ADVANCED VERIFICATION AND TEST METHODS FOR DISTRIBUTED CONTROL ARCHITECTURES IN WATER/WASTE SYSTEMS

Dr. Michael Rempe, Dr. Murat Ünlü
Airbus, Airbus

Keywords: *cabin systems development, distributed embedded systems, decentralized testing*

Abstract

Cabin systems of modern aircraft are increasingly designed as distributed systems. With the A380 Water/Waste System as an example it will be shown how design and testing have to follow the same principles of decentralization to achieve a sophisticated design and test strategy. It will be also shown how modern testing methods can – and have to - cover nearly the whole development process.

1 Introduction

The design of modern water/waste systems has to reflect several requirements concerning flexibility of the cabin and corresponding options like humidifiers, galley inserts etc. On the other hand it has to fulfill challenging operational interruption targets (0,018 for A380) to ensure competitiveness on the market. To cope with these requirements, for the A380 e.g. a basic system architecture comprising 4 independent Waste Sub-systems was selected. To support the general mechanical system architecture a corresponding distributed control and monitoring architecture based on a serial communication bus - the Controller Area Network Bus (CAN-bus [1]) - was chosen. Both the potable water system and the waste system are distributed systems („A distributed system is that composed of a collection of autonomous local small computers connected by a communication network, and equipped with software enabling them to coordinate their activities and share resources“, [2]). All control units and smart actuators/sensors are connected to CAN-buses.

There are four CAN-buses for the *waste system*: one CAN-bus for each side (left hand / right hand) per deck (main deck / upper deck). All waste CAN-buses are fully autonomous. I.e., the system is four times redundant. This is not important for safety reasons but for functional availability.

The *potable water system* comprises two CAN-buses. It has an extra CAN-bus for optional equipment. This architecture suites the requirements of highly operational and customizable systems.

A dedicated cabin interface computer integrates different functions for the water and waste system. It is the interface to the aircraft systems, cabin indications, the maintenance system etc..

Location and number of many components (lavatories, galleys, humidifier valves, etc.) of the water and the waste systems depend on the cabin layout, which is highly customizable. Also future optional equipment is to be taken into account when designing a cabin system.

These requirements show that the application of a distributed architecture is for both systems the most natural and appropriate solution.

The technology is state-of-the-art, but integration and verification concepts have to be adapted to this approach. The development and verification concepts follow the principles of the V-model [3].

From technical point of view it is the same characteristics that make the aforementioned architecture operationally robust and flexible that makes the system difficult to verify. First of

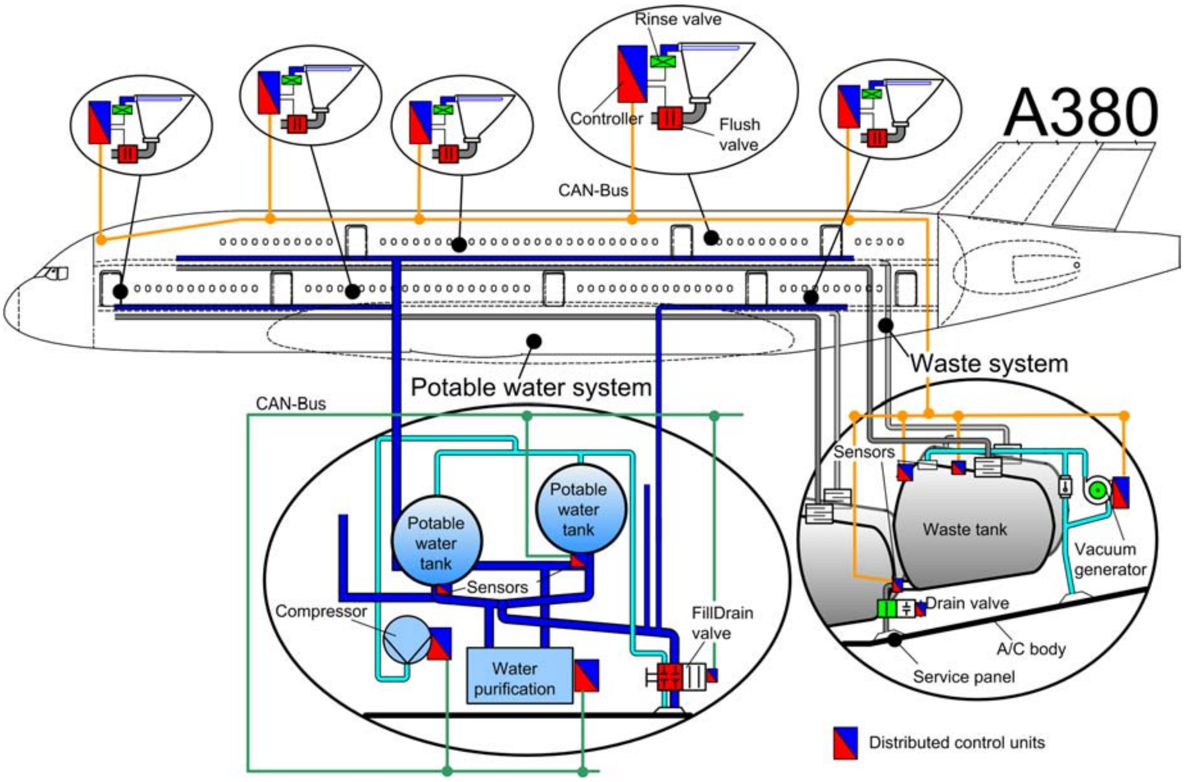

Fig.1: Distributed control architecture of the water/waste system

all the non-deterministic character of the CAN-bus and its nodes is to mention. This makes testing and the applicability of formal methods more difficult.

Integration into the aircraft and verification of such distributed systems requires not only a sophisticated test strategy but also lots of test equipment. The test equipment has to be capable to stimulate the test objects and record and evaluate the test results. The water/waste system of the A380 contains a couple of thousand parameters including measured values, logical values internal and external status parameters etc.

All system components (toilets, water faucets, tanks, valves, sensors etc.) are integrated in the water/waste integration test bench to achieve as far as possible the real behavior of the system on the aircraft. The main objective of such integration tests is the verification of the system and the interfaces to the other aircraft systems.

2 Testing distributed systems

The venture to connect all signals of the distributed controllers and smart equipment discretely to a central processing unit generally causes an error prone cable jungle with a lot of disturbances and a high risk of failures of any kind (wiring failures, perturbations, hardware/software failures, etc.).

This approach results in a test bench that is at least as complex as the system to be tested. The complexity of the system itself ranks higher than the complexity of the single units that the system comprises (controllers, smart actuators/sensors). Test experts know that the assumed detected failures in the system under test have to be thoroughly investigated to ensure that the detected “failure” is not a result of a failure in the test bench itself. Finally, the main result of extensive testing might be a failure free test bench but most certainly not a failure free system. Due to the complexity of a centralized test system it is also not very flexible in terms of modifications of the system to be tested.

However, this is one of the top-level-requirements.
Testing a distributed system with a centralized test architecture is not a straightforward approach.

2.1 Smart test controllers

A natural approach to test distributed systems is to apply a decentralized test strategy. Thus, the system under test can be separated in autonomous test nodes with mainly decentralized tasks. So-called smart test controllers perform these single test tasks. Equipped with a display and a keypad these test controllers can record, stimulate and evaluate signals. Generally, the decentralized test concept provides one test controller per unit to test. Each test controller focuses on the signals and communication data of one unit. This approach has several advantages. These test controllers cover a broad range of application for control units and smart actuators/sensors of the water/waste system. They can be modified easily to adapt to new equipment without extensive changes of the test bench itself. Also, the test controller are portable and can be used outside the test bench, e.g. in climate chamber, EMC-test laboratories or in the aircraft. This the last aspect in particular is of enormous importance. For ground and flight tests the same test controllers can be used. Obviously, one cannot use huge electrical cabinets for those purposes, whereas a small test controller in combination with a laptop is a mobile, handy and smart solution.

2.2 Client/Server-architecture for integration tests

For integration tests especially multiple test controllers need to communicate with each other, where data has to be recorded centrally, or stimulation values or other test parameters have to be set centrally. For this purpose, the test controllers can be connected to a test node via a CAN-interface and via an Ethernet interface. Not connected to a test node the test controllers work fully autonomously or can be controlled through a laptop. When connected to the test network, the test controllers provide additional functionalities [4].

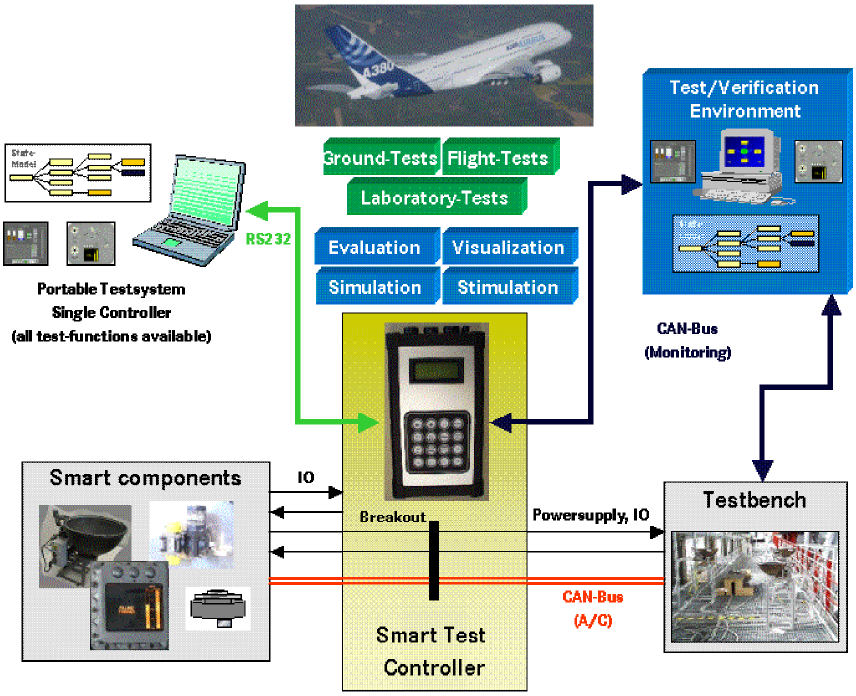

Fig.2: State Designer Test Controller

One special computer, the StateDesigner-Server, controls all actions that require coordination between multiple test controllers. This test server can be physically located on different kind of hosts. The host could be a personal computer with special interfaces, a so-called CAN-megabox, which is a mini computer (smaller than half the size of a laptop) or on a programmable logic controller or a comparable device.

Some of the main tasks of the *StateDesigner-server* are as follows:

➢ Acquisition of all required data of the system under test and transfer of this data to the StateDesigner-client (see below) for recording. The data also can be acquired by the test controllers and transferred to the server through the test network.

➢ Data exchange of test parameters and test data including manual or automatic generated stimulations.

➢ It provides the control of systematic tests. I.e., it allows batch processing of multiple tests without human supervision. The test stimulations can be distributed on the different test controllers. The server performs the synchronization of the consecutive tests.

➢ Interface between the StateDesigner-client (see below) and both the test network and the system network.

The *StateDesigner-client* is the human-machine-interface. It is hosted on a personal computer or laptop. Some of the functions it provides are as follows:

➢ Modelling of the test or simulation models. Template models for different purposes can be adapted to requirements of the test object or new models created.

➢ Generation of test cases for systematic tests. Special algorithms allow the generation of systematic test stimulations from the PTS and other technical documentations. Links for traceability purposes can be incorporated [5].

➢ In the test models modules for automatic evaluation can be incorporated. The evaluation is based on the requirements of the purchaser technical specifications (PTS). Links from the PTS to the models can be used for traceability purposes.

➢ A broad variety of visualization elements can be used to build individual visualizations. It is possible to run multiple visualizations at the same time.

➢ With test models and visualizations tests and simulations can be controlled interactively.

➢ Automatic documentation of test results can be generated in different file formats (xml, html, pdf, rtf). Documentation is tailored according to IEEE Std829 [6].

➢ The data acquired by the test controllers is centrally archived on mass storage.

These demonstrated test concepts have been applied for the testing of the A340 water/waste system and are also being used for the A380 water/waste system Lab, Ground and Flight tests. In the following one possible configuration of the test rig will be illustrated. It is common to all test methods that they are black-box test methods [7], [8].

These proven test methods support practically all phases of the product development beginning from the first design and specification to ground/flight tests with the same methods, tools and models.

3 Example

The demonstrated test methods allow numerous possible configurations for a test stand. Depending on the system and components maturity and availability different configurations are necessary.

Figure 3 shows a possible configuration for integration tests of the potable water system on the test rig. This system consists of the following components:

- Different valves to fill, drain, depress, shut-off the system.
- The potable water indication panel for servicing (PWIP).
- Sensors to measure the water level (PWLS).

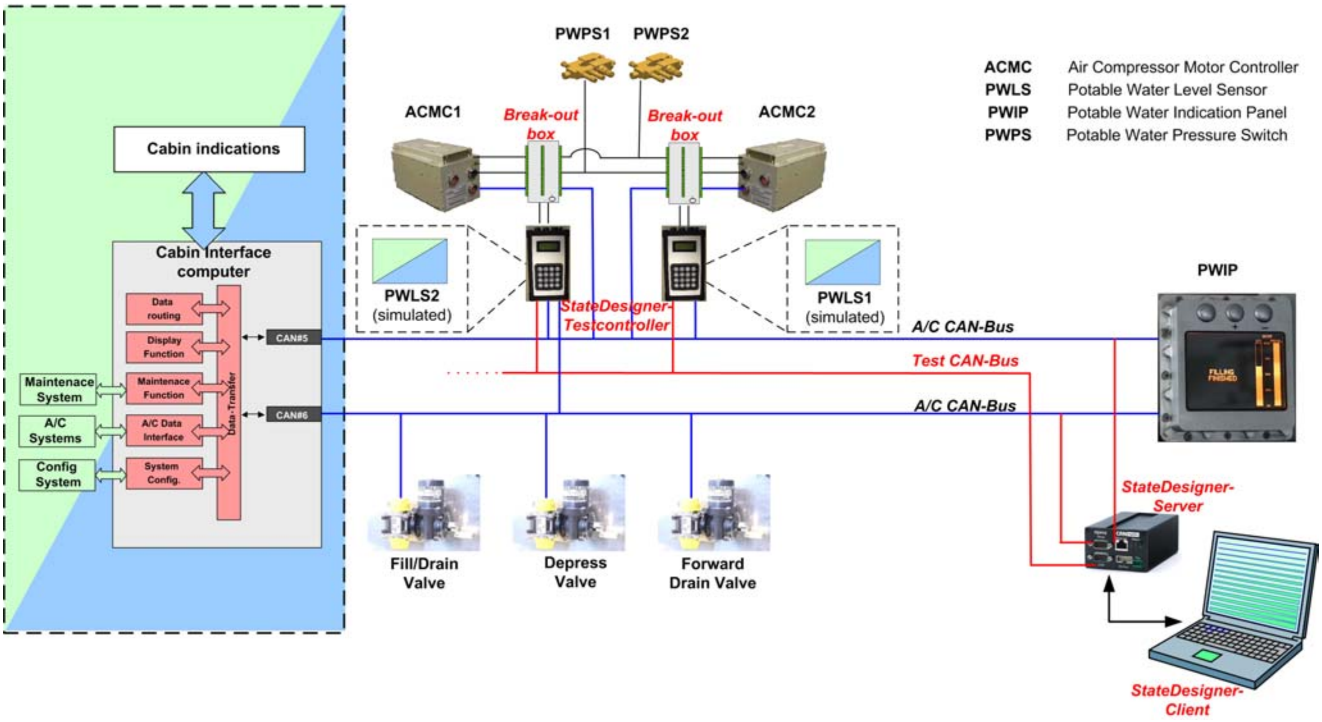

Fig.3: Example test setup for Potable Water System

- Compressors to provide the pressure for the potable water distribution to the consumers (ACMC).

The aforementioned configuration contains the compressors, with the pressure switches, some valves and the indication panel as real hardware components. The Cabin interface computer, the potable water level sensors are incorporated as simulation on the StateDesigner test controller or the server. The pressure switch information is used to activate/deactivate the compressors. The illustrated test setup allows overwriting the real pressure values by the test controllers.

In this way, the compressors can be tested by stimulating different water levels, tank pressures and flight phases. Furthermore the potable water indication panel, which incorporates extensive logics for the valves control can be tested.

For these tests it is very important, that not only the software related aspects like the system logic can be analyzed, but also the physical and electrical behavior of the system is in focus. For example the depress function, which is controlled via the PWIP by sending the corresponding commands to the smart valves, the pressure at several points in the systems is one of the main parameters, which has to be monitored. The results are used to verify all the physical and electrical requirements related to this functionality.

With the methods shown different configurations between original and simulated equipment depending on test task, equipment availability, etc. can be set up. As a consequence, an incremental integration testing is possible in different grades. Different configurations of the test rig can be set up gradually, depending on test progress, test objective, equipment delivery status, optional equipments, etc.

Same methods can be used for validation purposes. For example developing customizable equipments, the test controllers can be programmed as simulation of the equipment to be developed and integrated into the real (or partly simulated) system. Thus the specification can be validated in early an development phase to speed up the whole development process.

4 Conclusions

1. Distributed systems, as usually found in the cabin, require special test concepts that take into account their special characteristics.
2. In order not to have a test system that is ranks higher in complexity than the system to be tested, distributed systems require decentralized test concepts.
3. The decentralized testing method supports incremental testing allowing a flexible approach, planning and progress.
4. The shown methods support practically all phases of the product development beginning from the first design and specification to the flight tests with the same methods, tools and models. By using same methods for early validation and for verification the whole development process can be speed up.

References

[1] Lawrenz W. *CAN Controller Area Network – Grundlagen und Praxis*. Forth edition, Huthig, 2000.

[2] Coulouris G, Dollimore J and Kindberg T. *Distributed Systems. Concepts and Design.* second edition, Addison-Wesley, 1994.

[3] Dröschel W, Heuser W, Middendorf R. *Inkrementelle und objektorientierte Vorgehensweisen mit dem V-Modell 97*. Oldenbourg, 1997.

[4] Rempe M. *Entwicklung und Verifikation von Steuerungssystemen auf der Basis hierarchischer Zustandsmodelle*. PhD-thesis, Ruhr University Bochum, 1999.

[5] Ünlü M. *Toolgestützte Validierungsmethode für verteilte eingebettete Steuerungssysteme im Passagierflugzeugbau.* PhD-thesis, Ruhr University Bochum, 2002.

[6] IEEE. *Standard for Software Test Documentation, ANSI/IEEE Standard 829-1998*, IEEE Computer Society, New York, 1998

[7] Beizer B. *Black-Box Testing - Techniques for Functional Testing of Software and Systems*, John Wiley & Sons, 1995

[8] Riedemann E. *Testmethoden für sequentielle und nebenläufige Software-Systeme*, B.G. Teubner, 1997.