UDC 629.735.3.+681.3.06.519.85 (045)

[1]D. A. Prosvirin,
[2]V. P. Kharchenko

OPTIMIZED SAFETY – CRITICAL EMBEDDED AUTOMATIC FLIGHT CONTROL SYSTEM DEVELOPMENT APPROACH TO COMPLY WITH ARP 4754, DO-178C OBJECTIVES

Air Navigation Systems Department, National Aviation University, Kyiv, Ukraine
E-mails: [1]dimitry.prosvirin@gmail.com, [2]kharch@nau.edu.ua

Abstract—*This article deals with model based embedded software development approach of aircraft automatic flight control systems with using new model-based approach. Realization of air-borne equipment software requirements, regulated by functional safety standards such as ARP 4754, DO-178C is showed. This article explains how mentioned requirements can be obtained using SCADE. The possibility of documentation and qualified code generation from SCADE Display and SCADE Suite models is showed. Use of the mentioned approach allows facilitating embedded software development and certification process.*

Index Terms—Automatic flight control system; model-based design, SCADE, code generation, software certification, executable specification, verification, documentation, ARP 4754, DO-178C.

I. INTRODUCTION

The avionics industry requires that safety-critical software be assessed according to strict certification authority guidelines before it may be used on any commercial airliner. ARP 4754 and DO-178C are guidelines used both by the companies developing airborne equipment and by the certification authorities. Presently, numerous people play a role in defining and creating safety-critical systems for the avionics industry. The function and architecture of a system are defined by system engineers using some informal notation for the graphics and associated logic. The embedded production software is then specified textually and hand-coded by software engineers in the coding language augmented by a graphical library. In this context, search for new safety software development methods is important and actual task. These methods may help to reduce influence on the final product of next factors: inexact understanding by executor of customer requirements; late mistakes detecting and as a result, expensive process of alteration; considerable cost of certification. Also mentioned methods shall include a technology of qualified code generation from formal models that may carry strong Return on Investment (ROI), while preserving the safety of the application.

In addition, the U.S. Federal Aviation Administration (FAA), European Aviation Safety Agency (EASA) and other international authorities of aviation security insist on the use of functional safety standards, that to ensure the proper functioning of a complicated electronic equipment of aviation systems in any foreseeable conditions, to exclude defects and the possibility of the aircraft crashes.

II. PROBLEM STATEMENT

Today aviation companies gradually come to model-based embedded software development process for onboard systems. So, search for a new applications witch support a model-based development paradigm is a relevant task. This article deals with the approach description through a software model, including the graphics and the associated logic, and to automatically k ts

of cost and productivity improvement in the development of safety critical software for avionics systems and associated methods are considered [8]. Existing methods based on data flow block diagrams and allow saving rewriting the description of the controller when going to software development [9]. The majority of researches reviews the activities traditionally performed in such developments and how individual tools can increase productivity in such a context. In this scope of rigorous software development, various verification techniques have been proposed to streamline software verification while preserving the safety of the application [10].

This objective shall be achievable only if the proper engineering and design processes are deployed in conjunction with the proper development tools. This paper presents a combined approach to these needs and concludes with a status of current research activities on the topic and a summary of the benefits provided by this approach for automatic control systems.

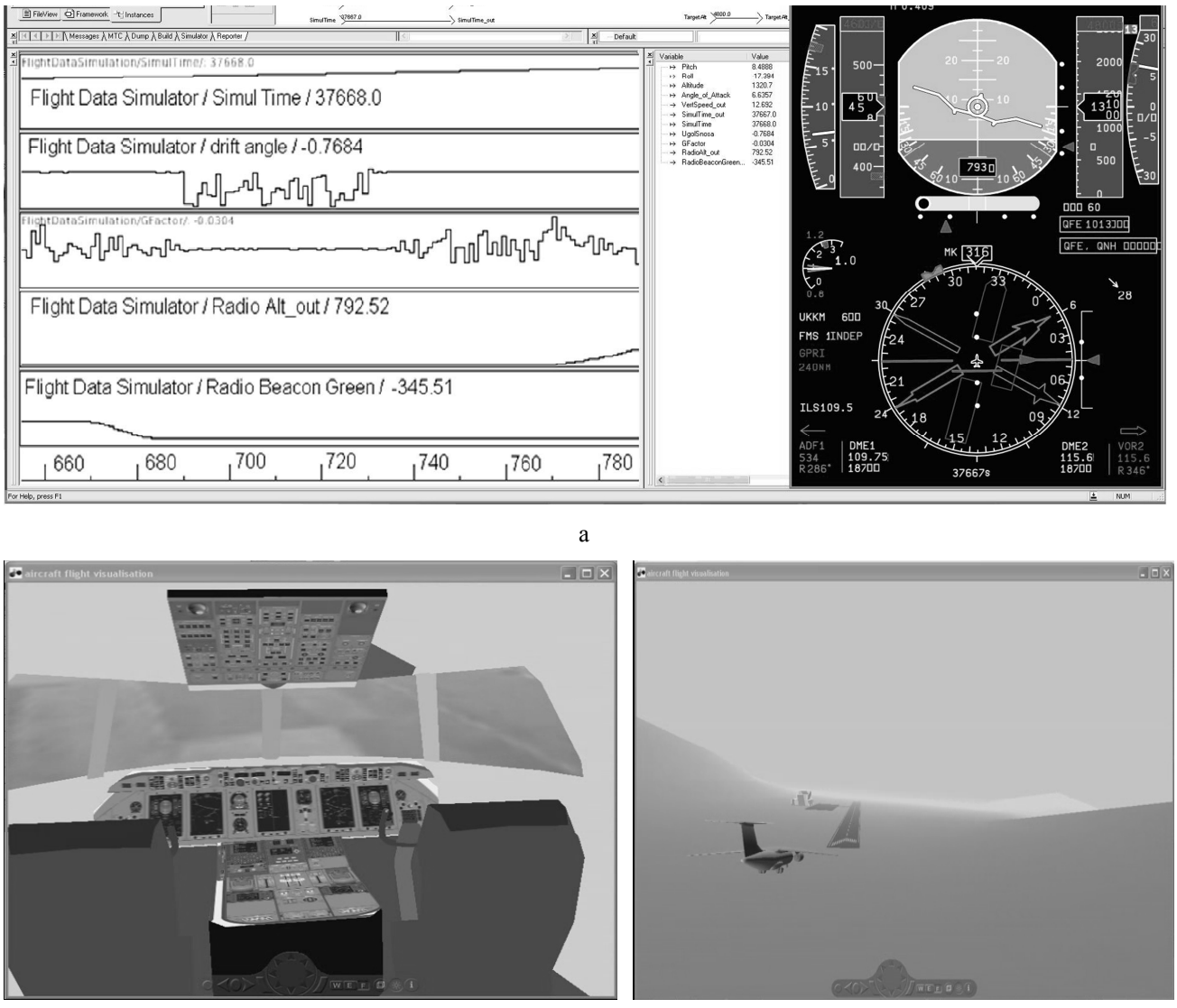

Fig. 1. Standalone virtual test bench for flight simulation:
(a) flight data (test scenarios) simulator; (b) aircraft view; (c) cockpit view

Название	Тип	Значение	Комментарии
ROLL_MIN	real	-45.0	
THROTTLE_FACTOR	real	0.15	
WX_FACTOR	real	0.1	
WZ_FACTOR	real	0.1	
ZERO_PITCH	real	0.0	
ZERO_ROLL	real	0.0	

3.1.3. Course Оператор

Declared as **public блок**

3.1.3.1. Графическое представление

3.1.3.1.1. Вид diagram_Course_1

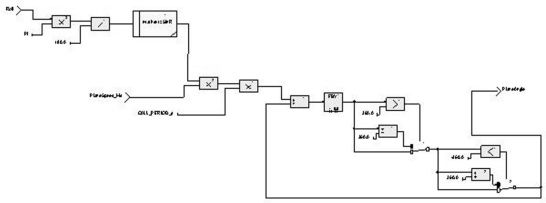

Рисунок 1: Вид diagram_Course_1

3.1.3.2. Интерфейс

Таблица 3: Входы Course

Название	Тип	Комментарии
PlaneSpeed_Ms	real	
Roll	real	

Таблица 4: Выходы Course

Название	Тип	Комментарии
PlaneAngle	real	

a

```
/* Object 1, Priority 0, Name: rectangle, Type: rectangle */
{
  sglSetActiveMasks(pContext->_parentmasks);
  /* filled object part */
  sglDisable(SGL_TEXTURE_2D);
  sglIndexColor1(51);
  sglDisable(SGL_POLYGON_SMOOTH);
  sglTransparency8(255);
  sglBegin(SGL_POLYGON);
    sglVertex2f(0.0F, 200.0F);
    sglVertex2f(90.0F, 200.0F);
    sglVertex2f(90.0F, -200.0F);
    sglVertex2f(0.0F, -200.0F);
  sglEnd();
  /* outlined object part */
  sglIndexColor1(41);
  sglIndexLineWidth1(0);
  sglIndexLineStipple1(0);
  sglDisable(SGL_LINE_STIPPLE);
  sglDisable(SGL_LINE_HALOING);
  sglBegin(SGL_LINE_LOOP);
    sglVertex2f(0.0F, 200.0F);
    sglVertex2f(90.0F, 200.0F);
    sglVertex2f(90.0F, -200.0F);
    sglVertex2f(0.0F, -200.0F);
  sglEnd();
}

/* Object 2, Priority 0, Name: panel group, Type: panel */
{
  sglPushMatrix();
  sglTranslatef(0.0F, -200.0F);
  sglBeginScissor(90.0F, 400.0F);


  {
    sglPushMatrix();

    /* Object 4, Priority 0, Name: Lines, Type: container */
    {

      /* Object 5, Priority 0, Name: line, Type: line */
      {
        /* outlined object part */
        sglIndexLineWidth1(1);
        sglBegin(SGL_LINES);
          sglVertex2f(2.0F, 211.0F);
          sglVertex2f(5.0F, 211.0F);
      }

      /* Object 6, Priority 0, Name: line, Type: line */
      {
```

b

Fig. 2. Text and code generation:
(a) generated documentation fragment; (b) automatically generated code fragment

IV. Solution Procedure

Formal Model-Based Development (FMBD) with SCADE consists of a rigorous unambiguous graphical semantic which combines data and control flow as well as a formal action language used to represent the architectural and behavioural aspects of a software-centric system. SCADE models formalize a significant part of the software architecture and design. The model is written and maintained once in the project and shared among all team members: from the specification team to the review and testing teams. Expensive and error-prone rewriting is thus avoided, interpretation errors are minimized. This formal definition can even be used as a contractual requirement document. Basing the activities on an identical formal definition of the software may save a lot of rework, and acceptance testing is faster using simulation scenarios. In the software requirements process, partial SCADE modelling is a good support for the identification of system functions, its interfaces, and data flows.

V. Simulation Results

Mentioned above advantages can be achieved by SCADE (Safety Critical Application Development Environment), environment for the development of safety-critical avionics software that includes the following components: SCADE Display for graphics design and SCADE Suite for onboard systems logic design.

SCADE models formalize a significant part of the software architecture and design. The model is written and maintained once in the project and shared among all team members: from the specification team to the review and testing teams. Expensive and error-prone rewriting is thus avoided, interpretation errors are minimized. This formal definition can even be used as a contractual requirement document with subcontractors, for example between the aircraft manufacturer and control systems supplier. Basing the activities on an identical formal definition of the software may save a lot of rework, and acceptance testing is faster using simulation scenarios. Some companies start using Esterel SCADE to prototype control systems during the definition phase. In the software requirements process, partial SCADE modeling is a good support for the identification of system functions, its interfaces, and data flows. SCADE -model control algorithm can be graphically specified using data flow diagrams.

The functional parts of the software, such as implementation of the geometrical transformations described, logic, filtering, and regulation can be fulfillment with SCADE Suite. SCADE Display is well-adapted for all the graphical display part of the software. It is well-suited to completely specify the

dynamic behavior of the developed control laws for automatic flight control system as illustrated on the Fig. 1.

It is now helpful to dynamically exercise the behavior of a SCADE Display/SCADE Suite model to verify its operation. As soon as a SCADE Suite model (or pieces of it) is available, it can be simulated with SCADE Suite Simulator. Simulation can be run interactively or in batch. Scenarios (input/output sequences) can be recorded, saved, and replayed later on the Simulator or on the target. For simulation scenarios, you can use aircraft flight testing data, so the SCADE Suite and SCADE Display models can be co-simulated to provide a fully realistic view of both graphics and logics. Combination of SCADE Display and SCADE Suite modeling can be use in the software design process to develop major parts of the requirements and the architecture. Use of both mentioned instruments allows execute joint logic and graphic debugging.

So, the SCADE architecture is defined, the main modules are refined to formalize the requirements. SCADE-model control algorithm and dynamic behavior of the Display application are specified. The objective of this activity is to produce a complete and consistent software model. The requirements of the project are described as executable model which can be send to aircraft electronic flight instrument system designer. For more clearness, usability and reporting there is a possibility to generate text specification from SCADE model that can be then agreed and signed between customer and developer (if it's needed).

Documentation is automatically and directly generated from the SCADE models (Fig. 2): it is correct and up-to-date by construction with the next benefits:

– flexible document generator settings;
– different language support;
– include all blocks and interfaces description;
– include function call tree and etc.

The SCADE model completely defines the expected behavior of the generated code. Code is automatically and directly generated from the models, with the KCG qualified Code Generators: the source code is therefore correct and up-to-date by construction (Fig. 2). Object code verification is based on a sample of source C code constructs that can be generated from SCADE Suite and SCADE Display models and that has to be tested on the target (e.g. on the nature bench before it will be imported on the aircraft equipment).

The key feature of model and generated code is determinism. It means that with the same inputs parameters we receive the same output result, so availability of uncontrolled input information is impossible. Determinism provides the possibility of verification at model level. That to guarantee mentioned important feature model shall meet the next requirements:

Signals are typed i.e. only basic C program language are used (boolean, integer, double, char) and their combination.

Global variable value can be read only.

It's necessary to specify maximum quantity of iterations.

SCADE models are based on structured programming.

From SCADE model we receive generated C code, which meets mentioned requirements and has static memory allocation, independent of hardware platform.

Fig. 3 illustrates comparison of classic programming method in which human factor (difference understanding) affect on additional functions appearance. Debugging of these additional functions is possible only during verification. It means that verification is the most time-taking stage. With SADE code generator it's possible to deployment all verification at the level of SCADE model. In this case it meets the main task of model-based approach - is formalization of text specification - i.e. usage of formal language instead natural.

The code generator qualified for avionic systems translate models into embedded C code is DO-178C compliant and allows shortening the certification process of avionics projects which make use of it. Using such a code generator allows the end user (the one that develops the critical embedded application) to reduce the development costs by avoiding the verification that the generated code implements the SCADE model (considered here as a specification). The verification and validation activities are reduced to provide evidence that the model meets the functional requirements of the embedded application.

In this way, a large part of the certification charge weighs on the SCADE framework and this charge is shared (through the tool provider) between all the projects that make use of this technology.

When the SW design and verification is completed, the code can be compiled and loaded on the target. ARINC 653 configuration files and "Glue code" must be written to integrate the code generated by SCADE Suite into the actual Modules of the Equipments.

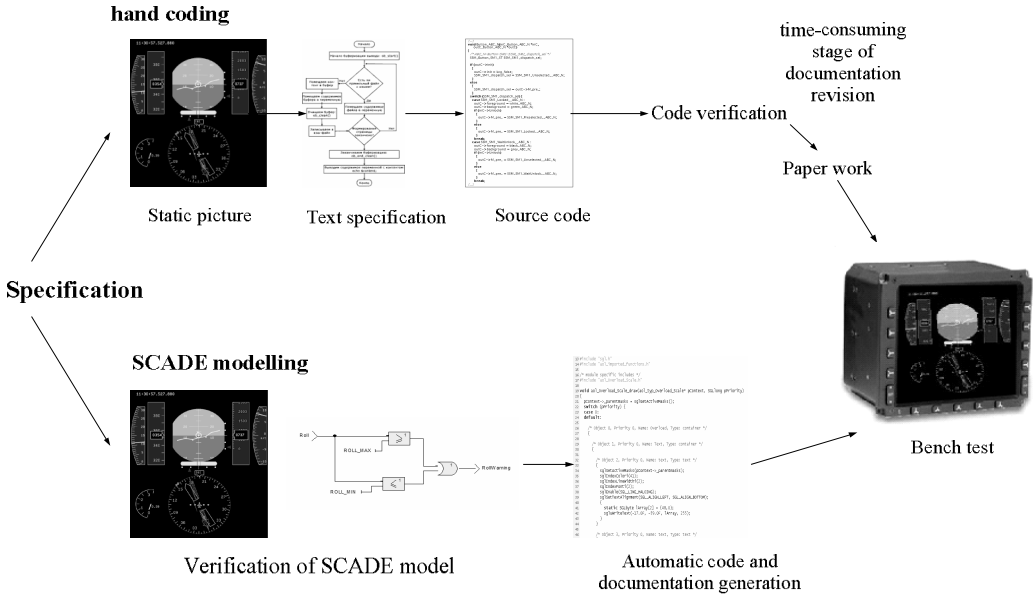

Fig. 3. Two software development process comparison - hand coding and SCADE modelling

VI. CONCLUSIONS

Today's systems and software engineers face mounting pressure to develop automatic flight control systems in a timely and cost effective manner. By utilizing model-based systems engineering, embedded software development processes and qualified code generation, they can meet these challenges. With proposed in this paper approach and its qualified code generators, developers can reduce development time and speed time-to-certification.

Application of the mentioned software development technology allows considerably reduce its development and maintenance due to effective organization of the most labor-intensive process – verification. It allows reduce costs of all software life cycle no less than 40 % as compared to the hand coding [2]. This economy consists of the followings factors: 1) development of the detailed specification, determination of software architecture and verification are passing at the level of model, it allows for saving the time spent on significant verification efforts because models can be verified as soon as they are available (even in parts) thus avoiding situations where code has to be developed before any verification can start and every error that is detected requires a lengthy change cycle; 2) replacement of the hand coding with code generator and automatic code verification, that allows to conduct the module and integral testing at the level of model; 3) automation of writing considerable part of project documentation, that allows to reduce certification costs, especially during alteration.

Ultimately, with this model-based approach models of software components are naturally designed and verified to address the requirements. These components are easily traced to requirements and are well-defined in terms of their interfaces and behavior. This subject of modularity and design for re-use is another paper topic on its own, however this proven FMBD methodology ensures that model-based components, with their associated requirements and documentation, are more easily created and maintained in formal models than traditional manual coding and development methodologies. The modularity provided by FMBD increases platform agility, platform extensibility, and component reuse in multiple aircraft and ground systems, while at the same time reducing cost of deployment and maintenance of the software [3].

REFERENCES

[1] G. Ellis, *Observers in Control Systems: A Practical Guid*e, Academic Press, 2002, 264 p.

[2] *Efficient Development of Safe Avionics Display Software with DO-178B Objectives Using SCADE Suite™*: Methodology Handbook, France: Esterel Technologies, 2012, 110 p.

[3] Jim McElroy, Relieving pressure for UAV software development, December 1, 1992.

[4] D. Luenberger, "Observing the State of a Linear System", *IEEE Transactions on Military Electronics*, vol. 8, pp. 74–80. 1964.

[5] D. Luenberger, "Observers for Multivariable Systems", *IEEE Transactions on Automatic Control*, vol. AC-11, pp. 190–197, 1966.

[6] D. Luenberger, "Introduction to Observers", *IEEE Transactions on Automatic Control*, vol. AC-16, pp. 596–602, 1971.

[7] H. Lens, and J. Adamy, "Observer Based Controller Design for Linear Systems with Input Constraints", *Proceeding of the 17th World Congress, International Federation of Automatic Control*, 6–11 July. 2008, pp. 9916–9921.

[8] D. McLean, *Automatic Flight Control Systems*, Prentice Hall, NY, 1990, 593 p.

[9] *Software considerations in airborne systems and equipment certification* (RTCA/DO-178B): DO-178B, December 1, 1992. Washington. D.C. 20036 USA, 1992, 112 p.

[10] V. L. Syrmos, C. Abdallah, P. Dorato, and K. Grigoriadis. "Static Output Feedback – A Survey", *Automatica*, vol. 33, 1997, pp. 125–137.

[11] C. C. Tsui, "Observer Design – A Survey", *International Journal of Automation and Computing*, vol. 12 (1), 2015. pp. 50–61.

Received June 4, 2015

Prosvirin Dmitry. Post-graduate student.
Department of Air Navigation Systems, National Aviation University, Kyiv, Ukraine
Education: National Aviation University, Kyiv, Ukraine (2010).
Research interests: navigation and flight control.
Publications: 10.
E-mail: dimitry.prosvirin@gmail.com

Kharchenko Volodymyr. Doctor of Engineering. Professor.
Holder of a State Award in Science and Engineering of Ukraine.
Acting Rector of the National Aviation University, Kyiv, Ukraine.
Head of the Department of Air Navigation Systems, National Aviation University, Kyiv, Ukraine.
Professor of Traffic College of Ningbo University of Technology, Ningbo, China.
Education: Kyiv Civil Aviation Engineers Institute with a Degree in Radio Engineering, Kyiv, Ukraine (1967).
Research area: management of complex socio-technical systems, air navigation systems and automatic decision-making systems aimed at avoidance conflict situations, space information technology design, air navigation services in Ukraine provided by CNS/ATM systems.
Publications: 474.
E-mail: knarch@nau.edu.ua

Д. А. Просвірін, В. П. Харченко. Оптимізований підхід до розробки програмного забезпечення з критичними вимогами по безпеці для системи автоматичного керування у відповідності до стандартів ARP 4754, DO-178C

Представлено новий модельно-орієнтований підхід до розробки програмного забезпечення для систем автоматичного керування літака. Показано реалізацію вимог до програмного забезпечення бортових систем у відповідності до стандартів ARP 4754, DO-178C. У статті продемонстровано як вказані вимогу можуть бути виконані з використанням технології SCADE. Показано можливість генерації текстової документації та кваліфікованого коду. Використання представленого підходу дозволяє полегшити та значно прискорити процес розробки та сертифікації програмного забезпечення бортових систем.

Ключові слова: система автоматичного керування; модельно-орієнтоване проектування; ARP 4754, SCADE; генерація коду; сертифікація програмного забезпечення; виконувана специфікація, верифікація, документація.

Просвірін Дмитро Андрійович. Аспірант.
Кафедра аеронавігаційних систем, Національний авіаційний університет, Київ, Україна.
Освіта: Національний авіаційний університет, Київ, Україна (2010).
Напрямок наукової діяльності: навігація та управління рухом.
Кількість публікацій: 10.
E-mail: dimitry.prosvirin@gmail.com

Харченко Володимир Петрович. Доктор технічних наук. Професор.
Лауреат Державної премії України в галузі науки і техніки
Виконуючий обов'язки ректора Національного авіаційного університету, Київ, Україна.
Завідувач кафедри аеронавігаційних систем, Національний авіаційний університет, Київ, Україна.
Професор Traffic College of Ningbo, University of Technology, Ningbo, Китай.
Освіта: Київський інститут інженерів цивільної авіації, Київ, Україна (1967).
Напрямок наукової діяльності: управління складними соціально-технічними системами, аеронавігаційними системами та автоматичними системами прийняття рішень, спрямованих на запобігання конфліктних ситуацій,

створення інформаційних технологій аерокосмічних систем, аеронавігаційне обслуговування польотів в Україні на основі супутникових систем CNS/ATM.
Кількість публікацій: 474.
E-mail: knarch@nau.edu.ua

Д. А. Просвирин, В. П. Харченко. Оптимизированный подход к разработке программного обеспечения, с критическими требованиями к безопасности, для системы автоматического управления в соответствии со стандартами ARP 4754, DO-178C

Представлен новый модельно-ориентированный подход к разработке программного обеспечения для систем автоматичного управления самолета. Показана реализация требований к программному обеспечению бортовых систем в соответствии со стандартами ARP 4754, DO-178C. В статье продемонстрировано как указанные требования могут быть выполнены с использованием технологии SCADE. Показана возможность генерации текстовой документации и квалификационного кода. Использование представленного подхода позволяет облегчить и значительно ускорить процесс разработки и сертификации программного обеспечения бортовых систем.

Ключевые слова: система автоматического управления; модельно-ориентировочное проектирование; ARP 4754, SCADE; генерация кода; сертификация программного обеспечения; исполнительная спецификация, верификация, документация.

Просвирин Дмитрий Андреевич. Аспирант.
Кафедра аэронавигационных систем, Национальный авиационный университет, Киев, Украина.
Образование: Национальный авиационный университет, Киев, Украина (2010).
Направление научной деятельности: навигация и управление движением
Количество публикаций: 10.
E-mail: dimitry.prosvirin@gmail.com

Харченко Владимир Петрович. Доктор технических наук. Профессор.
Лауреат Государственной премии Украины в области науки и техники
Исполняющий обязанности ректора Национального авиационного университета, Киев, Украина.
Заведующий кафедрой аэронавигационных систем, Национальный авиационный университет, Киев, Украина.
Профессор Traffic College of Ningbo, University of Technology, Ningbo, Китай.
Образование: Киевский институт инженеров гражданской авиации, Киев, Украина (1967).
Направление научной деятельности: управление сложными социально-техническими системами, аэронавигационными системами и автоматическими системами принятия решений, направленных на предотвращение конфликтных ситуаций, создание информационных технологий аэрокосмических систем, аэронавигационное обслуживание полетов в Украине на основе спутниковых систем CNS/ATM.
Количество публикаций: 474.
E-mail: knarch@nau.edu.ua