UDC 629.78.018

I.B. TURKIN, P.A. LUCHSHEV

The National Aerospace University "KhAI", Ukraine

FORMAL MODEL OF TECHNOLOGICAL PROCESSES'S DESCRIPTION OF COMPLEX TECHNICAL SYSTEMS'S TESTS

Theoretical aspects of application of the problem-oriented language for description the tests of the technological processes of complex technical systems are considered. The formal grammar of language and features of use in practice for specification (declaration) of technological processes the tests of satellite power supply system (PSS) is given.

tests, technological process of tests, formal language, formal grammar, problem-oriented language, complex technical systems

Introduction

One of the first and more important problems in the technological preparation of manufacture is the insufficient level of intellectual support for designer work by modern tools and their weak integration with automatized process control systems.

For the complex solution of all problems connected to designing, manufacturing and working of products, uniform methods and means for the decision of all tasks of designing and technological preparation of manufacture in frameworks of constantly improved automated systems is necessary and must be used [1].

1. General problem

In development process of complex technical systems, such as satellites, one of the important stages of life cycle are tests, that be explained by specificity of manufacture and functioning. Also for technological processes of test in difference from technological processes of manufacture demand the greater flexibility of hardware and software. In realization problem of tools for preparation of technological processes of tests and the organization of interaction with levels of storage and execution the major question is a dialogue language. This language should be problem-oriented without restrictions for perception of the information by human and should be effective for work on computer in real-time too. Experience of scientific work and professional researches in different areas demonstrate, that the problem-oriented language may provide the good ability for perception, transfer and storage of the information [2].

Actually use in the specialized spheres of activity of the problem-oriented language with target of the best communicativeness realization for search, storage and processing of the information, is necessary condition for increase of interaction accuracy. The problem-oriented language is based on syntax of natural language and thesaurus of subject area usually, but actually it is required for strict formalization of scientific descriptions unlike full language.

2. The review of unsolved problems

Modern SCADA systems use five specialized languages defined at standard IEC 61131-3 (IL, ST, LD, FBD, SFC), also many systems functioning on platform Windows, have built-in support VBScript. The basic purpose of use of these languages is reduction of expenses on preparation of programmers, creation of programs of the big sizes, realization of more and more complex hardware-software systems [3].

However actually from the moment of occurrence of the standard were marked a problem of human perception textual (IL, ST) and graphic (LD, FBD, SFC) forms

[4], and some other problems connected with cross-compilation from one format in another, distinctions in strictness of standards representations in graphic and textual format [5]. The next step of development of this standard of languages was directed on tools for distributed systems that are reflected in standard ISO 61499.

Taking into account mentioned above, it is possible to declare, that modern SCADA systems have well advanced tools of execution and management of technological processes, and also toolkit for creation of program complexes for already existing technological process of manufacture. However, when speech comes about a computerization of engineering activity of the technologist and about toolkit of development of technological processes available now in SCADA systems do not give ready decisions. Except for it, the given problem gets in area CAPP where at present the basic direction is the automated development of technological processes of machining which, in turn, receives the information from CAD. For complex technical systems, which cope microprocessors and have electric interfaces, special importance is got with information properties for which check it is necessary to carry out tests adequate to the physical nature and with the technological process which essentially differs from mechanical operations of manufacturing and assembly technological processes.

Technological processes of tests which dynamically may is changed. In this situation when or the technologist should have skills of programming or the programmer to be the technologist is created. This situation has a negative effect for quality of work. In this case the tools of automatization and a computerization of engineering activity of the technologist may serve as the compromise for developing technological processes of tests.

3. Task

The basic purpose is automatization of technological process of tests of the satellite power supply subsystem. Taking into account high variabilitiness of requirements to technological process of tests, some uncertainty on technical parameters and a high degree of parallelism of the satellite development and tools of its tests, on two parts software splitting was accepted: the automatized preparation of technological process of tests and its automatic execution. For achievement of an object in view at the first stage primary goal becomes development of the problem-oriented language which would allow to formalize and organize interaction of all levels of a computerization of engineering activity of the technologist, would allow to store the information on technological process and was easily and simply broadcast in necessary representation.

Such formalization, will allow organizing independent development of means of interaction of all levels of the engineering activity necessary as for manual creation of technological process of test, and for completely automated.

4. Research results

4.1. Concept

Languages interpret as means of dialogue, which consist of sign system (set of allowable sequences of symbols), sets of senses of this system and conformity between sequences of symbols and the declaration doing "sensible" allowable sequences of symbols. As basis for development of language it is possible to use model of the data necessary for realization of the software needed for tests automatization. Actually it's the minimal part of language which real-time required. For editing language of the description of technological process already should support the additional information necessary for perception by human. Syntax of language in which key words are added for its adequate perception by the human is represented below.

4.2. Language syntax

First of all for creation of the problem-oriented language syntax it is necessary to create the dictionary of a subject area and on this basis to define sets of terminal and not terminal symbols.

Thus, at the beginning of development we will reference to the formulation of the standard of the tests term (GOST 16504-81): tests is "experimental definition (estimation and/or checking) quantitative and/or qualitative properties of object of tests at its functioning, at modeling object and/or influence". Thus, it is possible to define key concepts for creation of language:

– qualitative / quantitative properties of object;

– functional properties of object.

According to it for definition of qualitative/quantitative objects properties necessary to execute checks in various modes of functioning and to give out messages about work and results of tests. Performance of checks assumes, at least, the following actions: gathering of the information on the current condition of system, check of the given set of conditions for estimation of a situation and influences control on object of tests. The description of technological process of tests of the complex technical system, controlling a sequence of checks, we shall name the scenario of tests. To make such description it is necessary to characterize dependence of external influences control and reports of information to the operator from the current condition and prehistory of tests object.

4.2.1. Formal grammar of language

As known formal model may be represented the ordered sequence $G = \langle V_T, V_N, S, R \rangle$, and named the formal grammar, where V_T – the terminal or basic symbols; V_N – the nonterminal or auxiliary symbols ($V_T \cap V_N = \varnothing$); S – start symbol or an axiom; R – finite set of rules or production kind $\varphi \rightarrow \psi$, where $\varphi \in (V_T \cup V_N)^* V_N (V_T \cup V_N)^*$, $\psi \in (V_T \cup V_N)^*$ – various chains, and $\rightarrow$ the special symbol not included in $V_T \cup V_N$ to separate the left part of rule φ from right ψ. In rules {} braces mean any recurrence 0…M, and [] square brackets an optional part. In rules bold font are used for **terminals**, and italics for *nonterminals*. The first rule is start symbol.

R = { 1. <*Scenario*> ::= **Test** <*Test Name*> ↵
Modes ↵ <*Mode List*>
Checks ↵ <*Check Set*>
Messages ↵ <*Message Set*> **.**

2. <*Mode List*> ::= <*Mode Num*>**. Mode** <*Mode definition*>
{<*Mode Num*>**. Mode** <*Mode definition*>}

3. <*Check Set*> ::= {**Check** ↵ <*Check definition*>}

4. <*Message Set* > ::= {**Message** <*Message Name*> ↵
Text: <*Message text*> ↵
Type: <*Message type*> ↵
Report type: <*Report message type*> ↵
Priority: <*Priority level*> ↵
[**Params:** <*Param*>{, <*Param*>} ↵] }

5. <*Mode definition*> ::= <*Mode Name*> ↵
Initialization ↵ <*Start sequence*>
Check state ↵ <*Check state change*>
Transitions ↵ {<*Transition*>}

6. <*Check definition*> ::= <*Check Name*> ↵
Actions ↵ <*Action Sequence*>

7. <*Start sequence*> ::= {<*Action*>}

8. <*Check state change*> ::= {**On** | **Off** <*Check Name*> ↵ }

9. <*Transition*> ::= **Goto** <*Mode Name*> ↵
Conditions ↵ <*Condition Set*>
{**Conditions** ↵ <*Condition Set*>}

10. <*Action Sequence*> ::= <*Action*> | **Reaction** ↲ <*Check reaction*>
{<*Action*> | **Reaction** ↲ <*Check reaction*>}

11. <*Action*> ::= **Command** <*Command Name*> ↲
[<*Operator*>]
[**Delay** = <*Value*> ↲]
[**Message** <*Message Name*> ↲]

12. <*Check reaction*> ::= **Conditions** ↲ <*Condition Set*>
Positive ↲ <*Check result*> ↲ [**Message** <*Message Name*> ↲]
Negative ↲ <*Check result*> ↲ [**Message** <*Message Name*> ↲]

13. <*Check result*> ::= <*Check Name*> = <*Value*>

14. <*Condition Set*> ::= <*Condition*> ↲ { <*Condition*> ↲ }

15. <*Operator*> ::= <*Param*> = <*Value*>

16. <*Condition*> ::= <*Param*> <*Relation*> <*Value*> | <*Param*> <*Relation*> <*Param*>

17. <*Message text*> ::= "String{**%S** |String}"

18. <*Param*> ::= <*Operand*>|<*Check Name*>

19. <*Value*> ::= int | float | Bool

20. <*Relation*> ::= < | > | = | >= | <= | <>

21. <*Bool*> ::= **true** | **false**

22. <*Priority level*> ::= int

23. <*Mode Name*> ::= "String"

24. <*Mode Num*> ::= int

25. <*Message Name*> ::= "String"

26. <*Operand*> ::= "String"

27. <*Command Name*> ::= "String"

28. <*Test Name*> ::= "String"

29. <*Message type*> ::= int

30. <*Check Name*> ::= "String"

31. <*Report message type*> ::= int

}. ↲ – end of line, %S –value substitution.

4.2.2. Grammar properties

Estimation of adequacy of the created language for application with the purpose of tests automatization it is necessary to estimate its properties as regular (automatic) or context-free languages allow receiving effective realization of the analysis program. Otherwise it will be necessary to execute equivalent transformations to grammar with the necessary properties.

On Homsky's classification criteria the given grammar is context-free. From viewpoint above, it is possible to say, that language for description of technological process of test is context-free and may be created enough the simple and effective compiler both for development of technological processes, and for implement in real time. Also, the grammar does not contain rules of a conclusion with priority that enables to avoid ambiguous situations, and there are no rules with brackets. That all will allow characterize grammar properties not far from regular grammar.

5. Practical application

5.1. Features of satellite's PSS test

At formation of structure and the testing procedure of a PSS the following features of workmanship, completing and the usual organization of works were taken into account:

1. The PSS include a set of physically diverse components with number of characteristics are determined by mainly external operational influences. That allows use limited number of tests with influences only for separate components without recurrence of similar tests of PSS as a whole;

2. The PSS in structure of the satellite has the advanced interaction with other on-board systems, so

close, that some kinds of tests of consumers of energy may not be carried out without connection with PSS that to do possible to combine some kinds of tests actually PSS with tests of other systems connected to satellite;

3. Cooperation of organizations – manufacturers of a PSS includes a number of the specialized enterprises carried territorially and organizational, so on headquarters plant assembly of PSS with additional equipment in its units of own development is made mainly, in this connection before this assembly components of a PSS should pass a cycle of tests, in particular what it it is inconvenient (sometimes and it is impossible) or is economically inexpedient to carry out on PSS in the collected kind;

4. During existence of practical astronautics the richest statistical material about behavior of PSS in real conditions of operation which, being based on principles of technical similarity, in full or in part it is possible to distribute and to new development of PSS in this connection the part of tests of a PSS may be replaced with calculations or mathematical modeling is saved up.

5.2. Developed tests

With features which are listed above, descriptions of technological processes for test of the PSS satellite were developed (fig. 1).

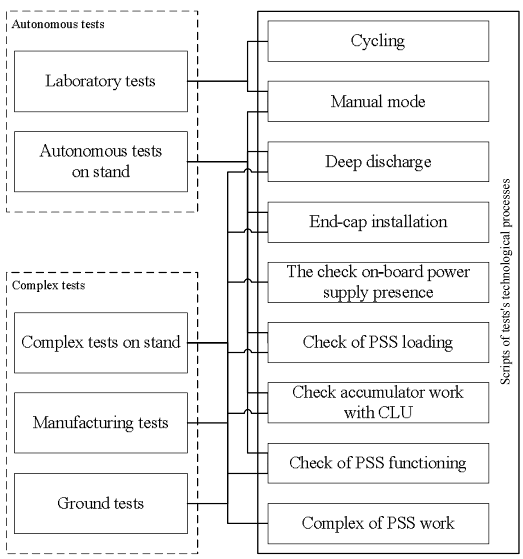

Fig. 1. Tests's scenarios and their kinds

The developed scenarios for PSS tests allow effectively to execute both technological work, and research.

Conclusions

As a result of the carried out works the context-free grammar for language of tests's technological processes was received that has allowed to separate software for tests for two independent subsystems – the program for tests execution and the program for tests's scenarios editing. Such division effectively allows solving questions on updating tests without software code changes.

References

1. К пятилетию стандарта IEC 1131-3. Итоги и прогнозы. – [Электрон. ресурс]. – Режим доступа: http://www.iae.nsk.su/~zyubin.

2. Особенности проектирования систем технологической подготовки производства / А.А. Калинин, В.Ф. Митин, В.Д. Костюков, А.В. Митряев (РКЗ ГКНПЦ им. М.В. Хруничева). – [Электрон. ресурс]. – Режим доступа: http://www.ihst.ru/personal/akm/section11(2002).htm.

3. IEC 61131-3: Programming Industrial Automation Systems: concepts and programming languages, requirements for programming systems, aids to decision-making tools / Karl-Heinz John, Michael Tiegelkamp. – Springer, 2001. – 446 p.

4. WinSPS. Programming according to IEC 61131-3. Software Manual by Bosch Rexroth AG, Erbach / Germany.

5. Лачинов В.М., Поляков А.О. Информодинамика или Путь к миру открытых систем. – С.-Пб.: Изд. СПбГТУ, 1999. – 336 с.

Поступила в редакцию 6.03.2006

Рецензент: д-р техн. наук, проф. В.М. Илюшко, Национальный аэрокосмический университет им. Н.Е. Жуковского «ХАИ», Харьков.