УДК 004.415.2 (045)
Сидорова Н.М.
Національний авіаційний університет

PROGRAMMING STYLES TAXONOMY

Сидорова Н.М. Таксономія стилів програмування. У статті наведено результати дослідження домена стилістики програмування, які представлено таксономією понять, що буде використано для побудови онтології застосування стилів програмування. Онтологія є складовою засобу, який будується згідно запропонованого методу застосування стилів програмування і спрямований на допомогу програмісту.
Ключові слова: програмне забезпечення, програмування, стилі програмування, онтологія, таксономія.

Сидорова Н.Н. Таксономия стилей программирования. В статье приведены результаты исследования домена стилистики программирования, представленные таксономией понятий, которые будут использованы для построения онтологии применения стилей программирования. Онтология является составной частью средства, которое строится согласно предложенного метода применения стилей программирования и направлено на оказания помощи программисту.
Ключевые слова: программное обеспечение, программирование, стили программирования, онтология, таксономия.

Sidorova N. Programming styles taxonomy. The results of the study of programming stylistics domain are presented. The result is the taxonomy. A taxonomy of concepts will be used to construct the programming styles ontology. Ontology is a part of the tools that is constructed according to the own method.
Key words: Software, programming, programming styles, ontology, taxonomy.

Formulation of scientific problem. Application of experience in software engineering plays an important role in improving the efficiency of development and maintenance of software products. Experience is applied through using of software development methods and life cycle models, based on the use of legacy software, and reuse [1, 10].

Application of these methods and models increases the complexity of software and the collective nature of its development and maintenance and requires the use of programming styles. [2, 7-9] Therefore, the study and solution of problems related to the application programming styles for a long time is of particular relevance.

Programming style ensures all processes of creating software, represented by a set of rules expressed by the linguistic resources and reflects prevailing during the software life cycle is not only technical, but also a cultural experience [2, 7-9].

Through collaborative development and reuse the style has a direct and through training - indirectly related to all processes of the software lifecycle. Application of styles in programming means the improvement of the efficiency of development and maintenance of software.

Analysis research. At various times the problem of style programming directly or indirectly was studied by E.Dejkstra, I.Kernigan, F.Plodger, W.Tassel, I.Velbitsky, A.Ershov, I.Pottosin, N.Sidorov. In programming, the concept of style was introduced with the advent of structured programming. I.Kernigan, F.Plodger were the first researches who began to use the style [3]. Later, there were different interpretations of style, for example, by A.Ershov, V.Borovin, N.Sidorov.

There are two approaches to solving problems of application programming style: language-oriented and technology-oriented [4]. The essence of the first approach is based on the assumption that the use of programming style is done by writing the texts of programs by means of a programming language, and hence texts of programs never go beyond language. This approach has the following disadvantages:

- a translator for a changing variety of styles cannot be build;
- some of the rules that describe the style, can not be converted into grammar;
- some of the rules that describe the style, can not be realized only lexical and syntactic computation.

The essence of the second approach is to develop and implement means, processes and methodologies for automation solutions of the style application. In this case, the means to meet the following requirements:

- ensure that the traditional notion of styles;
- implement the necessary actions associated with the use of empirical methods;

- does not depend on the phases of the life cycle.

The database is the basis of the means of applying the programming style. However, the use of databases to represent the domain knowledge shown its limitations and appropriate use of new tools, such as ontologies [5].

Main material and justification of the results. For the first time we propose a method of application programming style based on ontologies, providing greater efficiency through using of style complete, by the way precise and formal description of the domain ontology and using OWL-DL to automate processes associated with its creation and maintenance.

Fig. 1 shows the organization of the method. Processes are supported by three ontologies. One reflects the knowledge of programming styles, and the second, about the styles of programming languages, and the third, about programming languages.

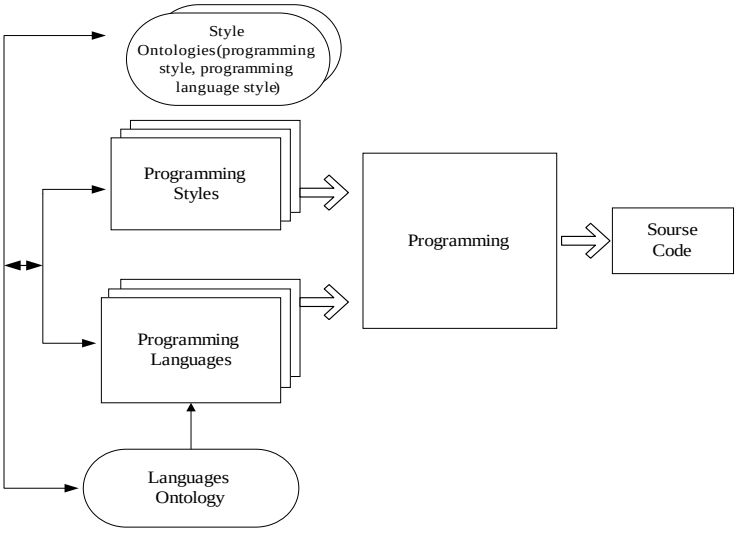

Fig. 1. Ontology-driven using of programming style

The use of ontologies requires some work, which consists in the analysis of the subject area and constructing the source structures (thesaurus, taxonomy dictionaries) (Figure 2).

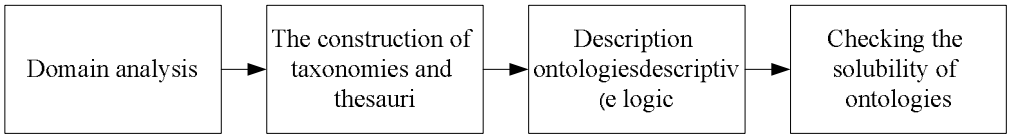

Fig. 2 The processes of preparation of ontologies

The article discusses the results of the first two processes.

Domain analysis is fulfilled with the help of the domain analysis techniques. As a result, a number of charts are created as the domain knowledge. Domain is called a programming stylistics. Knowledge about the domain is represented by three ontologies - programming style, programming language style, programming language (Fig.3).

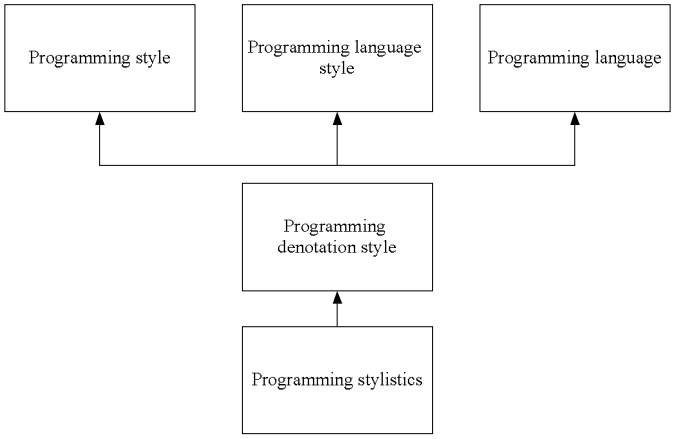

Fig.3. Programming stylistics

Knowledge is based on the definition of programming style. Programming style by the definition is the style that is used in human activity (domain), whose essence consists in programming (Fig. 4).

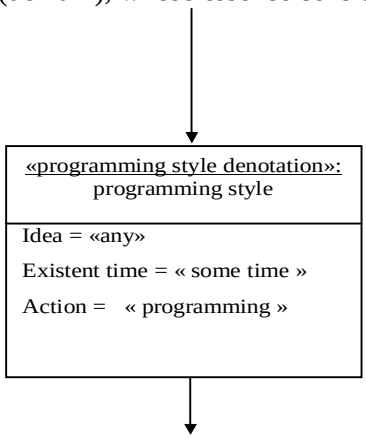

Fig.4. Class-programming style

The «programming» domain consists of three essences – subject (programmer), tool (programming language), and product (program) (Fig.5).

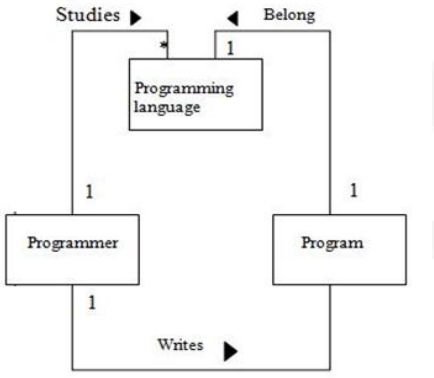

Fig.5. Programming domain

Programming style consists of the rules that apply to parts of the program text (Fig. 6)

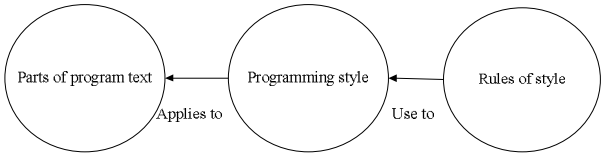

Fig. 6. Programming style with associated terms

The set of style rules consists of three types of rules - syntactic, semantic and pragmatic (Fig. 7).

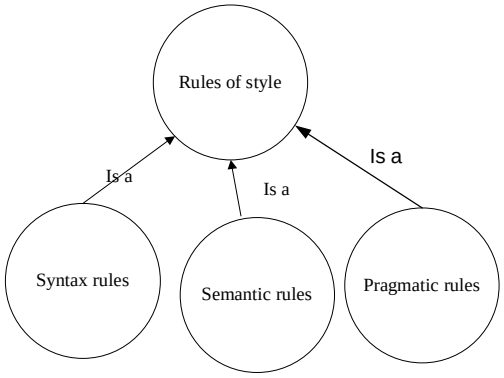

Fig.7. Rules of style

Description of the programming style is represented by the set of style rules. For example [6],

Syntax rule:
Synopsis: Do not use an underscore in identifiers
Language: C#
Level: 8
Category: Naming

Semantic rule:
Synopsis: Do not change a loop variable inside a for loop block
Language: C#
Level: 2
Category: Control flow
Description
Updating the variable loop within the loop body is generally considered to be confusing, even more so if the variable loop is modified in more than one location. This rule also applies to foreach loops.

Pragmatic rule:
Synopsis: Name an identifier according to its meaning and not its type
Language: C#
Level: 6
Category: Naming
Programming style is applied to parts of the program text. Part of the program text can be of two types – predefined by syntax parts, and parts, that can be defined (Fig. 8).

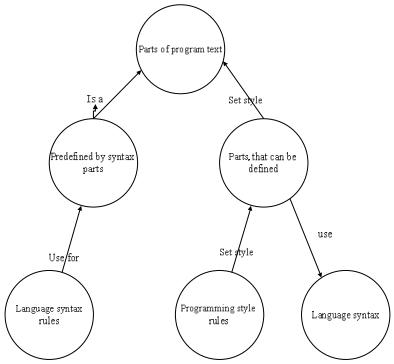

Fig.8 Parts of program text

Programming languages are represented with the help of the encapsulation levels [1]. Each level has its own type of the programming construction (Figure 9). There are lexems (lexical level), operators (operator level), subroutines (subroutines level), modules (module level), classes (class level).

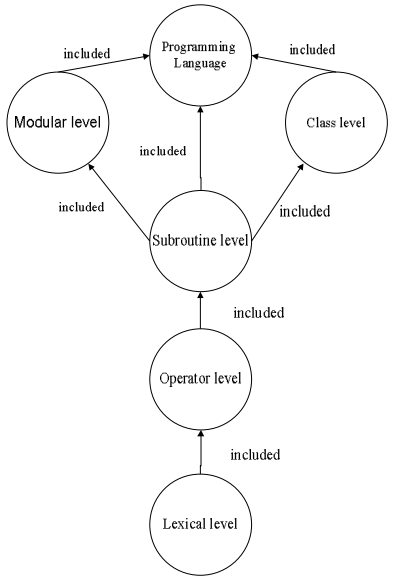

Fig. 9 Levels of encapsulation of programming language

Thus, using levels of encapsulation the style rules can be classified as the following (Figure 10).

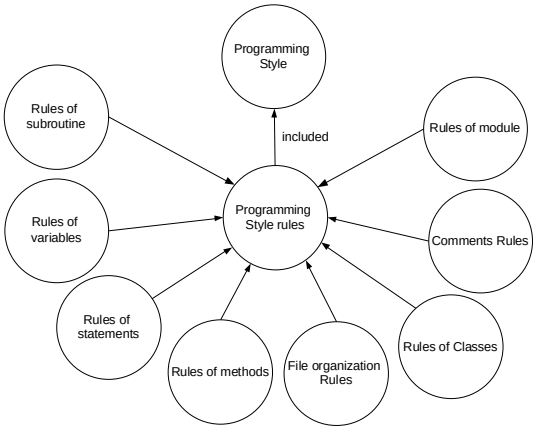

Fig. 10 Ontology of programming style's rules

The concrete programming styles are a created for concrete programming languages (Figure 11).

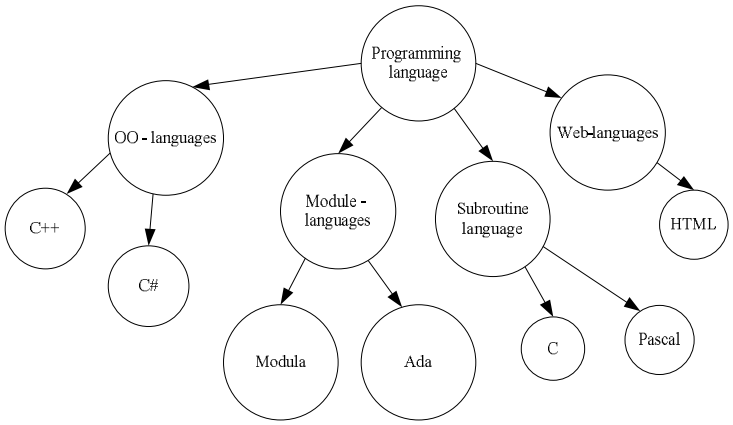

Fig. 11 Programming language styles

Fragment taxonomy rules for object-oriented language (C#) is shown in the Figure 12

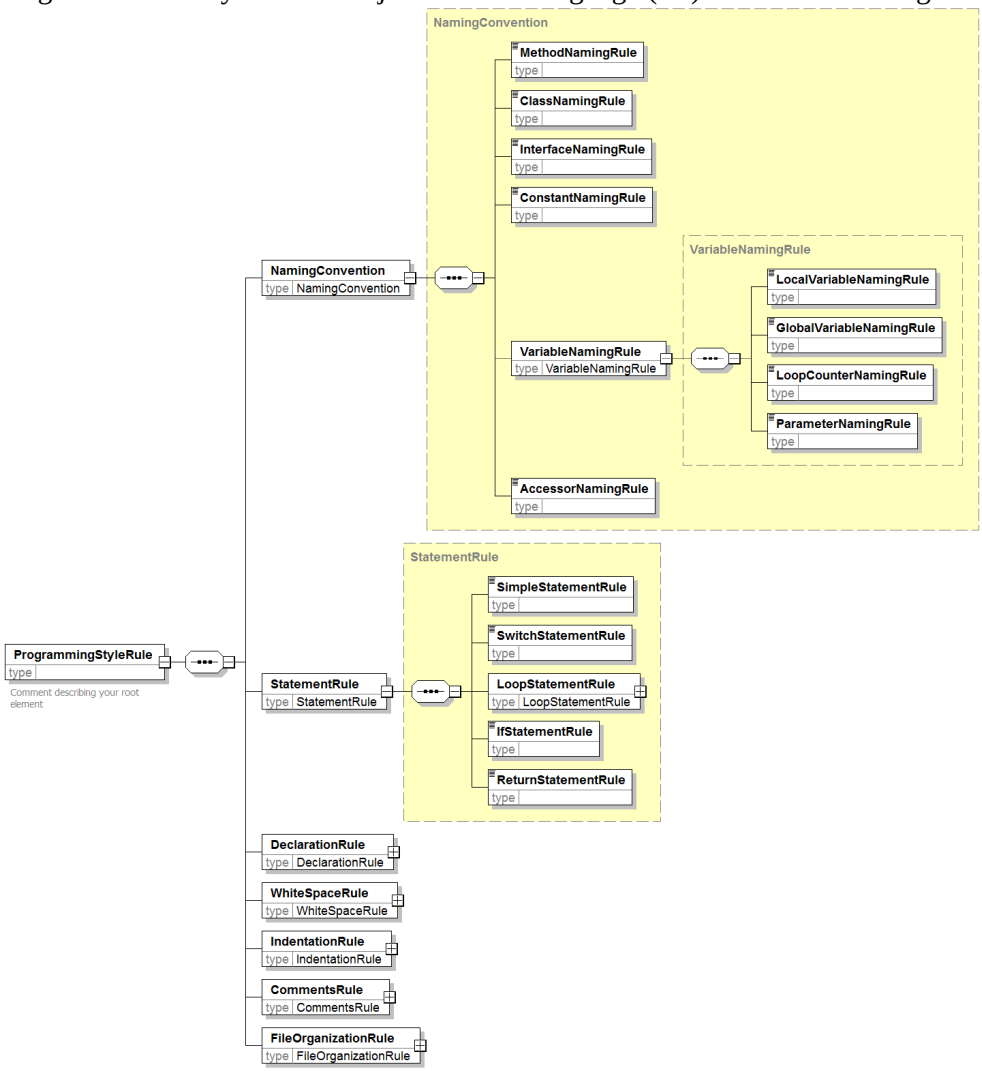

Fig. 12 Taxonomy of programming rules

Results and future researches. The results of the study programming stylistics domain are presented. The result is the taxonomy. A taxonomy of concepts will be used to construct the programming styles ontology. Ontology is a part of the tools that is constructed according to the own method. The future researches are creating programming style ontology and programmer assistant tool.

References

1. Sidorov M.O. Software engineering. [текст] /Sidorov M.O.// – 2007. – Kyiv. – NAU.- 135p.
2. Sidorov N.A. Software stylistics [текст]/Sidirov N.A.// Proc. of the National Aviation University – 2005. - №2. – c.98-103
3. Керниган Б., Элементы стиля программирования [текст] // Керниган Б., Плоджер Ф.// Радио и связь – 1984. –. 160с.
4. Крамар Ю.М. Средства для автоматизированного синтеза стилей программирования [текст]/Крамар Ю.М.// Весник НАУ 2002. –.- №2.- с.52-60
5. Sidorova N.M. Ontology of programming style [текст] /Sidorova N.M., Kramar Y.M.// – Proc. the sixth world longest "Aviation in the XXI-st Century.- 2014.- v.1.- P.1.13.28- 1.13.36"
6. Philips Healthcare – C# Coding Standart [текст].- Philips.- 2009.57p.
7. Knuth D.E. Literate are programming [текст] / Knuth D.E. //Computer Journal. - 1984. - Vol. 27, N 2. - P. 42-44.
8. Goldberg A. Programmer as Reader [текст] / Goldberg A.// IEEE Software -1987. Sept.-P. 62-70.
9. V. Railich Software cultures and evolution /V. Railich, N. Wilde, M. Buckellew // Computer. - 2001. - Sept. - P. 25-28.
10. Сидоров Н.А. Стилистика программного обеспечения [текст] / Сидоров Н.А. //Проблеми програмування. – 2006.– № 2, 3.– С.245-255.