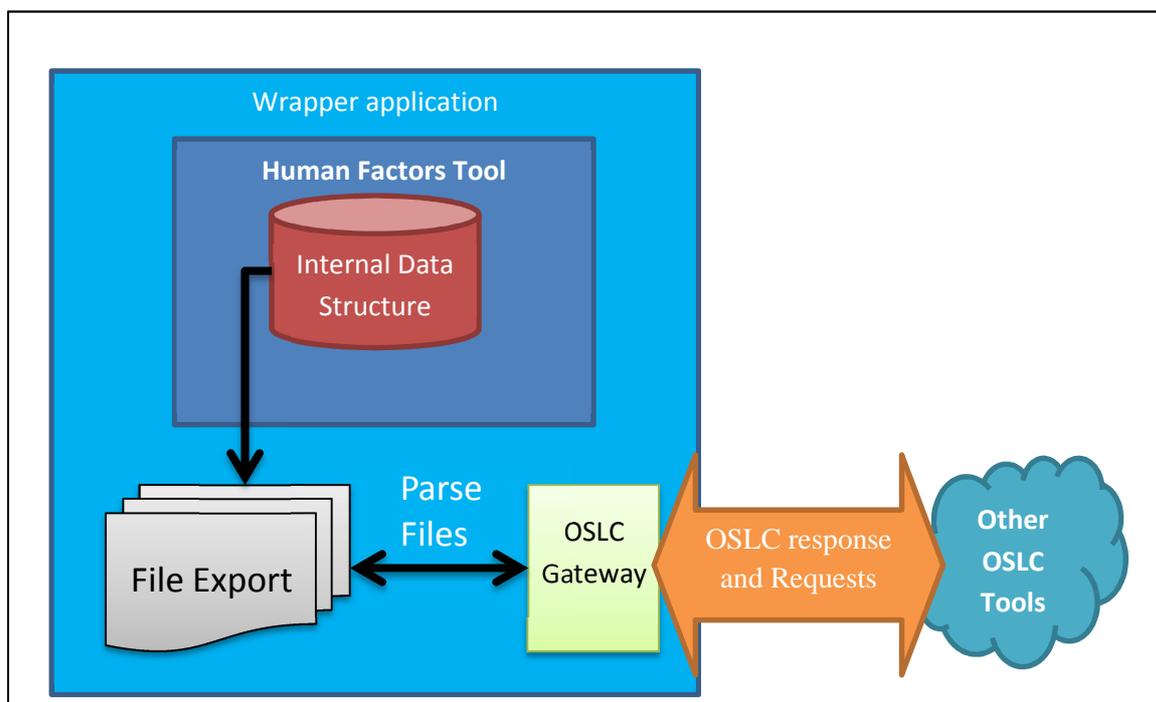
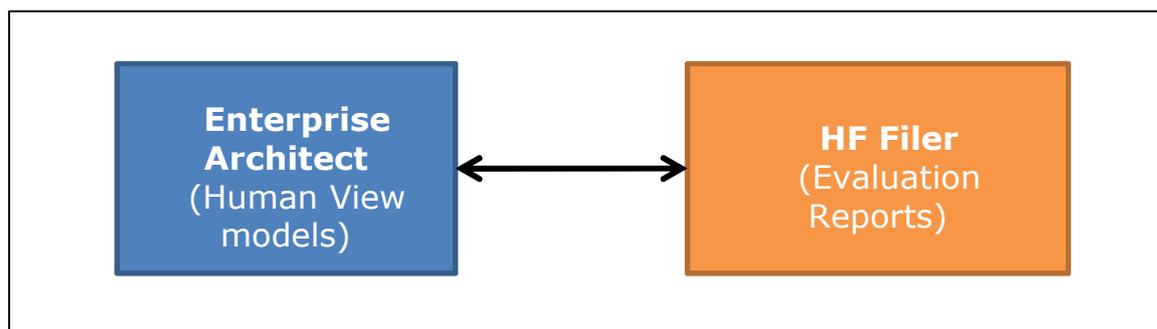


Figure 33: Extending a 3rd party tool without an API

#### 6.4.4 Example: Connecting two tools

The following scenario demonstrates how two tools in HoliDes, HF Filer and Enterprise Architect, were linked together using a mixture of option 1 and option 2.



##### *HF Filer Tool*

In one of the AdCoS system development processes (WP8) the need was identified to connect the HF-Filer with a modelling tool, Enterprise Architect. The HF-Filer application is a web based documentation tool for the tracking and managing of evaluation reports. Enterprise Architect is a modelling tool in which Airbus is using to model human factors. The desire is to be able to browse for resources in the HF-Filer from Enterprise Architect and link them to modelling elements in the human view.

Since HF-Filer is an application developed by Anywi who are in the HoliDes consortium and the application is being made from the ground up, it makes perfect sense that OSLC functionality is built in from the start.

HF-Filer was built in Java which meant it could take advantage of the Lyo libraries which have been created to speed up the development of creating an OSLC application. To get an understanding of what's required to create an OSLC application, AnyWi studied the OSLC4J Bugzilla Adapter. <http://open-services.net/software/oslc4j-bugzilla-adapter/>



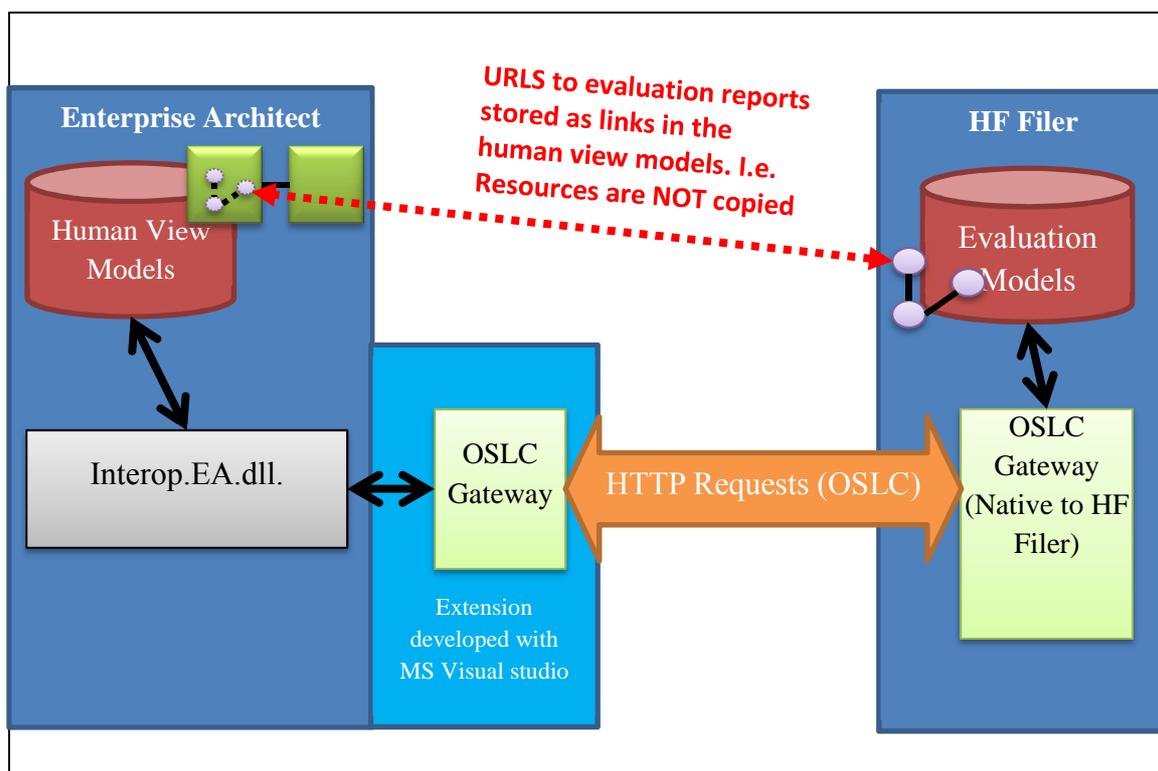
**Figure 34: The Lyo Project**

### Enterprise Architect

Enterprise Architect (EA) is a modelling tool made by a company called Sparx. Sparx are not in the HoliDes project but the tool is used by Airbus who is. EA has a C# API which allows a software developer to create an extension using a tool like Microsoft Visual Studio. This is done by loading the *dll* file into your visual studio project called **Interop.EA.dll**.

With the interop file installed, it was then possible to access the resources inside EA. All that was left to do is to write an application in C# which made calls to the HF Filer and then embed the links to the evaluation reports inside the EA artefacts.

Figure 35 shows the layout of the HF Filer and EA integration for link evaluation reports to human view data.



**Figure 35: The EA/HF Filer integration**

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

## 7 Summary

The objective of the HF-RTP Handbook is to support system developers in deploying the HF-RTP for their purpose. Therefore, the handbook describes first the concept of an RTP before broaching the issue of Human Factors. The HF-RTP can be consulted in different steps of the system development using the Platform Builder. This is a web application and the front end of the HF-RTP, i.e. what becomes visible for the user. It is aimed at supporting the user to solve a concrete HF issue through an RTP instantiation. This RTP instantiation is the result of an overall process of selecting and integrating tools into a tool chain and represents the final output of applying the HF-RTP to a concrete system development.

## 8 References

- [1] CESAR - <http://www.cesarproject.eu/>
- [2] MBAT - <https://www.mbat-artemis.eu/home/>
- [3] Crystal Project - <http://www.crystal-artemis.eu/>
- [4] Boy, G. (2011). A Human-Centered Design Approach. In: G.Boy (Ed.) *The Handbook of Human-Machine Interaction*. Farnham: Ashgate, pp. 1-22.
- [5] Mayhew, D.J. (1999) *The Usability Engineering Lifecycle – A practitioners Handbook for User Interface Design*. San Francisco: Morgan Kaufmann.
- [6] Wickens, C.D.; Lee, J.D.; Liu, Y & Becker, S.E.G. (2004). *An Introduction to Human Factors Engineering*, 2<sup>nd</sup> Edition. Upper Saddle River: Pearson, Prentice Hall.
- [7] Nielsen, J. (1999) *Designing Web Usability: The Practice of Simplicity*. New Riders Publisher.
- [8] Card, S. K., Moran, T. P., & Newell, A. (1983). *The psychology of human-computer interaction*. Hillsdale, NJ: Lawrence Erlbaum Associates.
- [9] Endsley, M. R. (1993). A survey of situation awareness requirements in air-to-air combat fighters. *International Journal of Aviation Psychology*, 3(2), 157-168.
- [10] Prümper, J. (1997). Der Benutzungsfragebogen ISONORM 9241/10: Ergebnisse zur Reliabilität und Validität. In *Software-Ergonomie'97*. Vieweg+ Teubner Verlag, 253-262.
- [11] Botta, M. U., Borchers, S. T., Curio, C. T., Collina, S. S., Gardas-Schmidt, D. H., Gründl, M. E., Guidotti, L., Herout, A., Ihme, K., &

Käthner, D. (2013). *D5. 2-Plan for Integration of Empirical Analysis Techniques and Tools into the HF-RTP and Methodology*. Published online: [http://www.holides.eu/sites/default/files/holides/files/content-files/deliverables/D5.2\\_IntegrationPlan\\_Empirical\\_Analysis\\_Techniques.pdf](http://www.holides.eu/sites/default/files/holides/files/content-files/deliverables/D5.2_IntegrationPlan_Empirical_Analysis_Techniques.pdf), retrieved on 14 July 2015.

[12] Hart, S. G. & Staveland, L. E. (1988). Development of NASA-TLX (Task Load Index): Results of empirical and theoretical research. In P. A. Hancock and N. Meshkati (Eds.) *Human mental workload*. Amsterdam: North Holland Press.

[13] Roscoe, A.H (1984). Assessing pilot workload in flight. In *Conference Proceedings No. 373. Flight Test Techniques*. AGARD, Paris.

[14] Endsley, M. R. (1988). Situation awareness global assessment technique (SAGAT). In *Aerospace and Electronics Conference, 1988. NAECON 1988, Proceedings of the IEEE 1988 National*, 789-795.

[15] Taylor, R. M. (1990). Situation awareness rating technique (SART): The development of a tool for aircrew systems design. In *Situational Awareness in Aerospace Operations*. France: Neuilly sur-Seine, NATO-AGARD-CP-478.

[16] Nielsen, J., & Molich, R. (1990). Heuristic evaluation of user interfaces. In *Proceedings of the SIGCHI conference on Human factors in computing systems*. ACM, 249-256.

[17] Lewis, C., Polson, P., Wharton, C., & Rieman, J. (1990). Testing a walkthrough methodology for theory-based design of walk-up-and-use interfaces. In *Proceedings of CHI, 1990*. Seattle, WA, ACM, New York, 235-242.