УДК 629.735.33.054.07:681.3.06

Prosvirin D.A.

ANTONOV, State-owned Enterprise. Ukraine, Kyiv

DEPLOYMENT OF MODEL-BASED DEVELOPMENT PROCESS OF SAFE APPLICATION SOFTWARE FOR SAFETY-CRITICAL AVIONICS SYSTEMS

This article deals with model based embedded software development of avionics systems within early field trial. Realization of air-borne equipment software requirements, regulated by functional safety standards is showed. This article explains how mentioned requirements can be obtained using SCADE. The possibility of mentioned approach practical application is showed. The possibility of documentation and qualified code generation from SCADE Display and SCADE Suite models is showed. Use of the mentioned approach allows to facilitate embedded software development and certification process for safety-critical avionics systems.

Keywords: model-based design, SCADE, code generation, software certification, executable specification, verification, documentation, DO-178B.

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-178B/C 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 display industry. The function and architecture of a system are defined by system engineers using some informal notation for the graphics and the logic associated with the displays. 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.

Problem Formulation

Today aviation companies gradually come to model-based embedded software development process for avionics systems. So, search for new applications which 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 generate the code from this model using a qualified code generator, in the sense of DO-178B/C, resulting in the following advantages to the development life cycle [1]:

- It fulfills the needs of the system engineers by providing notations to formally define the system including its behavior and graphical layout.
- It fulfills the needs of the software engineers by supporting the accurate definition of the software requirements and by providing efficient automatic code generation of software having the qualities that are expected for such applications (i.e., efficiency, determinism, static memory allocation, etc.).
- It allows for establishing efficient new processes to ensure that safety criteria are met.
- It saves coding time, as this is automatic.
- It saves a significant part of verification time, as the use of such tools guarantees that the generated source code conforms to the software models.
- It allows for identifying problems earlier in the development cycle, since most of the verification activities can be carried out at model level.
- It reduces the change cycle time, since modifications can be done at model level and code can automatically be regenerated.

Solution Procedure

Today, there is a heightened interest of aviation industry in software certification of aerospace systems according to RTCA/DO-178B. Compliance of aircraft systems to specified regulations as well as the costs for the design and development of equipment force developers to search for new ways to improve the design and meet requirements of these standards. In addition, the U.S. Federal Aviation Administration (FAA), European Aviation Safety Agency (EASA) and other international authorities of aviation safety insist on using these standards to ensure the proper functioning of the aircraft electronics systems in any foreseeable conditions, to exclude the faulty operation and aircraft accidents.

Correspondence to mentioned requirements can be achieved by SCADE (Safety Critical Application Development Environment), model-based technology for the development of safety-critical avionics software that includes the following components: SCADE Display for the design of embedded displays graphics and SCADE Suite for the design of display's logics.

This approach addresses the issue of cost and productivity in the development of safe embedded software for avionics applications. Such projects, driven by the DO-178B guidelines, traditionally require very difficult and precise development tasks, incurring high verification efforts. Mentioned approach reviews the regulatory guidelines and then presents the optimization of the development and verification processes that can be achieved with the SCADE methodology and tools that supports the automated production of a large part of the development life-cycle elements. The effect of using SCADE together with its qualified Code Generator is presented in terms of savings in the development and verification activities, following a step-by-step approach and considering the objectives that have to be met at each step

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 the cockpit display system 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 displays during the system 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, e.g. for primary flight display control algorithm, such as the one illustrated in Figure 1.

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 Display application. In mentioned example executable SCADE model was development for primary flight display frame of hypothetic civil aircraft. It is now helpful to dynamically exercise the behavior of a SCADE Display/SCADE Suite model to better verify how it functions. 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, we 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 pro-

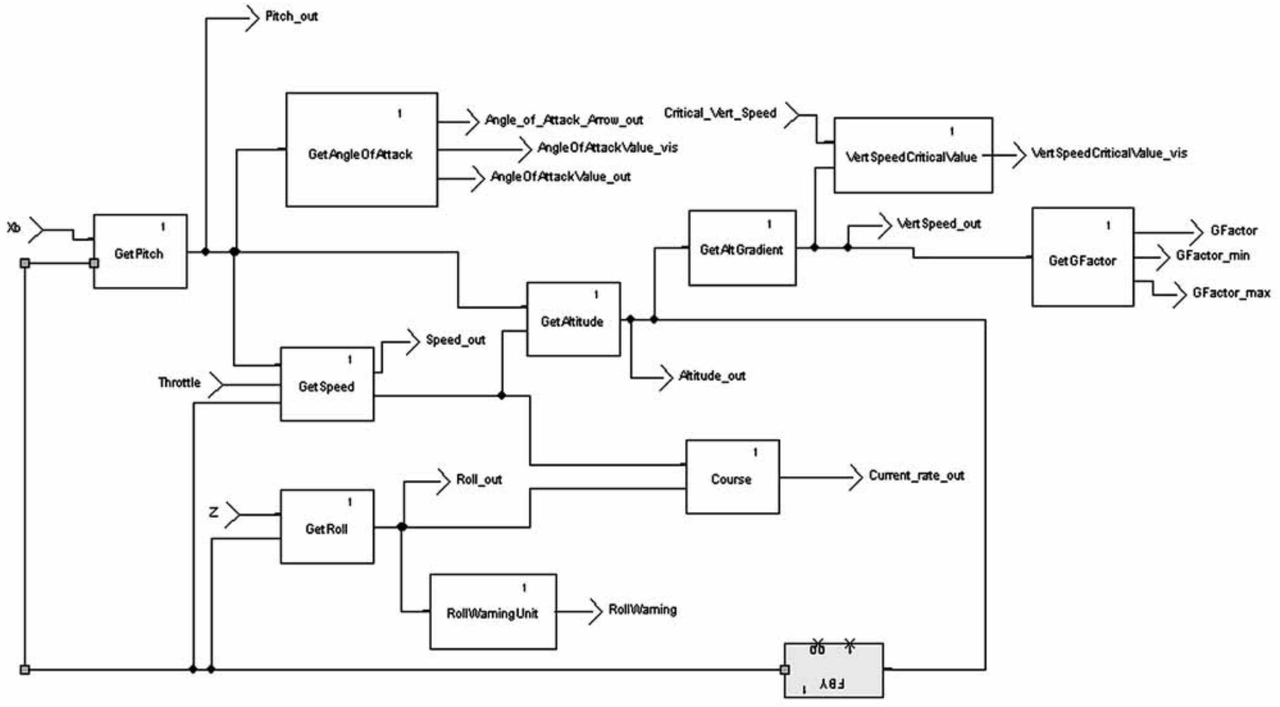


Figure 1. Scade-model control algorithm

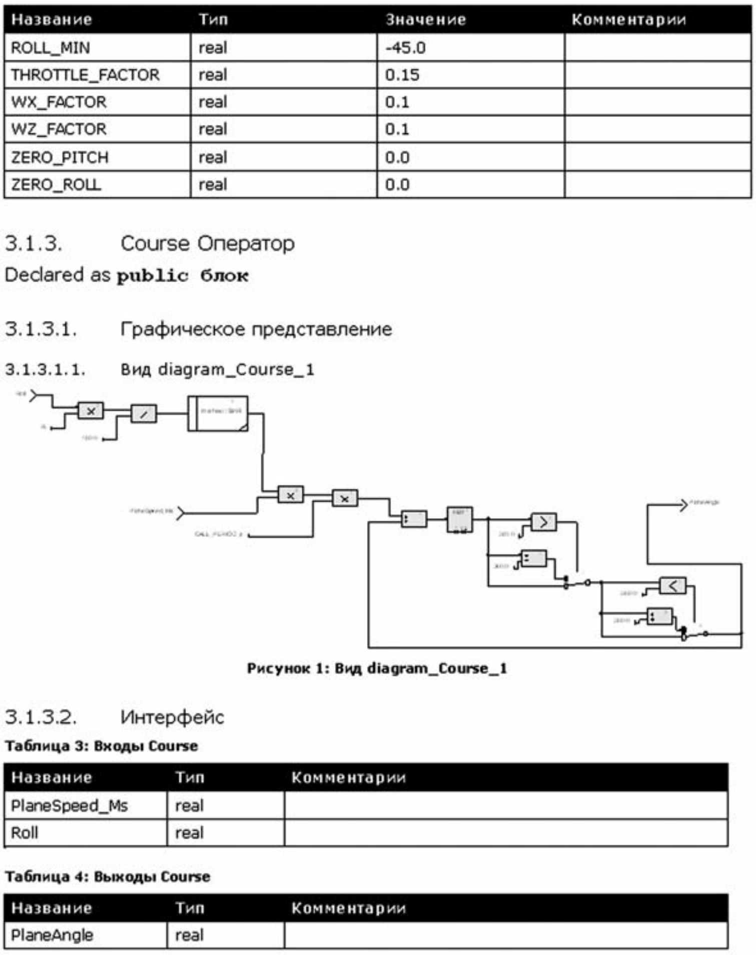
ject are described as executable model which can be sent to aircraft electronic flight instrument system designer. For more clearness, usability and reporting there is a possibility to generate text specification from SCADE model that can be then agreed and signed between customer and developer (if it's 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;
- Russian language support;
- Include all blocks and interfaces description;
- Include function call tree and etc.

The SCADE model completely defines the expected behaviour 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. 3). 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



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	

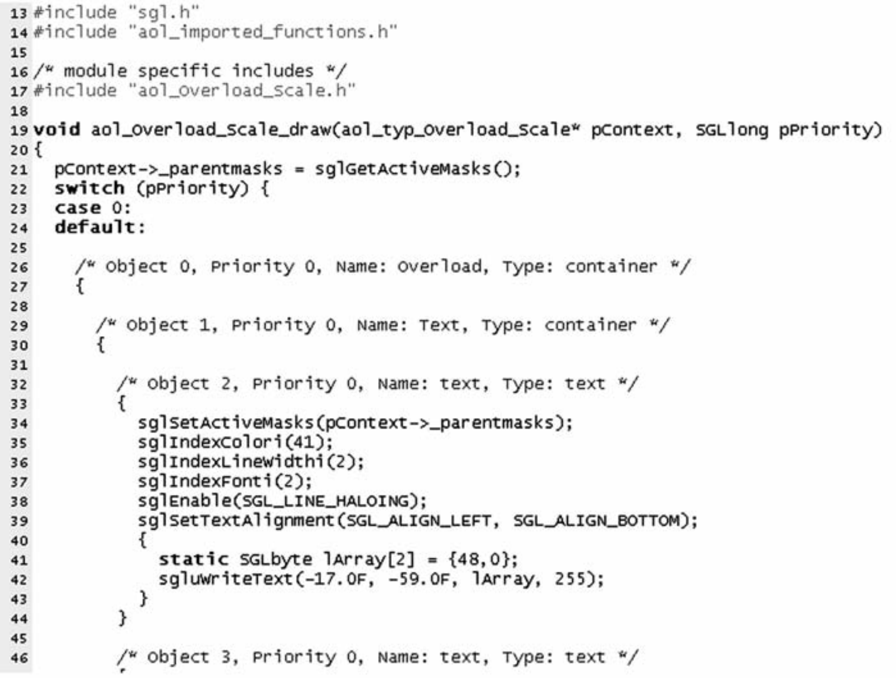
Figure 2. Generated documentation

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 on hardware platform.

Figure 4 illustrates comparison of classic programming method in which human factor (difference understanding) affects on additional functions appearance.



```
#include "sgl.h"
#include "aol_imported_functions.h"

/* module specific includes */
#include "aol_Overload_Scale.h"

void aol_Overload_Scale_draw(aol_typ_Overload_Scale* pContext, SGLlong pPriority)
{
  pContext->_parentmasks = sglGetActiveMasks();
  switch (pPriority) {
  case 0:
  default:

    /* Object 0, Priority 0, Name: Overload, Type: container */
    {

      /* Object 1, Priority 0, Name: Text, Type: container */
      {

        /* Object 2, Priority 0, Name: text, Type: text */
        {
          sglSetActiveMasks(pContext->_parentmasks);
          sglIndexColori(41);
          sglIndexLineWidthi(2);
          sglIndexFonti(2);
          sglEnable(SGL_LINE_HALOING);
          sglSetTextAlignment(SGL_ALIGN_LEFT, SGL_ALIGN_BOTTOM);
          {
            static SGLbyte lArray[2] = {48,0};
            sgluWriteText(-17.0F, -59.0F, lArray, 255);
          }
        }

        /* Object 3, Priority 0, Name: text, Type: text */
```

Figure 3. Automatically generated code fragment

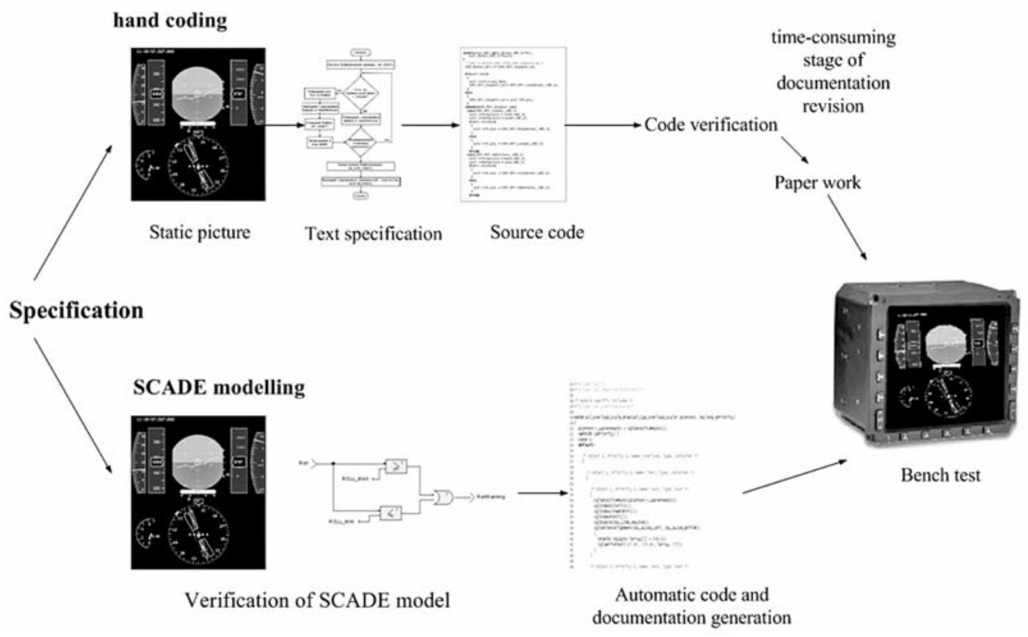


Figure 4. Two software development process comparison - hand coding and SCADE modeling

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.

Conclusion

The avionics industry has a very long tradition. The function and architecture of an embedded computer system (i.e., Flight Control, Braking, Cockpit Display, etc.) are defined by system engineers; the associated control laws are developed by control engineers using some informal notation or a semi-formal notation mainly based on schema-blocks and/or state machines; and the embedded production software is finally specified textually and coded by hand in C or Ada by software engineers.

This article addresses technology of software development that allows engineers to facilitate development and maintenance process due to effective organization of the most labour-intensive stage – verification. It allows reduce costs of all software life cycle no less than 40 % as compared to the hand coding [3]. This economy consists of the followings factors: 1) development of the detailed specification. Determination of the software architecture and verification are passing at the level of model. It allows to save the time, that usually takes to significant verification efforts (models can be verified as soon as they are available even in parts) and avoid situations when code is developed before any verification can start and every detected errors require a lengthy change cycle; 2) replacement of the hand coding with code generator and automatic code verification, that allows to conduct the module testing and integral testing at the level of model; 3) automation of writing project documentation, that allows to reduce certification costs, especially during alteration.

References

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

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

[3] Shligerski A., Model-based development of safe application software for safety-critical railways systems using scade tool environment/ Shligerski A., Umanski V. — M.: Functional safety – theory and practice, 2009. — pp. 13–21.

[4] Myers J. Software Reliability – Principles and practices/ Myers J. — N.Y.: IBM Systems Research Institute Lecture in Computer Science, Polytechnic Institute, 1976. — 360.

УДК 629.735.33.054.07:681.3.06

Просвирин Д.А.

Государственное предприятие «Антонов». Украина, г. Киев

ПРИМЕНЕНИЕ МОДЕЛЬНО-ОРИЕНТИРОВАННОГО ПОДХОДА ДЛЯ РАЗРАБОТКИ БЕЗОПАСНОГО ПРОГРАММНОГО ОБЕСПЕЧЕНИЯ АВИАЦИОННЫХ СИСТЕМ С КРИТИЧНЫМИ ТРЕБОВАНИЯМИ К БЕЗОПАСНОСТИ

В статье представлен модельно-ориентированный подход к проектированию встроенного программного обеспечения для авиационных систем с критичными требованиями к безопасности при выполнении пилотного проекта. Показана реализация требований к программному обеспечению бортовой аппаратуры, регламентируемые стандартами по функциональной безопасности. Показана возможность практического применения указанного подхода. Рассмотрена возможность автоматической генерации исходного кода и документации из SCADE Display та SCADE Suite моделей. Применение указанного подхода позволяет значительно ускорить процесс проектирования и дальнейшей сертификации программного обеспечения.

Ключевые слова: *модельно-ориентированный подход, SCADE, генерация кода, сертификация программного обеспечения, исполняемая спецификация, верификация, документация, DO-178B.*