Reuse of embedded software in rotor-craft training systems "Re-hosting approach to guarantee high representativity for TIGER simulation"

Holger Hauser, Bernard Chiavassa
Eurocopter "Software Products Department" (OTWA)
Aéroport International Marseille-Provence
13725 Marignane Cedex, France

Abstract

Due to more and more complex missions of modern helicopter and highly complex avionics systems to support the air crew, a very intensive preparation and training is required. One main difficulty of developing adequate training devices is the provision of representative system behaviour. This behaviour is mostly determined by embedded software functionality.

In the frame of the Tiger Aircrew Training Means (TATM) project, Eurocopter applied for the first time the principle to re-use embedded software for training simulation in Full Flight Mission Simulator (FFMS) and Cockpit Procedure Trainer (CPT). The embedded software is adapted to the simulation system and platform and extended by functions to fulfil simulation specific requirements.

The software development process is derived from the embedded standard, DoD 2167 A, but tailored to the constraints and requirements of the simulation project. Development phases, i.e. Requirement Analysis, and reviews, i.e. Software Specification Review, are maintained but test and qualification requirements reduced to take into account the lower criticality level of the application and to reduce effort and cost.

Software requirement specification and design are conceived as delta products, limited to the modified or extended parts for simulation. The majority of unchanged objects from the embedded development are only referenced.

The re-used embedded software is kept unchanged as much as possible. Interfaces and communication protocols as inherited from embedded design are maintained.

The software development environment from the embedded development is taken over which allows identical code organisation and configuration management between both developments.

A re-host impact analysis identifies an embedded test subset necessary to prove correct behaviour after the modification of re-used software objects. These tests are completed with non-regression tests for the unmodified re-used software parts and new tests for simulation specific extensions.

The Eurocopter test system ARTIST which is used for the embedded software test and validation was ported to the simulation platform and guarantees representative test execution of the simulation model.

Integration results in the simulation system show the high level of representativity of the chosen approach. Helicopter deficiencies caused by the embedded software are inherited and are re-produced by the simulation system for air crew training.

The re-use process and representatvity objective are proven and will be applied to other similar simulation projects as for example the “NH90 training means” development.

Introduction

The missions which modern civil and military helicopter must fulfil become more and more complex. To cope with this complexity, the aircrew is supported by highly complex avionics systems based on digital computation and display devices.

The operation of these systems requires a more intensive training of aircrew. This training is performed increasingly in modern ground based simulation devices, full flight simulators (FFS) or cockpit procedure trainers (CPT).

One technical difficulty of these training devices is the provision of representative system behaviour for the complex configurations. This representativity must be guaranteed over the whole helicopter life including corrective maintenance modifications and avionics system upgrades.

To underline this difficulty, it can be mentioned that for example in the Tiger helicopter the embedded software of one central computer of the basic or mission system package already fulfils several hundreds of functional requirements.

The development cycle of the software took more than 10 years and the helicopter life to be supported is supposed to be over 20 years.

Reproducing the system behaviour starting from the functional requirement specification and maintaining it represents therefore an enormous effort and risk to achieve the final goal.

The principle of simulation is to replace physical devices or equipment by software models. This principle and the fact that the system functional behaviour is mostly determined by embedded software functionality in the central computers lead to the approach to reuse available embedded software with some modifications directly in the simulation system.

Doing this provides simulation with representative system behaviour to affordable costs and with a limited risk. The maintenance can be guaranteed in the same way than the original embedded software.

Eurocopter applied this approach for the first time in the frame of the Tiger Aircrew Training Means (TATM) project.

The final products of this project are the training means, FFS, CPT and CBT, for the French and German Tiger variants HAP and UHT.

Eurocopter supplies in this context to simulation industry general information data-packages, flight model and especially the re-used adapted embedded software of the central computers of the basic management system (BMS) and mission equipment packages (MEP).

Tiger avionics system

The Tiger avionics system is divided into two subsystems: (1) the basic management system (BMS) covers the basic flight functionalities such as navigation, communication and system monitoring; (2) the mission equipment package (MEP) manages the mission specific sensors and the armament.

Both subsystems are built around a central digital computer, one redundant pair for the basic and one redundant pair for the mission system. These computers generate and manage, in addition to their control and monitoring functions, also the symbols and video image switching for the basic and mission displays.

Each computer is bus controller on a Milbus 1553 bus. To ensure the communication between both subsystems the mission computer is additionally remote terminal on the basic milbus.

Further interfaces are Arinc 429 and RS 422.

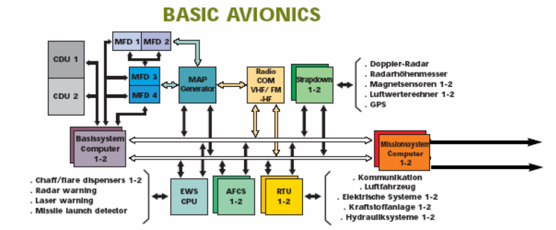


Fig. 1.- Tiger Avionics

The central computers are based on the same target hardware and firmware. The firmware, called equipment software (EQSW), provides an application programming interface to the computer hardware and to the physical I/O interfaces. It provides also the interface to the symbol generator (SG) software which runs within the target on a dedicated processing board.

The functional part of the computer is represented by the Operational Flight Resident Software (OFRS) developed by Eurocopter.

The development process has been performed following the standard DoD STD 2167 A. Software Design is performed using Hierarchical Object Oriented Design (HOOD)

methodology, supported by STOOD ™.
The coding language is ADA83 and the software development environment (SDE) is RATIONAL APEX™.

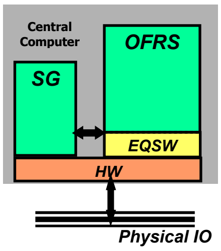


Fig. 2.- Central Computer block diagram

Software activities started in the early nineties. The requirement specification (SRS) covers several hundreds of requirements and the OFRS exceed one million source lines of code. Additional complexity was introduced in the development cycle by a number of important changes and system redefinition, for example the operational extension from the anti-tank to the multi-role mission of the German variant of the Tiger (PAH2 to UHT). The central computers manage several variants of the Tiger; the same OFRS is used on HAP and UHT variants.

Tiger Aircrew Training Means

For the training of the future Tiger aircrews, the French and German army founded together the Franco-German flight school in Le Luc in the south-east of France.
For the equipment of the school with training devices, a simulation industry consortium was selected to provide an initial set of state of the art synthetic training devices, Full Flight Mission Simulator (FFMS), Cockpit Procedure Trainers (CPT) and Computer Based Training (CBT).

One overall requirement to the FFMS and CPT is the provision of representative system behaviour. This system behaviour is characterised by the physical properties of the helicopter, its physical subsystems and cockpit elements and the functional scope of the equipments which form the BMS and the MEP, the avionics central computers and the standard equipment subsystems.
Standard equipments models such as radios or armament subsystems can be developed from scratch based on the documentation (functional specification of the equipment) and observations (real behaviour). This approach is widely established and satisfying results are obtained.

To guarantee the fulfilment of the representative requirement concerning the functional scope of the avionics central computers three approaches are possible.

The first one consists in the analysis of the functional requirements and the system behaviour in order to reproduce the found result in newly developed simulation software models.
With respect to the system complexity this approach represents a significant initial effort and delay to establish the functional requirements for the simulation system. The risk is not to meet the real system behaviour.
The necessary important software implementation and test phases increase the development duration.
During maintenance phase every system evolution might require again an extensive analysis and adaptation phase.

The second approach is to use the original central computer, target hardware and software, integrated in the simulation environment.
The advantage is the guaranteed representativity. On the other hand the embedded hardware has high purchase and maintenance costs and requires a significant integration effort in the simulation environment, i.e. embedded avionic busses.
Specific simulation functions, such as malfunction simulation, can only be stimulated outside the central computers.

The third solution is the re-use of the embedded operational software, OFRS, of the central computer.
It can be adapted to the simulation architecture interfaces and re-targeted to a COTS hardware platform. The core functional scope of the central computers is inherited. It is possible to implement simulation specific functions and to use generalised standard software tools and hardware.
The development cycle is short as the re-used operational part doesn't have to be re-developed. Only simulation specific modifications and add-ons must be implemented.
System upgrades and maintenance actions of the embedded systems during the operational phase of the helicopter can be easily ported to

the simulator. The adaptation process is again applied to the changed embedded operational software or changed software units from the embedded software are updated in simulation.
Nevertheless in order to keep the effort for the adaptation process low, one basic principle is to limit the modification of embedded software to a strict minimum.
The constraints of this solution are the required re-use of embedded software interfaces, i.e. target firmware interface, the functional simulation of the communication protocols, i.e. Milbus 1553, and the requirement to the other simulation models to support the embedded communication protocols and structures.

In the frame of the TATM project the third solution was chosen and implemented. Eurocopter performed the development and provided the software simulation models of the central computers for the FFMS and the CPT.
In the following paragraphs a description of this development process is given.

Re-hosting Development Process

The system analysis for the simulation system broke down the embedded central computer into the following functional components:

- Equipment (i.e. power supply simulation)
- Firmware (EQSW)
- OFRS
- Symbol Generator (SG)

The equipment and firmware simulation together with the adapted OFRS form one simulation CSCI. It has only data processing functionality and requires only data processing resources from the target computer.
The symbol generator is realised as a separated simulation CSCI. Due to the graphical interface it requires different hardware resources and separation permits an optimised target computer allocation.
The presented development process focuses on the first CSCI, equipment and firmware simulation with adapted OFRS.

The development process for the software development is derived from the standard DoD 2167 A but tailored to the constraints and requirements of the simulation project.

The development phase decomposition in phases (a) requirement analysis and specification, (b) preliminary and (c) detailed software design, (d) realization / implementation, (e) test and validation is maintained.
At the end of each phase reviews take place to achieve acceptance of the specific phase output.
The Software Specification Review closed the requirement analysis phase, the Preliminary Design Review closed the preliminary software design phase and the Critical Design Review closed the detailed design phase.
The used document templates are based on the corresponding DID description and partly tailored to the simulation project requirements.

The criticality level of the original embedded software requires a full unit test coverage of the source code. As this classification for simulation software no more applies and as most of the software units where taken over without any modification, the unit test coverage requirement is not maintained.
Furthermore a full requirement traceability of the inherited embedded functional requirements is not required and therefore not applied.

Requirement analysis and specification

The requirement analysis starts with the definition of the helicopter variants baseline. From this helicopter baseline the reference version for the OFRS is derived.
As at the beginning of the TATM project the final helicopter baseline was not yet available an intermediate baseline was chosen for the first developed version and in a second step the initial version was updated to the final project baseline.

The baseline defines the functional scope of the OFRS in the related Software Requirement Specification (SRS). This is the first group of raw requirements at the beginning of the requirement analysis.
Additional requirements are raised by the necessary modelisation of the target equipment behaviour, such as power supply management simulation.
Finally requirements are derived from the simulation system specification and architecture which is defined by the system integrator, the simulation industry consortium.

Further analysis identifies the requirements from the first group which are no longer valid or necessary, i.e. the on board maintenance mode is not part of the simulation so all maintenance mode related requirements are no more applicable. Few existing requirements are detected which must be adapted due to

simulation system specification and architecture. The majority of requirements stay valid without modification for simulation.
For this reason it is possible in order to limit effort and complexity to replace a complete SRS for simulation with a delta-SRS. This delta-SRS references the embedded SRS and lists the “no more applicable” and “modified” requirements with details of modification.
It comprises also the additional equipment simulation and simulation specific requirements, i.e. re-initialization and malfunction simulation.
Finally overall design constraints describe the applicable design principles for the next development phase, such as real-time features.

Software Design

For the software design the same statement as for the requirement specification is applicable. The development is based on an existing model so that a complete new design is not necessary. A similar delta approach is applied as for the requirement specification.

The embedded OFRS software design which followed the Hierarchical Object Oriented Design (HOOD) approach is performed with the design tool STOOD™ and resulted in a base lined design model.
The simulation design uses this tool and adds to the baseline model the necessary additional new design objects to cover the new simulation specific functionality and identifies the existing design objects which must be modified to fulfil new requirements and constraints.

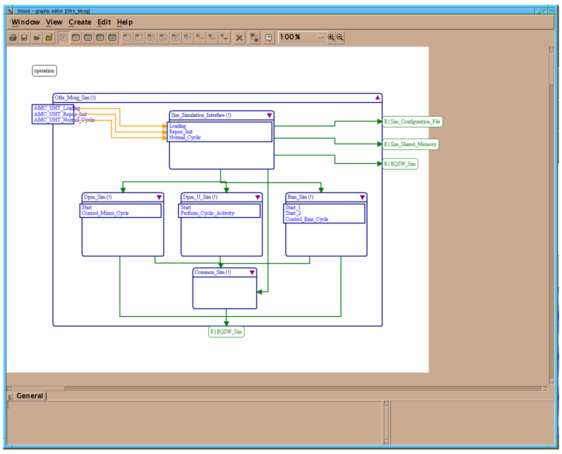
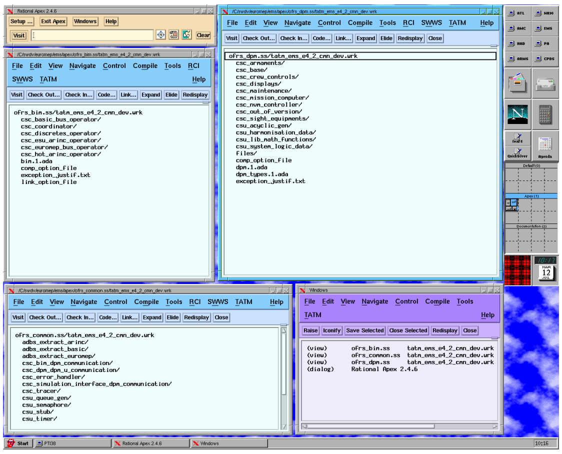


Fig. 3.- Graphical Design with STOOD™

The result is a modified and completed simulation design model. This model is used to create the Software Design Document and to identify the software code units which must be created and modified during implementation in the next development phase.

Implementation

The Software Development Environment (SDE) of the embedded development is taken over for the simulation development. This allows to keep identical code organisation, structure and configuration management. It is possible to maintain the simulation software units within the configuration management history of the embedded software as long as no simulation specific modification must be done. Update to new baselines for these objects is a very limited configuration management action.
To take into account the new target platform, a COTS Unix workstation, the SDE compilation model had to be configured to a new compiler, a COTS ADA95 compiler. The compilation is executed remotely on a target workstation.
The choice of a common SDE permits to optimise the maintenance and support over the operational lifetime of the helicopter respectively the simulator.


Rational Apex 2.4.6
Setup ...
Exit Apex
Windows
Help
Visit
Clear
/C/swdv/euromep/ems/apex/ofrs_bim.ss/tatm_ems_e4_2_cmn_dev.wrk
/C/swdv/euromep/ems/apex/ofrs_dpm.ss/tatm_ems_e4_2_cmn_dev.wrk
/C/swdv/euromep/ems/apex/ofrs_common.ss/tatm_ems_e4_2_cmn_dev.wrk
Windows
File Edit View Navigate Control Compile Tools RCI SWWS TATM Help
Visit Check Out... Check In... Code... Link... Expand Elide Redisplay Close
Raise Iconify Save Selected Close Selected Redisplay Close
ATL NH90 AMC EMS NAD PA ARMS CPDS
Default (0)
Apex (1)
Documentation (2)
10:17
MAR 12
Start
PTI38
Rational Apex 2.4.6
Windows
10:16

```
ofrs_bim.ss/tatm_ems_e4_2_cmn_dev.wrk
  csc_basic_bus_operator/
  csc_coordinator/
  csc_discretes_operator/
  csc_esu_arinc_operator/
  csc_euromep_bus_operator/
  csc_hot_arinc_operator/
  bim.1.ada
  comp_option_file
  exception_justif.txt
  link_option_file
```

```
ofrs_dpm.ss/tatm_ems_e4_2_cmn_dev.wrk
  csc_armaments/
  csc_base/
  csc_crew_controls/
  csc_displays/
  csc_maintenance/
  csc_mission_computer/
  csc_nvm_controller/
  csc_out_of_version/
  csc_sight_equipments/
  csu_acyclic_gen/
  csu_harmonisation_data/
  csu_lib_math_functions/
  csu_system_logic_data/
  files/
  comp_option_file
  dpm.1.ada
  dpm_types.1.ada
  exception_justif.txt
```

```
ofrs_common.ss/tatm_ems_e4_2_cmn_dev.wrk
  adbs_extract_arinc/
  adbs_extract_basic/
  adbs_extract_euromep/
  csc_bim_dpm_communication/
  csc_dpm_dpm_u_communication/
  csc_error_handler/
  csc_simulation_interface_dpm_communication/
  csc_tracer/
  csu_queue_gen/
  csu_semaphore/
  csu_stub/
  csu_timer/
```

```
(view)       ofrs_bim.ss      tatm_ems_e4_2_cmn_dev.wrk
(view)       ofrs_common.ss   tatm_ems_e4_2_cmn_dev.wrk
(view)       ofrs_dpm.ss      tatm_ems_e4_2_cmn_dev.wrk
(dialog)     Rational Apex 2.4.6
```

Fig. 4.- Software Development Environment, RATIONAL APEX™

The implementation itself is performed in three steps.
First the SimEQSW, simulated target firmware, is produced. Its application programming interface to the OFRS is kept identical which avoids OFRS modifications.
Then the necessary changes for the transition from ADA83 to ADA95 are performed.
After successful compilation of the existing code with the new compiler the third step consists in the coding of the new software

objects and the necessary modification of the existing software units. Both groups of objects had already been identified during the design phase.
Finally the complete software model is compiled. Different compilation options can be specified, i.e. debug for use of a debugger or optimisation for execution time.

Test and Validation

The requirement to the test and validation phase can be summarised to the following points.
It must be tested and proven that the nominal (embedded) functionality of the central computer is reproduced and not altered by simulation adaptations and that the simulation specific extensions and functions work as required.

For the first point the retained approach is to use the test definitions and descriptions from the embedded development and to replay it with the simulation model. Two problems are related to this approach.
First the amount of tests defined for the embedded system and the effort to perform these tests completely. As big software parts are used within the simulation model without any modification, there is no reason that the behaviour of these parts should differ from the embedded system.
Therefore a Re-host Impact Analysis is performed to determine a test subset which must be performed to prove the conformity of modified software objects. The analysis is based on the embedded requirement traceability matrix between software and validation tests.
First the list of requirements is established which are covered by the software units taken from the embedded software but modified for simulation. Then the validation tests are filtered with this requirement list to find the tests that test the requirements in the list. The result is a subset of validation tests which covers the functionality of the modified objects.
Nevertheless the not modified parts are globally covered by some specific non-regression tests already defined and existing for the embedded system and taken over to the simulation tests.
The second problem with the re-use of the embedded tests is the test system. In order to be able to replay the tests in a representative way an identical user interfaces for test inputs and model stimulation mechanism is required. For this reason the embedded tests system, the Eurocopter test system ARTIST, is ported to the simulation target platform and adapted to the simulation model interface.
The user interface and tool internal test organisation, i.e. test scenarios, remain unchanged which guarantees representative test execution without any significant difference to the embedded test system.

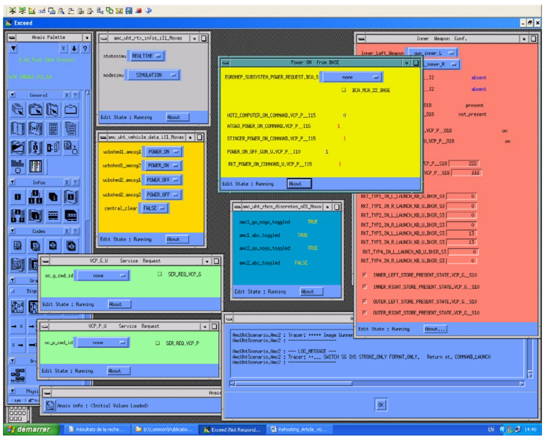


Fig. 5.- ARTIST test scenario user interface

The validation of the simulation specific extensions and functions is then performed with newly defined dedicated tests. These tests follow the same test principles and are based on the same test environment as the re-used embedded tests.

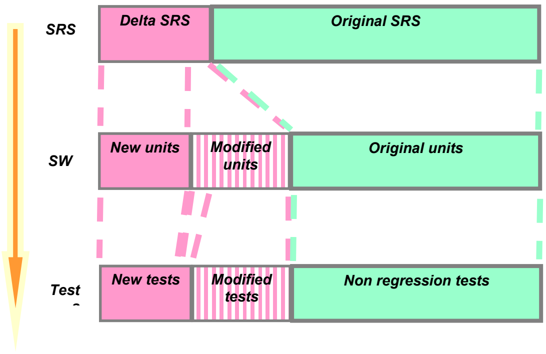


Fig. 6.- Requirement and Test coverage

Besides black box testing it is possible to use a COTS source code debug tool to investigate simulation model internal parameter and variables.

Integration

During simulation system integration the simulation models of the central computers are integrated together with the newly developed equipment simulation models on the final target computer in the simulation environment.

One of the first steps of this integration is to check and correct the implementation of the communication protocols.
These communication protocols are derived from embedded standards, Milbus1553 and Arinc 429, in order to ease embedded software re-use. Certain protocol requirements are not strictly necessary from a simulation functional point of view but must nevertheless be respected to ensure correct communication. The equipment simulation models have to fulfil this protocol.

Another point to stress during the functional integration is the functional representativity of the equipment simulation models. The re-used embedded software is adapted to the real equipment behaviour in the helicopter. This equipment behaviour is not always described in available documentation so that development of a representative simulation model is difficult. In addition certain equipments in the helicopter had still some deficiencies with respect to their nominal functionality at simulation baseline definition. Some of these deficiencies led to work-around implementations in the embedded central computers and therefore are inherited in the simulation model. This appeared as a potential source of problem during integration with nominal working equipment simulation models.

Deficiencies in the helicopter which are caused by central computer problems, known out of the baseline definition, have been re-detected during integration in the simulation system.

Globally the integration results show the high level of representativity of the applied re-use approach.

Conclusion

The concept of re-use of embedded software for training simulation is validated by the results of the TATM project.
The final result of integration shows the fulfilment of the representatvity requirement which was identified as difficult to attain due to the highly complex embedded system and software architecture. Helicopter behaviour is reproduced including system deficiencies and limitations.

The development process to come to a simulation model starting from the embedded software is established and validated. The development cycle is short as only delta functionality must be implemented in the existing software.
The process is as much as possible kept close to the embedded development process to allow optimisation and maintenance synergies of both development branches throughout the helicopter and simulator operational service for updates to new functional baselines.
Software modifications required for simulation and which are not in contradiction to the embedded requirements and constraints can be introduced in the embedded software baseline.

Both developments, embedded and simulation, are kept in one overall management organisation in order to benefit from close or even integrated development teams with transfer of knowledge, harmonisation of schedules and close support for defect analysis. This is considered as another key factor for success of the re-use approach.

Eurocopter applies an identical software re-use approach to the projects ARH Training Devices, Australian Tiger, and NH90 Aircrew Training Media. The lessons learned from TATM allow focusing directly on specificities and optimisation capabilities beyond the established standard process.

In the frame of this project identified standard simulation requirements and constraints can be taken into account already from the beginning for future embedded software developments. The result will be an easier and lighter adaptation of re-used embedded software for training simulation.

Abbreviations

BMS	Basic Mission System
CBT	Computer Based Training
COTS	Commercial Of The Shelf
CPT	Cockpit Procedure Trainer
CSCI	Computer System Configuration Item
DID	Data Item Description
DoD	Department of Defence
EQSW	Equipment Software
FFMS	Full Flight Mission Simulator
HAP	Hélicoptère Appui et Protection
HOOD	Hierarchical Object Oriented Design
MEP	Mission Equipment Package
OFRS	Operational Flight Resident Software
PAH	Panzerabwehrhubschrauber
SG	Symbol Generator
TATM	Tiger Aircrew Training Means
UHT	Unterstuetzungshubschrauber Tiger