A TOOL FOR DEVELOPING AND VERIFYING COMPONENTS OF CROSS-PLATFORM ON-BOARD SOFTWARE [1]

Mikhail V. Saramud, Igor V. Kovalev, Vasiliy V. Losev, Anna A. Voroshilova

Reshetnev Siberian State University of Science and Technology, Krasnoyarsk,
e-mails: msaramud@gmail.com, basilos@mail.ru
Russian Federation

Abstract: The article describes a tool for developing and verifying components of cross-platform on-board software. It consists of a database, the component program code editor and the module of the integrated development environment, which allows editing component attributes, verifies the compliance of component attributes with the development design architectural plan, and helps to search by attributes. The database stores both the formalized attributes of the developed software component and the informal description of its functionality. The implemented methods of unified component construction and the proposed tools of the development environment provide the possibility of component verification at the design stage and further identification of the developed components. Storing the attributes of software components allows to identify them for reuse effectively. It also allows one search for components using a specific queue, physical port or peripheral device.

Key words: Integrated Development Environment; on-board software; real-time operating system; execution environment.

1. INTRODUCTION

Development of on-board software (OBS) for unmanned aerial objects has its own specifics. The paper discusses the method proposed for creating a cross-platform OBS, described earlier in [1]. The method implies the use of the real-time operating system FreeRTOS with a number of modifications, including the decision block necessary for multi-version software. Since the software runtime is defined, there remains a task of developing the functional modules or the software components.

[1] This work was supported by Ministry of Education and Science of Russian Federation within limits of state contract № 2.2867.2017/4.6

In our case, the component of cross-platform on-board software implies a function in C, written according to certain requirements [1], which runs as a flow of the real-time operating system when run by the OBS. To ensure the possibility of designing, programming and verification of cross-platform OBS components, which guarantee their quality, both in the development process and in the process of long-term maintenance a development tool is proposed [2]. It is a module of the integrated development environment of cross-platform OBS - a program developed in C #, having a convenient window interface and the ability to work with databases (in this implementation MySQL server of version 5.7 is used).

2. METHOD OF VERIFYING COMPONENT ATTRIBUTES

To increase the quality of cross-platform OBS component development, it is also necessary to solve the problem of verifying component attributes for compliance with the requirements of a development project [3]. For informational support of the development process, a database will be used, which will allow not only to provide verification, but also to further facilitate the search and identification of previously developed components by their attributes.

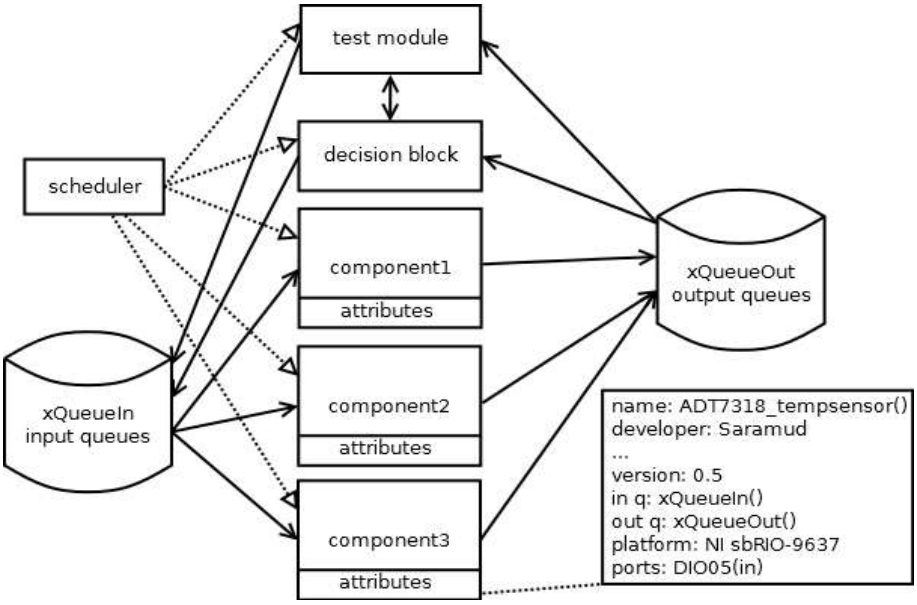


Fig. 1. Project Subsystem Architecture

Figure 1 shows the architecture of the test assembly of the project subsystem, which includes the scheduler, queues, decision block, the executable components themselves as well as the test module necessary to carry out the process of functional testing of components in a real execution environment. Each component

has its own attributes. The image allows us to compare the attribute values and architecture, for example, the input and output queues of the component.

In the process of developing components of cross-platform on-board software, the database will be populated with their functional attributes, by which it is possible to perform the functional identification of the desired module in the future. The database stores all important component attributes, such as: target hardware platform, physical ports used, queues, etc. The data scheme is shown in Figure 2.

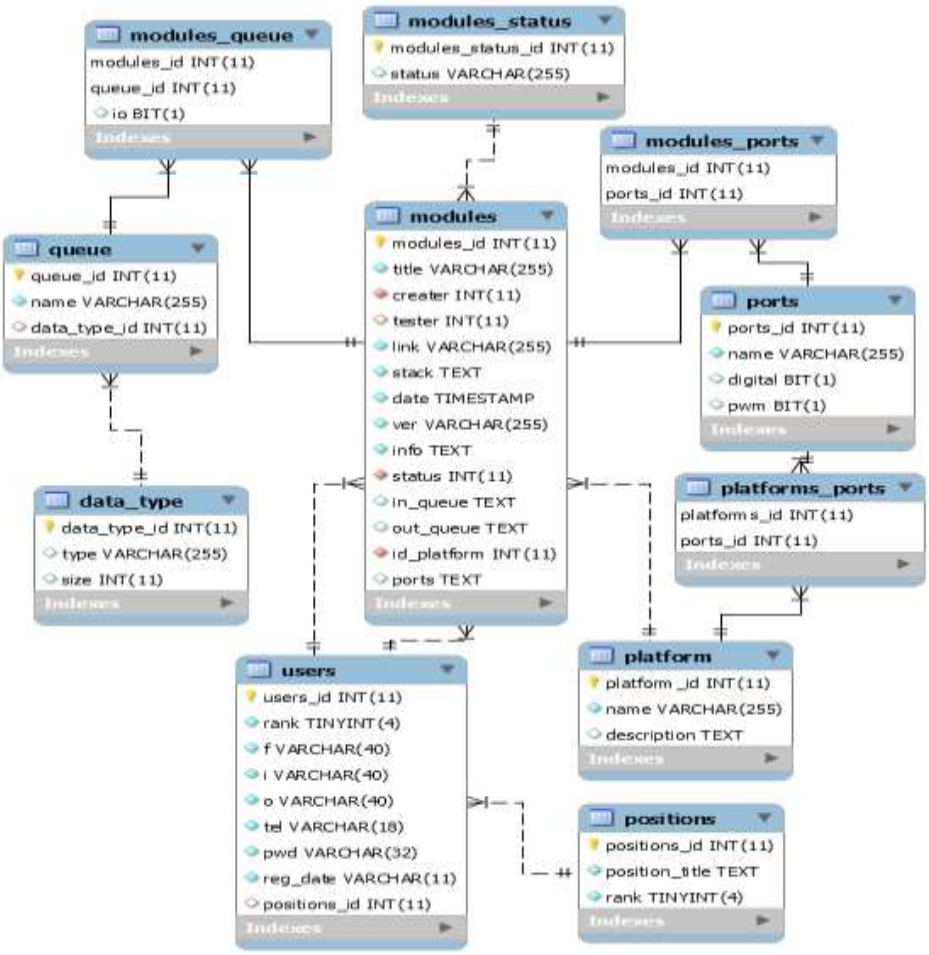


Fig. 2. Relationship scheme for entities responsible for the description of a specific component

To solve the tasks, a tool for developing and verifying components of a cross-platform OBS was proposed, the implementation of which is described below.

3. SOFTWARE IMPLEMENTATION OF THE PROPOSED TOOL

Start of the program launches a choice of running module. It is possible to create a new project, open the source editor of the modules separately, create a module, and add a hardware platform, a queue or a hardware port to the database.

If it is necessary to create a new component, the appropriate option is selected; the component creation module is launched.

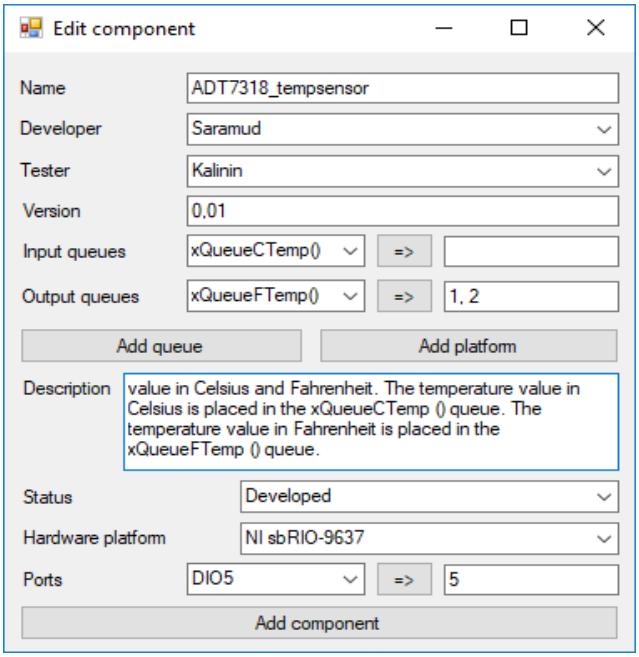


Fig. 3. Component creation module.

The window of the component creation module (Figure 3) indicates its main attributes:

• A headline that will also be the name of the component file with the extension ".c" and the name of the function;

• Developer - the person directly responsible for the development of the program code of the component;

• Tester - the person responsible for testing the developed component. The list of employees to select a developer and tester is loaded from the database and is selected from the drop-down list;

• Version - the number of the initial version of the component, which will be changed during the development process;

• Queues of input and output data are also selected from the drop-down list received from the database, if necessary, it is possible to add a new queue to the database. There is a button for calling the corresponding module;

• Description - an informal description of the component, which is necessary not only for the convenience of the developer, but also when searching and reusing ready-made components;

• Status - current component status, selected from the component status table entries in the database;

• The hardware platform is the target platform to which the developer is oriented and to which functional testing will be performed. It is selected from the list of available database platforms, if necessary, one should add a new platform to the database. There is a button for calling the corresponding module.

• Ports - the physical ports of the hardware platform used by the component.

After filling in all the required fields, the button "Add component" is pressed. The main record is added to the base - to the component table (Figure 4).

moc	title	crea	link	stack	date	ver	info	stati	id_p
26	STM-06-230_Servo	3	/ProtoProject/STM...	0,2=>7	2018...	0.5	Serovdrive control compon...	1	1
27	ADT7318_temps...	3	/ProtoProject/ADT...	=>1,2	2018...	0.1	This module must be conne...	1	3
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

Fig. 4. Component table in the database.

The command of adding a new record to the database returns the id of the added record. Having obtained the id of the created module, one can fill in all the intermediate tables (Figure 5), which are necessary to break the many-to-many relationship [4], since each of the components can use multiple queues and ports, and the same queues and ports can be used in a variety of components.

	modules_id	queue_id	io
	26	11	0
	27	1	0
	27	2	0
*	NULL	NULL	NULL

a)

	modules_id	ports_id
	25	1
	26	1
	27	4
*	NULL	NULL

b)

Fig. 5. Component connection table with a) queues b) ports in the database.

The Components-Queues table (Figure 5a) stores the components and queues id, which together are the primary key for this table, as well as the "io" parameter of the BIT type, which remembers whether the queue for the component is input or output (0 - if output queue, 1- if input). This parameter is necessary, since the same queue can be for one output component, where it will write data, and for another input component, from where it will take data.

The Components-Ports table (Figure 5b) stores the id components and physical ports, which together are the primary key for this table. This table is necessary not only in the module development process, but also for the convenience of project support and component identification - it can be used to obtain a list of all components that interact with certain ports.

4. EDITOR OF COMPONENTS

After finishing work with the database, the component file itself is created in the project folder, with the name specified on the form and the extension ".c". Some initial parameters are automatically written to the file. A comment is written about the description of the module. The function of the component with the selected name and the main infinite loop (a feature of the functioning of threads in FreeRTOS [5]) are declared. The first line in the loop is the vTaskSuspend (NULL); command, which will send the flow to the standby mode, after its variables necessary for operation are initialized. It automatically announces the reading of selected input queues and the writing of output data to the queues. Comments mark places for declaring variables and the main functional code of a component.

After creating a component file on the disk, it automatically opens in the editor (Figure 6).

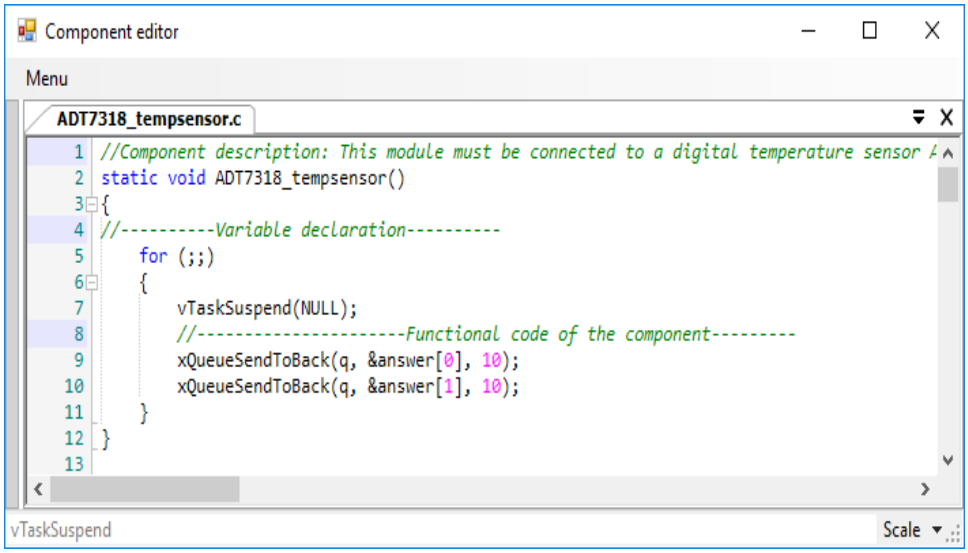


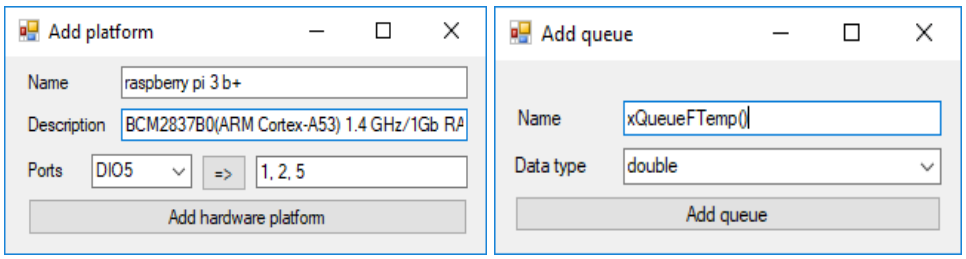
Fig. 6. Component source editor.

The built-in source code editor is based on the FastColoredTextBox component [6], supports syntax highlighting of the C language, changing the font and display scale, collapsing code blocks (searching for available areas for folding automatically), text search, many combinations of hotkeys, automatic indentation, bookmarks, tooltips, alignment of individual characters in lines, also supports auto-completion.

The development of components of cross-platform on-board software takes place directly in the editor.

5. ADDITIONAL MODULES OF THE TOOL

If the database does not have the necessary hardware platforms, queues or ports, one can add them with the appropriate tools (Figures 7-9).



a) *b)*

Fig 7. The module of adding a) hardware platform, b) queue into the database.

Figure 7a shows the interface of the module for adding a hardware platform, on which the platform name is set, its description and it is possible to immediately set all the physical ports of the platform being added. The list of physical ports is loaded from the database.

Figure 7b shows the interface for the add-queue module, which defines the name of the queue and the type of stored data. The list of available data types is loaded from the database.

Thus, the paper illustrates the process of creating a component and its attributes, describes the actions that occur in the process in the database. The interface of the proposed tool is presented.

6. CONCLUSION

The developed unified development tool for the cross-platform OBS components is a convenient tool for the developer, and due to the connection with the database and the implemented ability to store multiple component attributes, verification is provided at the design stage and further identification of the developed components.

The implemented methods for building unified components and the designed development environment tools provide the required level of component verification for compliance with the architectural and detailed design.

Storing the attributes of software components allows us to effectively identify them for reuse. It also allows searching for components using a specific queue, physical port or peripheral device.

The information stored in the database allows the implementation of an automated testing system for software components, since the sources of input data, queues for the results of calculations, physical ports, and hardware are clearly indicated.

REFERENCES

[1] Mikhail V. Saramud, Igor V. Kovalev, Vasiliy V. Losev, Peter A. Kuznetsov, Software interfaces and decision block for the execution environment of multi-version software in real-time operating systems, *International Journal On Information Technologies And Security*, №1 (vol. 10), 2018, pp. 25-34.

[2] Saramud M.V., Kovalev I.V., Losev V.V., Petrosyan M.O., Application of FreeRTOS for implementation of the execution environment of real-time multi-version software, *International Journal on Information Technologies And Security*, №3 (vol. 10), pp. 75-82.

[3] Leticia Fuentes-Ardeo, Jose Ramon Otegi-Olaso, Maria Eugenia Aguilar-Fernandez, How the project knowledge management and the sustainability in project management affect the project success, *9th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications (IDAACS)*, 2017, volume 2, pp. 884 – 887.

[4] Thomas Nield, Getting Started with SQL: A Hands-On Approach for Beginners, O'Reilly Media, Inc., 2016, 134 p.

[5] Hsuan Hsu, Chih-Wen Hsueh, FreeRTOS Porting on x86 Platform, *2016 International Computer Symposium (ICS)*, 2016, pp. 120-123.

[6] Fast Colored TextBox for Syntax Highlighting, https://github.com/PavelTorgashov/FastColoredTextBox (current January 2019)

Information about the authors:

Mikhail Vladimirovich Saramud - Junior Researcher of research department, Reshetnev Siberian State University of Science and Technology. Areas of Research are fault-tolerant software, system analysis;

Igor Vladimirovich Kovalev – Professor at the Department of systems analysis and operations research, Reshetnev Siberian State University of Science and Technology. Areas of Research are fault-tolerant software, system analysis;

Vasiliy Vladimirovich Losev – Associate Professor at the Department of automation of production processes, Reshetnev Siberian State University of Science and Technology. Areas of Research are automation control systems, system analysis;

Anna A. Voroshilova – Associate Professor of Reshetnev Siberian State University of Science and Technology. Areas of Research are automation control systems, system analysis.

Manuscript received on 14 January 2019