About the influence of virtualization and open source technologies on V&V lifecycle

Jean-Pierre Lebailly
EADS - EUROCOPTER
ETZSR
Marignane, France
jean-pierre.lebailly@EUROCOPTER.com

Nicolas Belanger
EADS - EUROCOPTER
ETZSR
Marignane, France
nicolas.belanger@EUROCOPTER.com

***Abstract* —the present paper discusses the strategic evolution of avionics test systems at EUROCOPTER as requested by virtualization turn. Until now considered as a necessary evil in the global avionics product lifecycle, test systems are taking the path to be promoted as key productivity enablers. Industry competitiveness pushes avionics actors to revisit and strengthen their Verification & Validation capabilities. To support it, EUROCOPTER has launched dimensioning projects that are introduced in this article. EASI (Eurocopter Avionic System Ide) which will be the new Test System at EUROCOPTER is presented as a case study. The benefits of open source technology compared to old fashion proprietary test systems are discussed. On top of that virtualization turn is illustrated through the SDMU (System Digital Mock Up) tool developed among others to simulate interfaces between system components in order to fix equipment specifications as early as possible to reduce the lead time of integration tests on rigs and flight tests. Perspectives are then given in terms of Test and Simulation system near convergence.**

Keywords - Model Based Testing; Open source; Avionic System Rig; Virtualization; EASI; Avionic Simulation.

I. INTRODUCTION

For the past 20 years, the major question when building test systems was to ensure the real time performance. Due to technology limitation, the solutions were proprietary operating system embedded in specific VME (VERSA Module Europa) [2] hardware. The test methodology was an important parameter but constrained by the technology. The real time expertise was the core competency for test systems departments in avionic companies. Nowadays, the evolution of hardware allows us to imagine new real time solutions based on standard and low cost components. The performance of the hardware becomes less important. At the same time, industry competitiveness becomes the major parameter. The systems are more and more complex but the cost and delay are drastically reduced. The consequence is that focusing on validation and test at hardware level is no longer enough because it occurs too late in the project life. The way to improve the process is to perform validation at each step of the V cycle. That means using: Model Driven Architecture for requirement capture and architecture definition; Simulation to validate system specification and detailed specification; Tests on rigs to qualify equipment and system integration. This approach has a major impact on test systems domain: the validation cycle is distributed along the whole V cycle; heterogeneous tools cooperate to the final result; new populations are concerned by test and validation. Whereas test systems were until now considered as a necessary evil, they are now entering into the phase of potential productivity enablers. The scope of the department Test Systems is becoming larger, services level agreement must remain at the highest level while expertise resources cannot increase significantly. Only critical components may remain EUROCOPTER internal, the others shall be open source components or delegated to third parties. Needed expertise becomes more integration oriented. This new model is illustrated in the present paper through a current status of test system at EUROCOPTER, a study of potential benefits of virtualization approach at system architecture level, the EASI (EUROCOPTER Avionic System Ide) case study. A current status about long term test systems practice at EUROCOPTER is performed, then virtualization turn in terms requirements and benefits is introduced. At least in chapter 4 we present the EASI case study with a particular focus on open source technology interest for test system change. Moreover, the case study introduces the SDMU tool for avionics interfaces simulation.

II. CURRENT STATUS

EUROCOPTER test systems activity began in middle 80's. Three tools, Mona Lisa, Anais and Artist have been developed successively. The necessity to build a new tool was justified by a major modification of the hardware environment or the addition of a major functionality. In anyway, the architecture principles stay unchanged and we can speak globally of EUROCOPTER Test System Tools in the present article.

The first intention was to take into account three main aspects:

1) Test and qualification of avionic systems: Software Test Bench and Integration Rigs;
2) Specification and debug of embedded software through simulation capabilities;
3) Development framework for training facilities.

Our ambition was to address not only the right part of the verification and validation cycle, equipment test and system integration test, but also to be able to address the left part at the specification level. We will try to understand why some

limitations have made it a relative success and why it is now the good period to reach our objectives taking advantage of recent technical gaps.

A. Test tools architecture

The architecture built in the middle of the 80's is still operational today. It is organized around:

- A real time platform
- A workstation dedicated to piloting;

For the last twenty years, full VME architecture was the best solution to manage avionics systems hard real time testing. VMEBus is particularly efficient to allow I/O event management: multi processing synchronization, access to different hardware resources (CPUs and I/O boards) in a transparent way.

EUROCOPTER has integrated VMEBus as a standard backbone for embedded helicopter systems integration test bench activities. The current architecture used at EUROCOPTER for System Integration Rig is introduced by Figure 1:

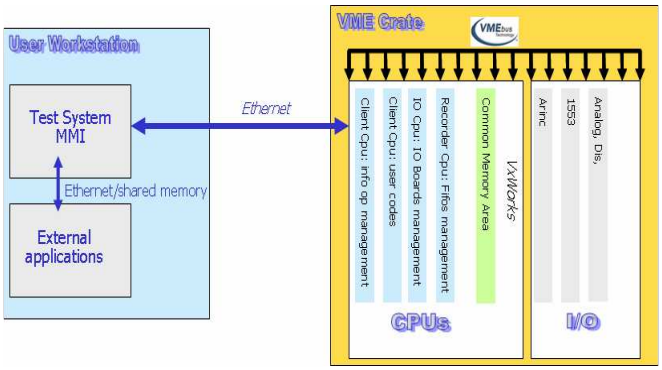

Figure 1. EUROCOPTER Integration Test Bench current architecture

Three categories of real time CPUs using VxWorks operating system are embedded in the crate. Client CPU are dedicated to computing tasks which consist in: first, coding and decoding data coming or sent to the avionic equipments; secondly, in simulation models. IO CPUs are managing the interface between the Test environment and the specific board connected on the avionic buses (ARINC 429, MIL-STD-1553, etc.). Recorder CPUs are recording data's in the dedicated protocols format with no discrepancy introduced by the tool. A common memory area allows synchronized and controlled real time exchange between CPUs.

B. Limitations

The use of EUROCOPTER tools for test, qualification and training was globally successful. Nevertheless, we have encountered some limitation against the competitiveness of the product and the use of specific functionalities, especially during the last years.

1) Structural constraints

As in other companies, we have to produce quicker, cheaper and keeping product quality at the same level. In our model, all the functions are made internally that needs maintaining an important development team and induces costs. The use of external software like: Oracle, pSOS and VxWorks generates license cost.

2) Real time debug

EUROCOPTER test system offers real time debugging capabilities for user codes. This capability is mandatory when you want to use the tool for embedded system development through simulation. In both real time environments, pSOS and VxWorks, there were some issues which prevented us to put the functionality in service. First, Multi real time environment for pSOS and Tornado environment for VxWorks induce high license cost. Secondly the real time debugging capabilities were not completely mature in case of multiple CPUs configuration. As an example, propagation of a breakpoint on all the CPUs was not possible with Multi-pSOS and generates unpredictable and unacceptable jitters with Tornado-VxWorks.

3) Customized BSP (Board Support Package)

We have been obliged to customize the CPU BSPs in order to guaranty some real time functionalities like:

- Use of the SYSFAIL to synchronize board on a physical signal;
- Message passing between CPUs;

Each time we decide to use a new type of CPU, it generates high recurrent costs to propagate these modifications.

4) Customized IO boards API (Application Programming Interface)

As some functionality's of the IO boards were not usable through the supplier standard API, we have developed our own customized API. In the same way as for the BSPs, each time we integrate a new board we have recurrent costs.

5) Test bench user profile

Since 20 years, the testers profile has changed. Test teams were composed of technical profiles aware of avionic buses protocols. Nowadays, the profile is more a generalist software engineer with good knowledge about avionic buses but some lack at the protocol level. In case of failure during a test they have difficulties to diagnose if the original cause of the problem is located: in the tool, in the procedure or due to the equipment under test. The philosophy of EUROCOPTER tools fits the needs of people familiar to avionic bus protocol but this philosophy is limited with less mature team. This may induce blocking point and increase delay.

6) Ergonomics

During the past 20 years, the industry of personal computing has increased. Everybody is now working on a personal computer for office automation and has also one or

more personal computers at home. Ergonomic evolution on these platforms is continuous and very fast. In people's minds, it should be normal to have the same evolution at work with the tools supposed to help them to improve their productivity. Even if the tools are operational and able to fit the needs, there is a rejection if the tool does not provide a familiar environment with all ergonomics facilities.

III. VIRTUALIZATION

A. Competitiveness constraint

As other companies, EUROCOPTER is strongly challenged by competitors in terms of Time to Market, quality, product costs. It requires finding new leverages to improve our competitiveness factors. Jumping to virtualization process will allow validation loop at each step of the V Cycle thus shortening time frame deliveries, improving product quality and reducing final costs. Virtualization is an emerging project at EUROCOPTER in which we are making progress in the preliminary analysis stage: no choice is already made in terms of tools, methods, architecture.

B. Brief state of the art

Even if still far from moving to "source model" concept rather than "source code", the emergence of Model Driven Architecture (MDA) [3] supported by the Object Management Group (OMG) has become a reality. In its 2006 Workshop on Model Driven Architecture [12], the Carnegie Mallon Software Engineering Institute stated that "MDA has been endorsed by various entities of the U.S. Department of Defense (DoD)". Moreover, in [13], a Defense Science Board report on the Joint Integrated Fire Support Systems (JIFSS), a large distributed system, it was stated that JIFSS "should employ the Model-Driven Architecture (MDA) development approach for designing the JIFSS architecture and in implementing its component systems. The MDA [5] approach ensures adherence to standards across the components and has been shown to substantially reduce costs in the development of large-scale systems-of-systems". In the same way, Lockheed Martin [6] in the frame of F-16 Modular Mission Computer Application Software development, proved the real interest of model driven development techniques for industrial purposes close to the ones of EUROCOPTER. On top of that, Forrester Consulting conducted a major study in 2008 on 132 companies about Model-Driven development methodologies [8], and concluded that putting the model as the central artifact of development life cycle has major benefits. Last but not least, TOPCASED project [7] recently provided some important results consecutively to a proof of concept performed in 4 different functional areas which argue in favor of Model Driven development.

C. EUROCOPTER users requirements

EUROCOPTER needs a total development approach meaning that architects, designers and engineers should be involved from start to finish of a project. The first requirements which are pushed by EUROCOPTER system architects are:

- Being able to automatically parse each requirement in order to diagnose its correctness according to potential errors formally described in [14]
- System requirements coverage analysis based on modeling approach: being capable to provide automatic validation about the requirements consistency and completeness.
- Being able to re-use use cases at corresponding steps of the V-cycle (see Figure 2)

It requires to be supported by tooling enabling Virtualization. This is the mandatory path to support avionics design office. We, therefore, plan to build a global framework (cf. Figure 2) in which interact the following components:

- Modeling, Test System Functionality, System Digital Mock-up (SDMU) [11],
- Standard Interfaces to real avionics & IT hardware,
- Modular inclusion of Tools: interface databases, Test Automation, any COTS (Commercial Off-The-Shelf),
- Available on every engineer's desktop.

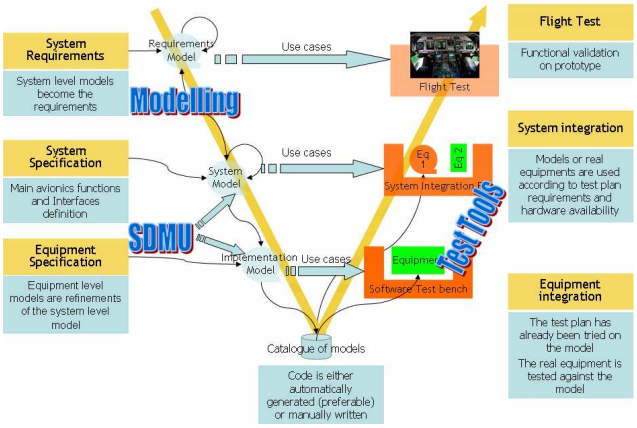

Figure 2. EUROCOPTER avionic design office framework

D. Virtualization benefits

The main benefits expected by EUROCOPTER in the jump to Virtualization perspective are:

- Reach "avionic product first time right delivery"
- Shorten Time to Market
- The capability to capitalize from one project to another through models library
- Extend the test completeness : capability to test failures in the whole flight domain in simulation mode;

IV. THE EASI CASE STUDY

In the near future, we must be able to federate heterogeneous people working in heterogeneous environment with heterogeneous tools. The success factor will be our availability to provide the good tools for each kind of job and a framework allowing EUROCOPTER teams to work together, avoiding redundancy and improving quality by optimizing the work at each level. How could we face all these new subjects without increasing unreasonably the size of our test tool team?

This can be achieved relying on 3 basic pillars (cf. Figure 3): Collaborative Approach, Open Source and SDMU:

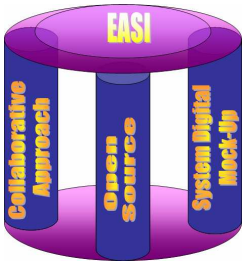

Figure 3. EASI framework pillars

A. *Collaborative approach*

After a detailed overview of the market [9], we decided to build a collaborative solution with EADS TEST&SERVICES. EASI project aims to build an EADS collaborative development platform focusing on avionic systems integration tests. The original and ambitious strategy of EADS TEST&SERVICES is to propose a modular plug-in solution at MMI (Man Machine Interface), communication and also at real time level. This relies on three mains components:

- U-TEST RTC (Universal Test Real Time Component), an open real time component distributes **third party modules** in a unique and integrated real time test system;
- A plug-in based MMI allows association of development from various parties and fits it to real time system modularity;
- An open framework manages a network based communication between all the parts of the test;

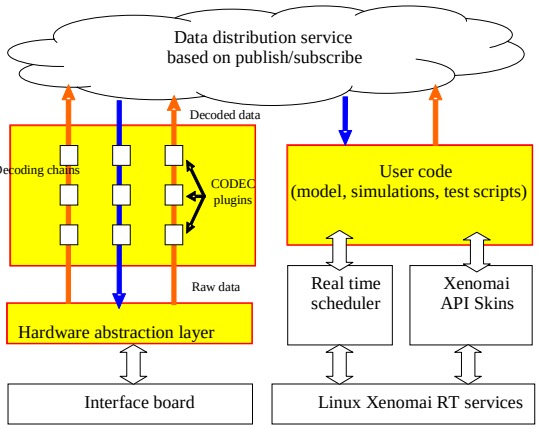

Figure 4. U-TEST RTC architecture

On figure 4, all the non blank parts of U-Test are concerned by the real time plug-in approach. This allows U-Test integrators to connect, under their own, specifics hardware to the system. Concretely it is one key feature that has driven EUROCOPTER's choice for Test System change. This is the guaranty 1) to be able to keep all our existing hardware without any modification of the wiring of the bench, 2) to keep the availability to adapt under our own the tools to future EUROCOPTER specific needs. For EADS TEST&SERVICES it is the opportunity to inherit EUROCOPTER specific developments and to integrate them to its offer. At EADS level it is a chance to share test systems development between different Business Units. The Figure 5 presents the organization proposed between EUROCOPTER, EADS TEST&SERVICES, EADS other Business Units and Open Source world in the frame of EASI Project.

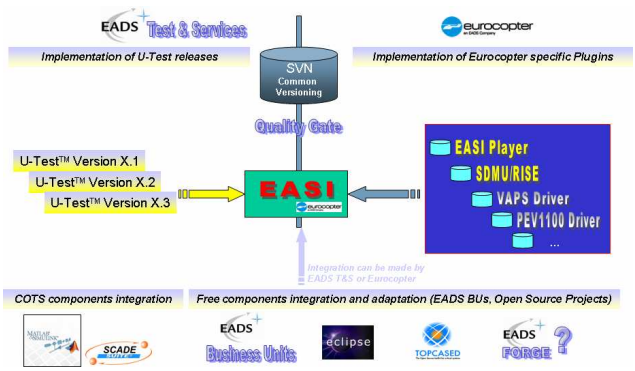

Figure 5. EASI project organization

The innovative collaborative model imagined for the EASI project proposes to share the developments between EUROCOPTER and EADS TEST&SERVICES under a common versioning (SVN). The provider/customer classical model is then slightly changed. In order to face the Virtualization step and its test systems renewal, EUROCOPTER needed to be partnered by a test system core

specialist which role was to become a pure "services backbone" as keeping the ability to contribute into the developments in terms of close helicopter test systems specificities. EUROCOPTER will then keep test system expertise knowledge to design specific developments. These developments will be performed through plugins and used as temporary versions. The specific plugins shall pass a development quality gate to be accepted by EADS TEST&SERVICES and to be integrated to major versions delivered twice a year. After having being integrated to major versions, the EUROCOPTER plugins will then be maintained (fixes) by EADS TEST&SERVICES.

B. *Open Source*

An important challenge for EADS TEST&SERVICES and EUROCOPTER is the capability to develop a new test system from scratch with strong constraints in term of budget and delay. We have made the choice of using Open source components; the potential advantages are well known:

- Ease and accelerate the development phases;
- License free
- Supported by large communities;

One challenging aspect of using Open Source is to have in minds that our Test systems must have a long term life. These are costly systems that must support avionic system development during all the product life which exceeds tens of years. It is important to choose well known and stable Open Source components:

- U-Test real time tasking is based on orocos (Open Robot Control Software) components;
- U-Test MMI is based on Eclipse. Consequently, it inherits of all the C C++, fortran and ADA Eclipse development and debugging platform capabilities. This is a great advantage for developing simulation models;
- U-Test data distribution service is based on the popular EPICS framework. That opens to U-Test all the data control and visualization tools which shares this framework.

Another challenging aspect is to rely on communities whose constraints and goals may be completely disconnected of ours. We have no assurance that these components will evaluate in a way which fits our needs. Perhaps that means that we must become actors in Open source projects.

C. *Advantage of openness by examples*

Since the end of our initial study, the project has completed two important steps

- A prototype phase;
- An operational mockup phase;

The goal of the prototype phase was to validate the main principles of this new collaborative approach, demonstrating the capability to work in a collaborative model between two EADS Business Units and that the plug-in approach proposed by EADS TEST&SERVICES allows EUROCOPTER to implement real-time connection to its specific hardware without degrading performances.

On another hand, the goal of the operational mockup phase may be seen as the first industrialization step. Its goal was not to demonstrate the technical ability of the project, but to show that based on the Open Source strategy it is possible to build a complex test integration platform at the state of the art in term of ergonomics and which is well accepted by our customers.

1) *U-TEST RTC Real-time plug-in*

The goal of this paragraph is to describe how it has been possible to reach EUROCOPTER wish to implement by our self the connection of EASI to our VME interfaces boards.

To avoid costly modifications, the main requirement of EUROCOPTER is to keep as much as possible the hardware interface of the rigs unchanged and particularly the wiring. It is why we impose that EASI must be able to drive the system through the IO boards in VME crates which are used with ARTIST, which is the actual test system. The problem is that U-TEST RTC component is based on a PC platform and consequently it is using the PCI bus for board connection.

The first and mandatory step was to find a way to connect the VME crate and the PCI Express. Of course this link must provided all the services we are using for IO boards management:

- DMA;
- Interrupt management;
- Mail boxes

We had the opportunity to be a privileged partner of IOxOS technology for the development of the PEV1100 [15] PCIe bridge. The very impressive results of the test with a prototype of the PEV1100 convinced us to select it as the basis of the connection of EASI to the VME world.

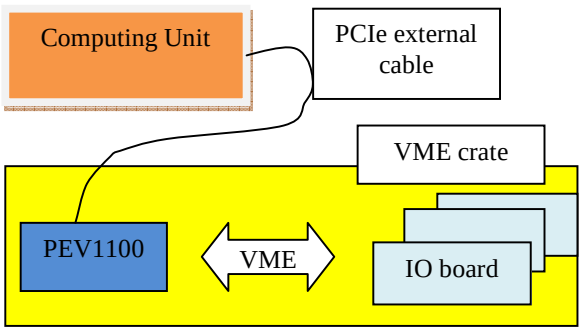

Figure 6. PEV1100 Overview

At this step, the hardware platform is completely defined and the remaining task is to implement the access of EASI test system to the IO boards embedded in VME crates through the PEV1100 Bridge using the plug-in approach allowed by U-TEST RTC.

The plug-in approach of the U-TEST RTC component is based on codec and protocol. Data is decoded / encoded using a chain made up of a list of user definable modules. These modules can be either used to extract the payload from a message (e.g., identifying the real data associated to an ARINC 429[21] label based on the label given – these are called protocols) or to convert data (called codecs) such as endianness conversion, scaling, BCD decoding, etc. Protocol and codec chains are executed sequentially in order to transform raw data to system level data. The result of this process is published in the Global Data Space making it available to other system parts (which can interconnect with via a LAN). In addition, it is possible to publish intermediate calculation between codecs to Global Data Space so that the user can analyze the result of each step inside the decoding / encoding chain. This modular approach eases capitalization between projects by sharing real time data processing code. For a given test and for each system variable, the decoding chains are defined in an XML Interface database. At the beginning of the test or on demand, the chains are instantiated according to the database definition. Each codec and protocol plug-in contributes to enlarge a C++ Interface Class library. An Xml description of the plug-in allows to make a dynamic link between U-TEST RTC and the library where the entry point of the plug-in implementing the methods of the Interface Class are provided.

An important point is that the way to access each IO resource is described in the Xml Interface database and may consist of multiple protocols and codecs chained sequentially. So, what is the application to the particular case of accessing IO boards in a VME crate?

First EUROCOPTER has implemented the elementary protocols and codec plug-in:

- A protocol to open a PCI expresses access to the VME area through the PEV1100 bridge;
- For each type of IO board embedded in the VME crate, a protocol to manage the board;

It has not been necessary to implement codecs because codecs to encode/decode engineering values inside ARINC 429 labels (BCD, BIN) or MIL-STD-1553[22] message were already provided by U-TEST RTC;

Secondly, we have described in the Xml Interface database that the board embedded in the VME crates must be accessed by chaining:

- The PEV100 protocol plug-in;
- The dedicated IO board protocol plug-in
- The codecs plug-in;

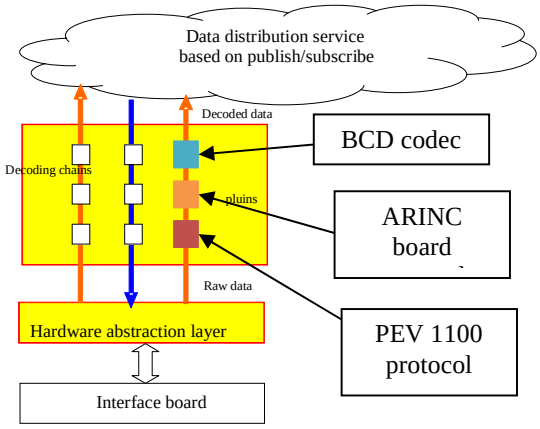

Figure 7. VME chain of codecs and protocols

At the end of the operational mockup phase, the real-time plug-in strategy, as shown on Figure 7, is fully demonstrated. Of course some problems are remaining, for example, we are looking for a way to ensure a complete synchronization between the PCI express world and the VME world especially in term of event dating. There are known solutions like IRIG [24] link, but it is a costly solution and not available with all our park of IO boards. In any way, these kinds of problems are not linked to the plug-in and collaborative approach.

2) Open source strategy the Eclipse and CSS (Control System Studio) example

Another main strategy of the EASI project is to remain on Open source. Here are two typical example of how Open source could help to build complex environment.

The Eclipse RCP framework

The modular approach proposed by U-TEST RTC on the RTOS (Real Time Operating System) side had to be matched with the same level of customizability on the MMI side. Thus, EADS TEST&SERVICES used Eclipse RCP as the base framework for its client side application.

Eclipse RCP is a platform for building and deploying rich client applications, supporting the famous Eclipse Java Integrated Development environment. It provides fundamental workbench functionalities such as movable and stackable window components (editors and views), menus, tool bars, push buttons, tables, trees, and much more with an OS native look and feel for applications and features.

Moreover, Eclipse RCP offers commercial off the shelf modules needed in our EASI project test system:

- Development environment for user code, eclipse CDT [16].
- Decoding chains Database modeling (EMF) [17]
- Test and projects configuration management (Subversive) [18] [19]

Each of these modules can be “overloaded” in order to customize it to match our needs. For instance, EADS TEST&SERVICES added auto-completion capabilities to

the user Code Development Module (CDT), based on avionic signal declaration in the Interface Configuration Database. That means that when the end user wants to access an avionic signal in a simulation code, he may use auto-completion based on the definition of the avionics signals done by the architect designers.

The Control System Studio

EASI is an avionic test system, it must be able to interface and control complex avionic configuration. It is then mandatory to provide an ergonomic interface to interact with the system. Based on Open source approach, the choice for EASI has been to do an investigation on existing products. Of course the criteria have been to find an existing tool fulfilling the requirement in term of ergonomic but also in term of integration with EASI. We have just seen in the previous paragraph that EASI is based on the Eclipse RCP framework, it has also been mentioned that the data distribution service used in EASI is the popular EPICS. Comparing all the solutions based on Eclipse and EPICS, we have chosen the CSS (Control System Studio) [20] tool because it is worldwide used in scientific community this ensures its perenniality at mid and long term, its components cover a large scope of EASI needs and will drastically improve graphical control capability compared to our actual tools.

The main effort to integrate the CSS to the EASI project has been to merge the EASI avionic configuration database and the CSS signal description. But, the task has been really eased by the fact that the two projects are base on the same EPICS frameworks. Finally, with a reasonable effort, we can take advantage of a very powerful tool usable as it is. For the future, we plan to developed some new graphical control more avionic oriented. Of course these new components will be shared with the CSS community as CSS plug-ins.

At the end of the operational mockup phase of EASI, we can say that the Open source strategy is a real success. The two previous examples show that if we build a coherent architecture, for example in our case, choosing Eclipse framework for GUI, EPICS framework for Data distribution service and then CSS based on the two previous frameworks, it reduces drastically the cost and delay for building complex IDE. Another advantage is that it lEADS in a natural standardization of the interface between the different components of the system. Therefore, it will be Easier to develop and integrate new functionalities in our test system. In a simplified way, providing an EPICS connection to our text system extension seems sufficient to ensure connectivity. The Open source strategy based on popular component is also, at mid and long term, a guaranty to stay at the state of the art in term of functionality and ergonomic due to the large community which improve continuously the quality of the products. In EUROCOPTER, and especially in the Test system department, it is really a 180-degree turn compared to the 20 passed years. We must go from a complete proprietary model to an Open source one and of course if we hope to maximize the advantages we can expect of this new way of work, we must become actors and contributors of the Open Source.

D. SDMU

SDMU [11] is an internal project dedicated to simulation of avionic systems. One of the main difficulties in avionic system development is to produce a complete and consistent interface definition. The main goal of SDMU will be to simulate interfaces between system components (as shown on Figure 8) in order to fix equipment specifications as early as possible to reduce the lead time of integration tests on rigs and flight tests.

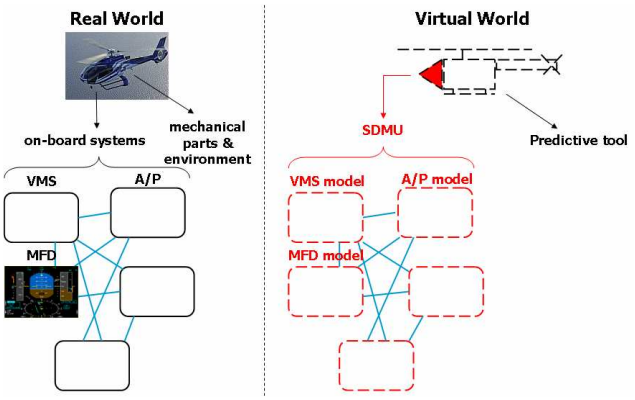

Figure 8. SDMU concept

1) Description of SDMU objectives

The simulation general objective is to increase Validation & Verification activities earlier in the development cycle on a mock-up before performing classical rigs and flight tests qualification.

Specifically, SDMU will be used in a static mode to refine the consistency of systems interface definitions and in a dynamic mode to control the functional behaviour. It allows the simulation of complex (redundant) architectures using the real data flow (A429, A653, Analogues, Discretes) for a realistic representation and an easier coupling with real equipment (hardware in the loop). Last, but not the least; SDMU is an automatic generation process based on the Avionic System definition.

2) GILDA

GILDA is a new tool developed by EUROCOPTER for IMA/ARINC 653 [23] avionics and takes also into account ARINC 429, analogs and discretes. GILDA is based on a set of XML files which are managed by configuration via SVN. The goal of GILDA is to collect information in order to:

- Describe system and equipment communication (as well as internal A653)
- Define equipment and partitions outputs (extension of ADBS - Avionics Data Base System which the central database for all avionics signals - to data structures exchanges)
- Define equipment and partitions inputs

GILDA description is then used to generate IRS (Interface Requirement Specification) documents (for CIGALHE equipment and for partitions), to feed ADBS in order to configure bench and flight test tools and to feed SDMU.

3) SDMU simulation building process

The 5 steps of SDMU process presented on Figure 9 are explicated hereunder.

Step 1:

The first step required to build SDMU simulation would be Interface analysis. Due to processes included in the forecast SDMU tool set, all the interfaces would be analysed to check the consistency of data exchanges, then a set of files would be produced for all the signals exchanged through the system and all the functions needed to manage these signals.

Step 2:

The second step would consist of using a specific process to translate all the interfaces described in the database into files that are usable by the SDMU simulation (libraries, headers).

Step 3:

Today, almost all of EUROCOPTER's sub-system specifications are entirely defined using SCADE® tool. The forecast tool would integrate a C-code generator to translate functional description sheets into C-code files. These generated files would be then embedded in SDMU simulation to be managed as other simulation models are.

Step 4:

The HMI partition used to display flight and vehicle data to the crew is defined using SCADE® for the logical part and VAPS® for the graphical part. These two tools include a C-code generator to obtain a graphical model communicating with other models through ARINC 429 exchanges.

Step 5:

In order to emulate the assessed sub-systems, a set of models are to be developed to "feed" the simulation with representative data from the physical world. All these models would produce or receive data in engineering format (physical data) which will have to be translated into interface value like frequencies or voltages (raw data). To perform this translation, a set of sensor models or/and actuator models will be developed. To encode or decode values from engineering data format to raw data format (or vice-versa) these models would use functions and services provided by the forecast SDMU translator. They would be thus automatically updated when a new simulation is to be integrated after delivery of a new version of the centralised interface data base or a software upgrade.

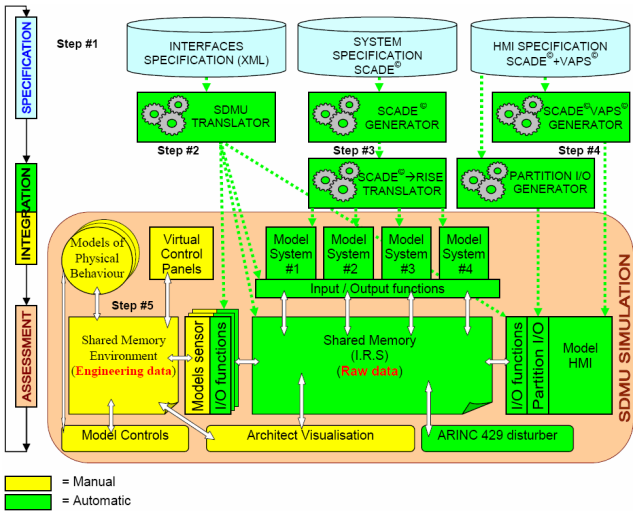

Figure 9. SDMU simulation building process

4) SDMU features and application domains

The main characteristic of SDMU is a powerful simulation addressing low cost hardware platform, running on standard PC or Laptop and compatible with Unix/Linux environments.

It covers many application fields like Rapid prototyping, simulation including hardware in the loop capability, training media framework. SDMU is based on RISE (Real time Interactive Simulation Environment), which is the EUROCOPTER internal simulation tool. Later, SDMU and RISE will be integrated into the EASI framework.

An SDMU implementation in RISE tool is presented hereunder on Figure 10.

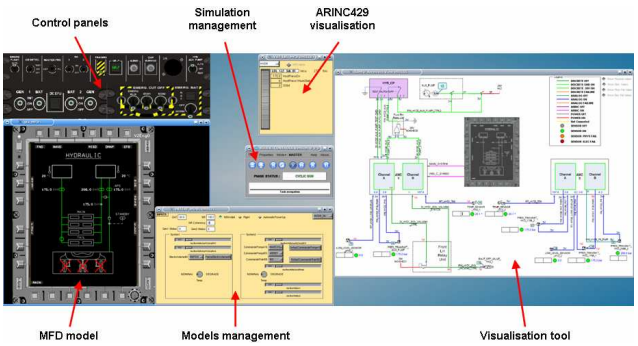

Figure 10. SDMU implementation in RISE

E. CENTRAL REPOSITORY FOR MODELS

To reach the virtualization objectives, it will be necessary to have first a repository where the interfaces of every equipment will be managed in configuration including all input and output signals and secondly a flexible simulation platform allowing to integrate models together building a realistic simulation.

The models forecasted to be used for simulating the system would be:

- Realistic simulation of the helicopter flight
- Realistic simulation of avionic equipments
- Models of the physical behavior of components included in the system simulation (hydraulic system, electrical generation system, flight dynamics, …)
- Sensor or actuator models that translate engineering data from the physical world to system format parameters,
- On-board software models (Vehicle Management System, Automatic Flight Control System, etc).

The management of models is becoming a central topic. In this idea, it is planned to develop a new tool called EASI Repository. This tool aims to centralize all objects used to develop, test and simulate Avionic systems as described on Figure 11.

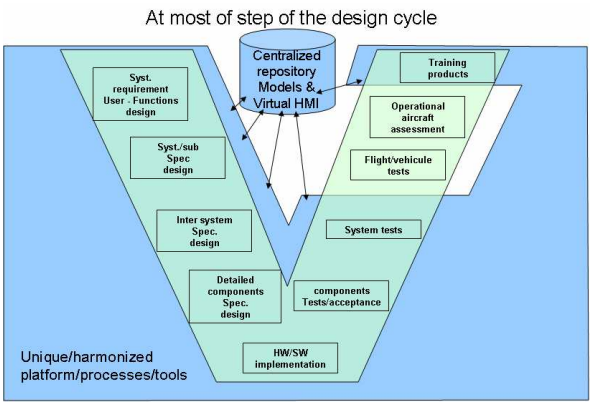

Figure 11. Central repository of models – EASI repository

In a first step, EASI Repository shall contain Avionic Interfaces (equipment, frame & data definitions) and hardware logic definition (FPGA, BSP, relay, diodes).

In a second step, EASI Repository should contain models (Equipment models or re-hosted code, Virtual Panels), Environment codes (Aerodynamic, atmospheric, sensor), simulation panels and dashboards.

EASI repository will improve sharing Virtual aircraft unitary bricks and services between developers through project life cycle. It will ensure that avionic definitions, system models (generic and specific) and environment models are defined and developed once by the specialist and offer to the system community. All EASI repository services shall be based on a strong and rigorous configuration management linked to the project.

Some standard formats should be defined to ease integration and adaptation of repository bricks to the simulations. These standards will authorize automated process for building simulations by connecting models according to the avionic architecture and interface stored in EASI repository. The automation based on specific communication layers services allows retargeting avionic software without modification constraints. The communication between the simulated avionic interfaces shall be transparent and must offer internal monitoring capability with no added effort of the end user.

V. PERSPECTIVES

EASI Tester prototype phases are completed. The EASI Tester operational mockup phase is on the way and will be completed in September 2010. On a Linux quad core platform, we have demonstrated:

- The capability for EUROCOPTER to integrate real-time plug-in to manage its current hardware including hardware located in external VME crates;
- The performance level to perform representative avionic real-time simulation connected to avionic equipments through multiple IO including ARINC 429, Mil1553B, analog, discrete;
- The advantage which should be expected of the Open source strategy.
- The possibility for EUROCOPTER and EADS TEST&SERVICES to share the development of a complex avionic IDE.

We plan to deliver into production the first EASI bench in early 2011. SDMU is already operational on desktop benches and allows anticipating system interface tests and validation.

The next tasks will consist in:

- Launching a model driven approach proof of concept at system requirement level;
- Building progressively a convergence between EASI and SDMU in order to set up hybrid platforms mixing simulation and hardware
- Building a global repository allowing to share within EUROCOPTER: avionic definition, simulation models, flight test data.

REFERENCES

[1] N. Belanger and JP. Lebailly, “Promoting tests system as productivity enablers: The EASI case study”, IARIA, ICONS 2010, International Conference on Systems, pp. 66-70 , April 11-16 2010, Les Menuires, France.

[2] VERSA Module Eurocard (IEEE 1014) http://en.wikipedia.org/wiki/VMEbus

[3] Object Management Group, “Executive Overview: Model-Driven Architecture”. http://www.omg.org/mda

[4] N. Belanger, N. Favarcq and Y. Fusero, “An open real time test system approach”, IARIA, VALID 2009, IEEE International Conference on Advances in System Testing and Validation

Lifecycle, 20-25 September 2009, pp. 38-41, Porto, Portugal
[5] D. Flater, "Impact of Model-Driven standards", National Institute of Standards and Technology, Gaithersburg, USA, HICSS, vol. 9, pp.285, 35th Annual Hawaii International Conference on System Sciences (HICSS'02)-Volume 9, 2002
[6] Lockheed Martin aeronautics, Lockheed Martin (MDA Success Story), www.omg.org/mda/mda_files/LockheedMartin.pdf, January 2003
[7] N. Pontisso, D. Chemouil, "TOPCASED Combining Formal Methods with Model-Driven Engineering", French Space Agency, Automated Software Engineering, ASE '06. 21st IEEE/ACM International Conference, pp. 359-360, Toulouse, France, 2006
[8] "Modernizing Software Development Through model-Driven Development", Forrester Consulting, A commission study conducted by Forrester Consulting on behalf of Unisys, 13rd of August 2008, Headquarters, Forrester Research, Cambridge, USA.
[9] T. Schlichtherle, "Open Source Business Models in Transition, Part 2: life cycles, market sectors and new competition strategies", yeebase GbR, 2007
[10] A. Raundahl Gregersen, N. Jørgensen, "Extending eclipse RCP with dynamic update of active plug-ins", Journal of object technology, Vol. 6, No. 6, pp. 67-89 July-August 2007
[11] E. Guillon, N. Brisset, N. Damiani and M. Goasdoué, "System Digital Mock-Up – A new approach for the design and development of helicopter avionic systems", 35th European Rotorcraft Forum conference, 25-28 September 2009, Hamburg, Germany
[12] G. A. Lewis, B. C. Meyers and K. Wallnau , "Workshop on Model-Driven Architecture and Program Generation", Software Engineering Institute, Carnegie Mallon University, Technical Note CMU/SEI-2006-TN-031, 2nd June 2006, Pittsburgh, USA
[13] Office of the Under Secretary of Defense For Acquisition, Technology, and Logistics, "Report of the Defense Science Board Task Force on Integrated Fire Support in the Battlespace", Washington, DC, October 2004, http://www.acq.osd.mil/dsb/reports/ADA428791.pdf
[14] I. Hooks, "Writing good Requirements", Proceedings of the Third International Symposium of the NCOSE - Volume 2, 1993
[15] N. Belanger, J. Bovier, JF. Gilot, JP. Lebailly and M. Rubio, "Multi core computers and PCI express, the future of data acquisition and control system", ETTC 2009, European Telemetry Conference, 24-26 June 2009, Toulouse, France.
[16] Eclipse CDT homepage, http://www.eclipse.org/cdt/
[17] Eclipse EMF homepage, http://www.eclipse.org/modeling/emf/
[18] Subversion homepage, http://subversion.apache.org/
[19] Eclipse subversive plug-in, http://www.eclipse.org/subversive/
[20] CSS homepage, http://css.desy.de/content/index_eng.html
[21] ARINC 429 Wiki, http://en.wikipedia.org/wiki/ARINC_429
[22] MIL-STD-1553 Wiki, http://en.wikipedia.org/wiki/MIL-STD-1553
[23] ARINC 653 Wiki, http://en.wikipedia.org/wiki/ARINC_653
[24] IRIG, Inter-Range Instrumentation Group time codes Wiki, http://en.wikipedia.org/wiki/IRIG_timecode