Citation Notice

[1] Maximilian Laiacker, Andreas Klöckner, Konstantin Kondak, Marc Schwarzbach, Gertjan Looye, Dominik Sommer, and Ingo Kossyk. Modular scalable system for operation and testing of UAVs. In *American Control Conference*, pages 1460–1465, Washington, DC, 17-19 June 2013. American Automatic Control Council (AACC), IEEE. ISBN: 978-1-4799-0176-0. ISSN 0743-1619. URL: http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=6580042.

```
% This file was created with JabRef 2.9.2.
% Encoding: Cp1252
@INPROCEEDINGS{laiacker2013modular,
  author = {Maximilian Laiacker and Andreas Kl\"ockner and Kon{\-}stan{\-}tin
        Kon{\-}dak and Marc Schwarz{\-}bach and Gertjan Looye and Dominik
        Sommer and Ingo Kossyk},
  title = {Modular scalable system for operation and testing of {UAVs}},
  booktitle = {American Control Conference},
  year = {2013},
  pages = {1460-1465},
  address = {Washington, DC},
  month = {17-19 June},
  organization = {American Automatic Control Council (AACC)},
  publisher = {IEEE},
  note = {ISBN: 978-1-4799-0176-0. ISSN 0743-1619.},
  abstract = {In this paper we present a system for operation and testing of different
        UAVs. The system allows easy development and modification of control
        and mission software. The system is composed of hard- and software
        modules with a standardized interface. We have been using the system
        with rotary and fixed wing UAVs with a take-off mass between 10 and
        100 kg. For larger platforms the system can be used in a redundant
        setup. The software modules are integrated in a special real-time
        framework, which supports execution, scheduling, communication and
        system monitoring. A modular simulation and control infrastructure
        allows for flexible, integrated design and analysis of control laws.
        The code for the computational part of the modules can be generated
        from Matlab/Simulink-models or from Modelica-models. The system supports
        debugging, soft- and hardware in the loop simulations, operator training
        as well as real flight experiments. The main design concepts are
        explained at hand of our solar powered high altitude platform ELHASPA
        and the 10 years experience in development and operation will be
        summarized.},
  url = {http://ieeexplore.ieee.org/xpl/articleDetails.jsp?arnumber=6580042}
}
```

Modular scalable system for operation and testing of UAVs

Maximilian Laiacker, Andreas Klöckner, Konstantin Kondak, Marc Schwarzbach, Gertjan Looye, Dominik Sommer and Ingo Kossyk

German Aerospace Center (DLR), Robotics and Mechatronics Center, 82230 Wessling, Germany

***Abstract*— In this paper we present a system for operation and testing of different UAVs. The system allows easy development and modification of control and mission software. The system is composed of hard- and software modules with a standardized interface. We have been using the system with rotary and fixed wing UAVs with a take-off mass between 10 and 100 kg. For larger platforms the system can be used in a redundant setup. The software modules are integrated in a special real-time framework, which supports execution, scheduling, communication and system monitoring. A modular simulation and control infrastructure allows for flexible, integrated design and analysis of control laws. The code for the computational part of the modules can be generated from Matlab/Simulink-models or from Modelica-models. The system supports debugging, soft- and hardware in the loop simulations, operator training as well as real flight experiments. The main design concepts are explained at hand of our solar powered high altitude platform ELHASPA and the 10 years experience in development and operation will be summarized.**

I. INTRODUCTION

In recent years, UAV-related research has been extended to new subjects and applications. Numerous new airborne platforms have been created for special application requirements. The platforms differ in scale as well as in aerodynamic configurations.

The conversion of an experimental system into an operational one requires substantial efforts and can be problematic due to changes in the software and/or hardware. Therefore, it is important to have technologies and tools supporting for this transformation.

In this paper we present a modular scalable system, which simplifies the development of UAS, the experimental work and can be used as an operational system. The core of the system is composed of soft- and hardware modules which can be combined in order to get a setup for a particular platform and for a particular class of applications. The system can be used with or without a double redundancy and has interfaces for different tools like Matlab/Simulink and Modelica allowing the usage of these tools for development, simulation, automatic code generation and debuging. The system has been used with different helicopter and fixed-wing platforms with a take-off mass between 10 and 100 kg.

In Sec. II the main hardware modules of the system are presented. The explanation of the system is based on the example of our solar powered high altitude platform ELHASPA where the system is used with double redundancy which we were able to develop with minimal efford based on our existig experimental setup. In Sec. III, the software framework is presented and the main software modules of the system are explained. In Sec. IV, the tools and approaches for multi-physics simulation and for controller design are presented. The algorithms described in this section are used for testing within the framework described earlier. In Sec. V, we give more information on the platforms and flight experiments conducted using the presented systems and summarize our experience in developing and testing UAV/UAS. In Sec. VI, conclusions are made.

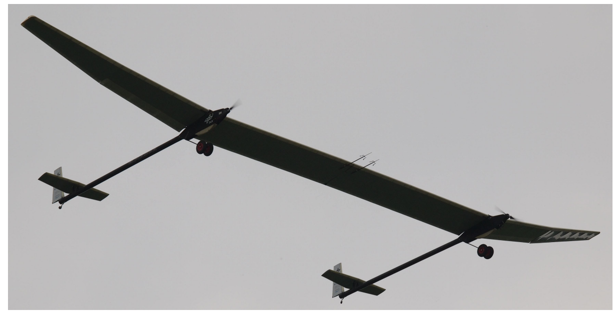

Fig. 1. Solar HALE UAV ELHASPA in flight

II. SYSTEM OVERVIEW

We will now describe how we modified our experimental system to a system that can be used operational on a high altitute long endurance (HALE) platform. We decided to implement a double redundant setup for our high altitude platform ELHASPA seen in Fig. 1. The main idea of the developed redundancy concept is the fulfillment of the following requirement: In the event of a single failure the mission should be aborted so that a safe landing can be performed. Different to the standard solution based on triple redundancy, the detection of a failure in the suggested double redundancy system is problematic. Here a simple voting procedure is not sufficient and elaborated methods based on modeling and estimation on different system levels have to be used. The main reasons to use a double redundancy are the requirements for weight and costs reduction for UAV platforms, especially for platforms used in research and development activities. The main componets are shown in Fig. 2 and Fig. 3.

The system is composed of airborne and ground segments. The ground segment is composed of two identical parts, called *left ground-segment* and *right ground-segment*, as well as operator PCs connected to the TCP/IP-network of the

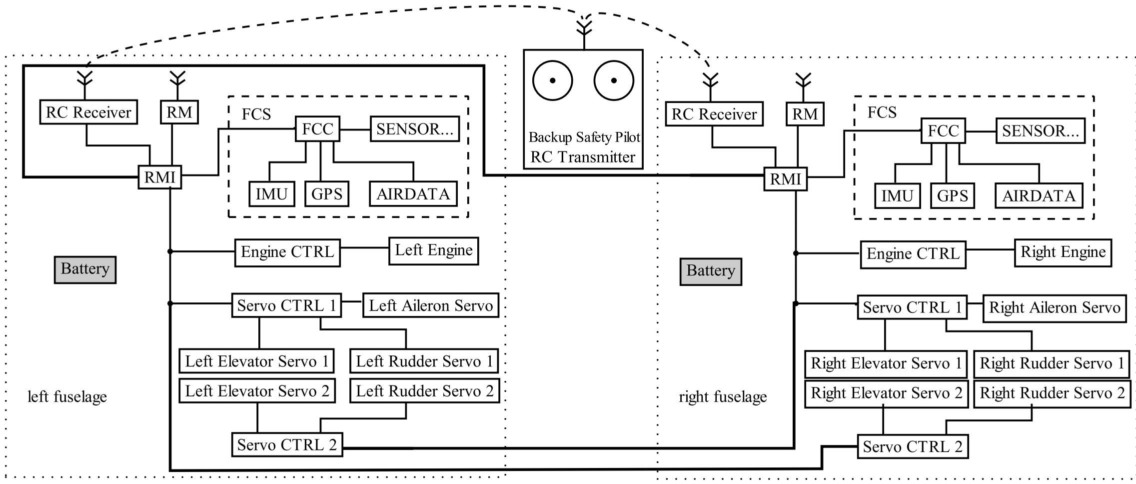

Fig. 2. Redundant ELHASPA airborne segment

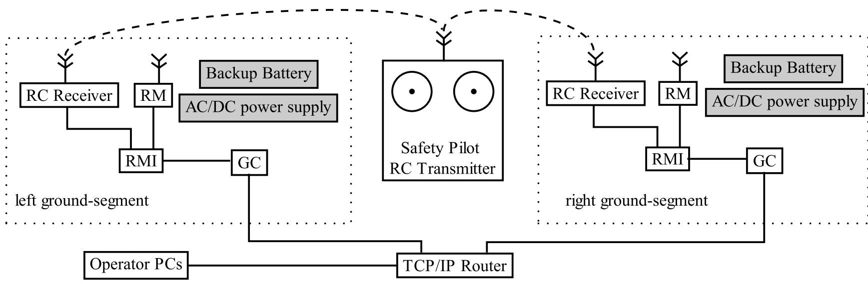

Fig. 3. Redundant ELHASPA ground segment

ground segment. The airborne segment is also composed of two identical parts, denoted as *left fuselage* and *right fuselage*, which are connected via two special blocks called *RMI* (radio modem interface). The *RMI* is the key element in our redundant setup that is why especially for this module careful planning and verification was performed.

Each part of the airborne and ground segments can work independently. When using the system without redundancy as we do in our UAV helicopters and smaler fixed wing UAVs the airborne and ground segments contain only one corresponding part. The part of the airborne segment called *left fuselage* is physically installed in the left fuselage of the ELHASPA UAV and is composed of following components: flight control system (FCS), actuators and power supply. The FCS is composed of a flight control computer (FCC), IMU, GPS, airdata sensor, actuator interface and a radio link (RM) to the ground station. The FCC runs the software described in section III. The sensors and actuators are connected by our standard interface to the FCC. Each FCS can operate independently but only one FCS can control the actuators at the same time. The decision which FCS controls the actuators can be made automatically using some logic and system state data or can be made by an operator in the ground station. In our flight experiments we used the second, more simple, possibility.

As seen in Fig. 1 the UAV has two rudders, two elevators, two ailerons and two engines. Each elevator and rudder has two actuators because the plane cannot be controlled with only one of the two rudders or elevators working. Each aileron has only one actuator because controlled flight is possible with only one working aileron. The engine on the left side is controlled and powered only by the left side, the right engine by the right side. The plane cannot fly with one engine running but it is able to glide with both engines stopped. With this actuator setup it is possible to fly and land the UAV even if all the electric power in one fuselage fails.

The RMIs are used in each part of the airborne and ground segments. They handle the data to be sent to and received from the RMs of airborne and ground segments. The RMI is based on a ARM microcontroller. Three main tasks are performed by the RMIs. 1. Multiplex data streams to allow the usage of the same wireless link for telemetry and control data as well as for safety pilot commands. 2. Decide which input commands the actuators. 3. Communicate with the actuators using CAN bus.

A decision tree is implemented to analyze the status of all control sources and mode switches, this logic leads to manual (normal or backup safety pilot) control, autopilot control or a fail-safe actuator command. Additionally it is required by the redundancy concept to synchronize the actuators.

The synchronization is achieved by selecting one RMI as master with the other side following. Special modes for a lost cross connection between the two airborne RMIs is also implemented.

The system supports two safety pilots. The main safety pilot controls ELHASPA with a normal RC-Transmitter, which is relayed over two long range radio modems to each fuselage shown in Fig. 3. The backup safety pilot uses a normal RC-Transmitter directly from ground to the UAV as shown in Fig. 2. The backup safety pilot takes the control when the main safety pilot connection fails.

Most of the hardware modules have a common interface which means they have the same connector with the same signals and the same pin-out. We use serial RS232 communication and a power supply as our standard hardware interface. We have chosen the serial RS232 interface because of its low latency, easy handling and it is widely used by commercial available sensors and actuators. For sensors with a high data rate we can also use other interfaces like USB or Ethernet. The flight control computer (FCC) is the central component with up to 12 of these standard connectors. We use different computing platforms for the FCC to meet different requirements. With this standardization it is easy to exchange hardware and software between UAVs.

III. SOFTWARE FRAMEWORK FOR UAV PLATFORMS

The software framework was designed to meet the following requirements: modularity, standardized communication and synchronization between modules, easy maintenance and extendability [1]. The scheme of the software framework is shown in Fig. 4. This framework allows to consider the whole system as a set of modules, which communicate using a blackboard. Each module can write or read data slots to the or from the blackboard independently from other modules. The modules are modeled as operating system processes. The system can be adapted to one particular UAV platform and application by choosing the appropriate set of modules. The extension of the system by an additional functionality means starting of the corresponding additional set of modules without changing the existing system.

The framework is composed of three parts, s. Fig. 4: the part running on FCC (green block on the top in the right corner), the part running on the ground control station real-time computer (GC) (yellow block on the top in the left corner), and the part running on the operator PCs (yellow block on the bottom). Fig. 4 shows different groups of modules, denoted by rectangles, running on each part of the framework. The blackboard communication system is denoted by an orange rectangle and is composed of data exchange and system check mechanisms. The data exchange mechanism is implemented using shared memory with synchronization for read/write access. The system check mechanisms implement the system self-monitoring and repairing on low level, e.g. checking for memory integrity or watchdogs for modules. The module *comm* is a communicator process which transfers the data between FCC and GC. This module can be configured for mirroring the whole blackboard or its parts between different CPUs. The modules *sys. monitoring management* perform runtime integrity checks and monitoring of the system on a high level, e.g. restarting of single modules and module groups, detection of hardware failures and reconfiguration of the system. In the system configuration with double hardware redundancy, like for the ELHASPA platform, elaborated algorithms for hardware failure detection can be implemented using model-based estimation and prediction technics. Other modules shown in the Fig. 4 implement system functionalities, e.g. sensor data acquisition and processing, control, navigation and mission execution. The data exchange between modules via the blackboard communication system is configured by a configuration file. This means that an exchange of different modules and reconfiguration of the system does not require code change and recompilation of the modules. The execution order of the modules is chosen to minimize the data processing time over module chains, e.g. starting from sensor data acquisition to the calculation of actuator signals and is implemented using process priorities together with an appropriate OS scheduler.

Each module can be composed of the following components: functional code, blackboard communication library, library for integration of code generated by Matlab/Simulink RTW, library for integration of code generated from Modelica functional mockup interface (FMI). The last two libraries allow generation of a functional code using corresponding standard tools. The modules which contain Matlab/Simulink RTW support a communication using Simulink external mode in real-time operation on the target platform. This opens up possibilities for fast debugging and monitoring of the modules behavior during flight using their Simulink-models running on one of the operator PCs, which are connected to the TCP/IP network of the ground control station. Besides the debugging interface provided by Simulink we have developed custom user interface programs or interfaces to other middleware like DDS [2] described in [3]. An important user interface is the universal telemetry display used in all our UAV operations. Numerical values are displayed on an operator PC in a table as key-value pairs. The values that will be displayed to the operator can be defined in the UAV configuration file. The telemetry display can also be configured to highlight important values with a green/red background when inside/outside of a safe range. The different colors are very helpful for pre-flight checks and in flight health monitoring as has been also suggested in [4].

The implemented modular concept allows the realization of SIL and HIL simulations. For the HIL simulation an additional module *model* - purple block in Fig. 4 - is started on FCC. This module implements the numerical integration of the physical model of an UAV platform. Using an adapted configuration file the data flow is modified in such a way that the sensor signals are provided not by sensor data acquisition modules, but are generated from the system state calculated in the module *model*. The remainder of the system is not modified. All components including

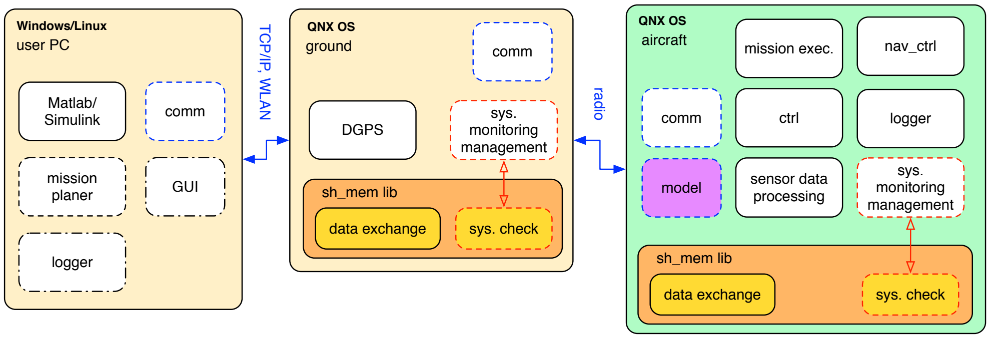

Fig. 4. Scheme of the software framework

the actuators, the propulsion, the ground control station and operator PCs are running in the same mode as in a real flight. The HIL simulation is used to test the functionalities, the software, the hardware as well as to train the crew to work with the system and to perform flight missions. For the SIL simulation the modules can be started in a software environment like VMware virtual QNX machines running on user PC instead of the target hardware. The SIL simulation is used to test functionalities and software of the system. The modularity of the framework allows to create all possible intermediate combinations between HIL and SIL simulation without significant effort.

The current version of the software framework is implemented for the QNX operating system. The usage of the POSIX standard for implementation of the main low level functions make a port for other operating systems possible.

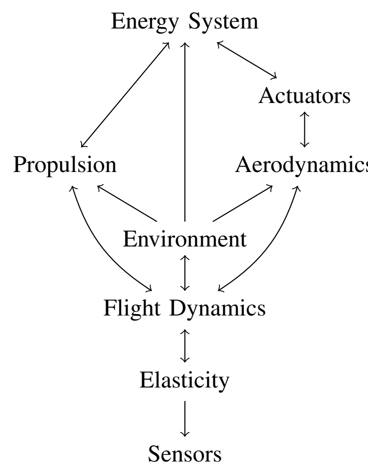

Fig. 5. Modules of the integrated simulation for ELHASPA

IV. Multi-physics Simulation and Control Design

Also when developing simulation models and flight control laws for a range of heterogeneous configurations and applications, it is important to resort to consistent and flexible schemes and tools. They should use an integrative approach, still have good performance and allow derivation of simplifications easily. The approaches developed within the RMC Institute for System Dynamics and Control are used for a range of aircraft including missiles, passenger aircraft and UAVs.

Flight dynamics modelling of the Center's UAVs is done using the dedicated modeling language Modelica. Modelica allows for direct coding of physical model equations, without the need for transferring them into ordinary differential equations first. This on the one hand allows for true multi-physics integrated modeling, and on the other hand, allows a single model to be used for generating dedicated runtime models for various types of applications.

Based on Modelica, a dedicated Flight Dynamics Library [5] has been developed, which is fully compatible with a large number of other Modelica libraries, like multi-body systems, control system blocks, and electronics. This e.g. allows to connect arbitrary components such as moving payloads to the airframe.

These capabilites are used extensively for the integrated ELHASPA model as described in [6]. It consists of modules for flight dynamics, systems, structural flexibility and the environment, which interact in numerous bidirectional ways (see Fig. 5). The energy system is carefully modelled according to the system parameters and includes aspects like solar radiation on the individual solar panels as a function of (local) attitude, position, date, and daytime.

Exploiting Modelica features, model components can easily be exchanged for different levels of detail and different applications. This capability has been of great use in the development process of the ELHASPA aircraft, since the model has been constantly adapted to changing design configurations and new measurements. The first preliminary model versions have been derived from pure geometry data using Vortex/Doublet Lattice Methods, Blade Element Methods and CAD mass estimations. Later models employ e.g. actual thrust and mass measurements. High-end model versions can be made available through model identification [7].

Despite the level of detail, simulation is much faster than real-time and capable of being simulated with a controller in the loop. Thanks to the Modelica philosophy, the system can still be easily reduced to e.g. mass-point inverse dynamic models for control or mission simulation.

The process used for designing flight control laws heavily

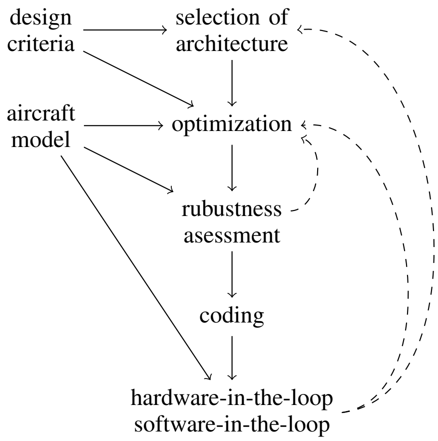

Fig. 6. Flight control design process

Fig. 7. Aerial manipulation with a 7 Dof manipulator

builds on the developed simulation models. The process is depicted in Fig.6 and described in more detail in [8]. The resulting controller for ELHASPA is described in [9]. The same controller has already been flight tested with only minor modifications on DLR's Advanced Technologies Testing Aircraft System (ATTAS).

Selection of a flight control law architecture is based on functional design requirements. For the individual control law functions, modern or classical synthesis techniques may then be used. For ELHASPA, a readily available generic autopilot structure with all modes known from civil aircraft operation has been selected. The inner controller is based on non-linear dynamic inversion [8], while the outer loop employs a total energy control system [10]. The autopilot is developed using Matlab/Simulink.

The control law design parameters are tuned based on detailed function-specific design criteria. To this end, multi-objective optimization is used, available in the multi-purpose in-house developed Multi-Objective Parameter Synthesis (MOPS) environment [11]. The method allows a large number of criteria and constraints to be addressed simultaneously. These criteria are computed from nonlinear simulations, linear analysis, or even robust control-based analysis methods (e.g. μ-analysis). Their relative importance is expressed by means of appropriate proportional or fuzzy-type scaling. The tuning parameters depend on the control law synthesis methods. For example, controller gains in case of PID structures, or weighting function parameters in case of robust control techniques. Several local and global, gradient or non-gradient-based methods may be chosen from for the actual optimization task.

The next crucial step in the design process is robustness analysis, in order to make sure the control laws function properly in off-nominal conditions and all uncertain parameter combinations that may be realistically expected. Of course, in case the UAV has not been flown before, larger tolerances are assumed, which may subsequently be reduced as flight tests progress. For robustness analysis various methods are available. The MOPS environment offers simple parameter gridding for cases with a limited number of parameters, as well as methods like optimisation-based worst-case search [12] and Monte Carlo analysis [13] in case of large numbers of model parameters. Robustness analysis based on linear parameter varying models is used in various applications. When performance specifications are not met in specific cases, these cases may be included in the tuning of design parameters [14].

After passing extensive assessment, control laws and simulation models may be auto-coded and implemented in the on-board software framework for software- and hardware-in-the-loop, and eventually flight testing.

V. Flight experiments, experience

In different projects and for different applications we have to operate platforms with take-off mass between 10 and 100 kg. The usage of the same system in different configurations makes it possible to maintain a wide range of platforms with moderate effort. In Fig. 1 our largest platform ELHASPA with a take-off mass of 100 kg and with a wingspan of 23 m is shown. As described before, for this platform the system is used in double redundancy configuration. For a helicopter platform, e.g. with the take-off mass of 20 kg, the system is used in a configuration without redundancy. These two platforms have totally different flight properties and are used for different applications but the source code for both systems is identical to a degree of 70%. As seen in Fig. 7 we added an 7 DoF manipulator to a helicopter and were able to integrade the control for the manipulator with minimal effort. In Fig. 8 a system for load transportation using three helicopters is shown [15]. Here the system is configured to be used with multiple UAVs.

Early model development also yields more reliable software components, when actually going into flight experiments. During the ELHASPA design phase, the model provided valuable feedback on stability and performance to the design team. The tailplane sizing could be adjusted to yield a more stable aircraft geometry using parameter studies with early model stages. And the offline studies provided several control settings and trim conditions to start flight testing.

Fig. 8. Load transportation with three helicopters

In many research projects, e.g. the EU-projects ARCAS[1] and SAFEMOBIL[2], special flight experiments are required where access to all levels of control as well as mission execution/planning is needed. Preparation of these experiments implies many changes of code and this can be done with reasonable effort only in a modular clearly structured system. The usage of automatic code generation is also an important issue. First of all it reduces the effort for the software testing before going to the flight experiments. Secondly for crew training of complicated flight experiments and for novel platforms pilot trainings are required, therefore the support for that should be provided by the system, e.g. described pilot training set-up and HIL simulation. The set-up proved very helpful in training the safety pilot for steering the aircraft before it was actually available and also to identify gaps in situational awareness and designing appropriate feedbacks.

The presented design ideas and structure for the system is the result of our experience gained in experimental work with UAVs in research projects. In addition to the presented material we would like to point out the following two requirements to the system, which simplify the experimental work with UAVs significantly. First of all, the system has a high level of reliability. The core of the system should be well tested and should not be changed. Only a small part of the system should be changed for a particular experiment. Secondly, the system should provide elaborated possibilities for debugging, including debugging during the flight, data logging and parameter adjustment. Failures of new functionalities are often detected during field testing and flight preparations. Tools for fast system analysis and failure detection make the experimental work more effective.

[1] http://www.arcas-project.eu
[2] http://www.ec-safemobil-project.eu

VI. CONCLUSIONS

In this paper we explained the design of a modular scalable system for operation and testing of UAVs. The proposed and implemented concept of double redundancy is a compromise between cost, complexity, weight and operational safety of research platforms. The ongoing research on the system is devoted to reliable failure detection and automatic decision making in case of a failure. The modularity on soft- and hardware level allows an easy configuration of the system for usage with different types of platforms, with and without redundancy, as well as an easy adaptation of platforms for a particular application. The system is combined with high-end-tools for high fidelity modeling, simulation and control design.

REFERENCES

[1] M. Bernard, K. Kondak, and G. Hommel, "Framework for development and test of embedded flight control software for autonomous small size helicopters," in *International Workshop on Embedded Systems: Modeling, Technology and Applications*, 2006.

[2] "Real-Time Innovations," http://www.rti.com/.

[3] M. Schwarzbach, K. Kondak, M. Laiacker, C.-Y. Shih, and P. J. Marron, "Helicopter uav systems for in situ measurements and sensor placement," in *IGARSS*, 2012, in print.

[4] J. Del Frate and G. Cosentino, *Recent Flight Test Experience with Uninhabited Aerial Vehicles at the NASA Dryden Flight Research Center*. National Aeronautics and Space Administration, Dryden Flight Research Center, 1998.

[5] G. Looye, "The new DLR flight dynamics library," in *Proceedings of the 6th International Modelica Conference*, vol. 1, 2008, pp. 193–202.

[6] A. Klöckner, M. Leitner, D. Schlabe, and G. Looye, "Integrated modelling of an unmanned high-altitude solar-powered aircraft for control law design analysis," in *CEAS EuroGNC*. Delft, The Netherlands: CEAS, 10-12 April 2013 2013, accepted for publication.

[7] T. Lombaerts, "Aerodynamic model identification of frauke uav," in *AIAA Modeling and Simulation Technologies Conference*. Minneapolis, Minnesota: AIAA, 13 - 16 August 2012 2012, aIAA 2012-4512.

[8] G. Looye and H. Joos, "Design of robust dynamic inversion control laws using multi-objective optimization," in *Proceedings of the AIAA Guidance, Navigation and Control Conference 2001, Montreal CA*, 2001.

[9] N. Kastner and G. Looye, "Generic tecs based autopilot for an electric high altitude solar powered aircraft," in *CEAS EuroGNC*. Delft, The Netherlands: CEAS, 10-12 April 2013 2013, submitted for publication.

[10] A. A. Lambregts, "Vertical flight path and speed control autopilot design using total energy principles," *AIAA Paper 83-2239*, 1983.

[11] H. Joos, "A multiobjective optimisation-based software environment for control systems design," in *Computer Aided Control System Design, 2002. Proceedings. 2002 IEEE International Symposium on*. IEEE, 2002, pp. 7–14.

[12] H.-D. Joos, "Application of parallel nonlinear programming methods for worst-case parameter search," in *Optimization Based Clearance of Flight Control Laws*, ser. Lecture Notes in Control and Information Sciences, A. Varga, A. Hansson, and G. Puyou, Eds. Springer Berlin Heidelberg, 2012, vol. 416, pp. 233–252. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-22627-4_13

[13] F. Holzapfel, J. Kladetzke, S. Amelsberg, H. Lenz, C. Schwarz, and I. De Visscher, "Aircraft wake vortex scenarios simulation for takeoff and departure," *Journal of Aircraft*, vol. 46, no. 2, pp. 713–717, 2009.

[14] G. Looye and H. Joos, "Design of autoland controller functions with multiobjective optimization," *Journal of guidance, control, and dynamics*, vol. 29, no. 2, pp. 475–484, 2006.

[15] I. Maza, K. Kondak, M. Bernard, and A. Ollero, "Multi-uav cooperation and control for load transportation and deployment," *Journal of Intelligent and Robotic Systems*, vol. 57, no. 1-4, pp. 417–449, 2010.