Object-Oriented Design of the Specialized Software for Automation of the Metallographic Analysis

Vitaliy A. Yemelyanov, Nataliya Yu. Yemelyanova
Department of Business Informatics
Financial University under the Government of the Russian Federation
Moscow, Russia
v.yemelyanov@gmail.com,

Aleksey A. Nedelkin
Department of Informatics
Plekhanov Russian University of Economics
Moscow, Russia
aa@nedelk.in

Evgeny V. Shved
Department of Higher Mathematics
Plekhanov Russian University of Economics
Moscow, Russia
kafedra_vm@mail.ru

Egor O. Rubak
Department of integrated information security
Admiral Makarov State University of Maritime and Inland Shipping
St.Petersburg, Russian Federation
egr.work@yandex.ru

***Abstract*—The relevance to develop the specialized software for automation of the metallographic analysis has been substantiated. The object model of the specialized software for automation of the metallographic analysis has been designed. The proposed object model is basis to develop the specialized software for automation of the metallographic analysis. The model represents basic abstractions of the metallographic analysis, software use-cases and organization of the program modules. Bases on the proposed model the specialized software for automation of the metallographic analysis has been developed.**

Keywords—specialized software; object model, object-oriented approach; metallographic analysis; use-cases

I. INTRODUCTION

In order to improve the quality of products, enterprises are constantly increasing the volume of control operations and the number of inspection personnel. An important means of solving the problem of the quality control at the enterprise is to use objective physical control methods, such as metallographic analysis [1]. The constant increase in the requirements for the quality of ferrous metals of various groups and classes necessitates the development of models and means of automated product quality control.

As it is known [1-4], the metallographic analysis is based on the interpretation of images of metal microstructures. Reducing the time spent for this operation is possible with using the software that will provide the technologist with the capability to process automatically images of metal microstructures. Therefore, the task of developing specialized software for the metallographic analysis process, which will analyze the images of metal microstructures, becomes urgent

II. DEVELOPMENT OF AN OBJECT MODEL OF SOFTWARE FOR METALLOGRAPHIC ANALYSIS

At the present stage of the development of information systems and technologies, special attention is paid to the structure and organization of software.

When developing the software structure, an object-oriented approach was used [5]. The object-oriented approach is based on object decomposition, i.e. presentation of the developed software in the form of sets of objects, in the process of interaction of which the required functions are performed through the transmission of messages [6].

The specification of the software being developed combines the following models:

1. Usage model, i.e. a description of the functionality of the software from the user's point of view.

2. Conceptual model, i.e. a model that describes the main abstractions of the subject area, which provide the required software functionality and their interaction;

3. Implementation model, i.e. a model that defines the real organization of software modules and files.

To develop these models, a unified modeling language (UML) was used [6].

In the usage model, the projected system is represented as a set of entities or actors interacting with the system using the so-called use cases. Use case is applied to describe the services that the system provides to the actor. At the same time, the model does not display how the interaction of the actors with the system will be implemented. The usage model is shown in figure 1.

978-1-7281-0339-6/19/$31.00 ©2019 IEEE

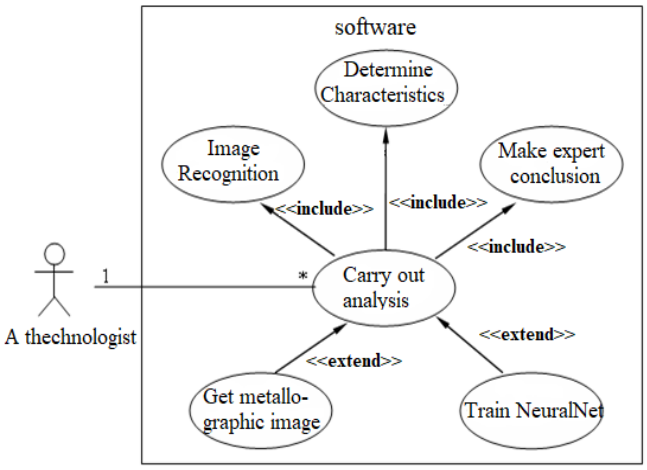

Fig. 1. Use case diagram of the software for metallographic analysis

In the model, six use cases and one actor (technologist) are distinguished, between which the inclusion and expansion relations are established. The values of the multiplicities indicated in the model reflect the general rules for processing images of metal microstructures. According to these rules, one technologist can process many images. In the model, the "Carry Out Analysis" use case is refined based on the introduction of three additional use cases. It results from a more detailed analysis of the process on its own, which makes it possible to distinguish as separate services such actions as image recognition, determining the characteristics of the recognized image and generating expert conclusions about the sample under study. These actions disclose the behavior of the original use case in the sense of its concretization, and therefore an inclusion relation will take place between them.

On the other hand, the analysis assumes the existence of an independent information object, i.e. a neural network. In our case, a neural network can be trained by a technologist on an existing set of images of metal microstructures.

Thus, based on the description of this model, it seems possible to develop a conceptual software model.

The conceptual software model is shown in figure 2.

The model is represented as a set of classes. This model is characterized by the relationship "whole - part". In the model, we can distinguish the class "Software" as a "whole", and all other classes are its "parts".

In the model such classes as "Image", "Expert conclusion", "Neural network", "Characteristics" exist. Between the classes "Image" and "Expert conclusion" there is an association showing that for the image of the microstructure of the metal there may be an expert conclusion reflecting information of a quantitative and qualitative nature. Moreover, for each specific image of the microstructure its own expert evaluation is designed for, which is reflected in the form of a one-to-one relationship. The classes "Expert system" and "expert conclusion" are connected by the "one for many" relationship, i.e. one expert system can generate multiple expert conclusions for multiple metal images.

The classes "Software", "Neural Network" and "Expert System" incorporate several operations that enable one to manipulate their parts. These operations are included because of their significance for maintaining data integrity (for example, changing the type of neural network in the "Software" class affects other classes, such as "Neural Network").

The software can analyze any number of images of metal microstructures, which is reflected by the corresponding relationship "1 .. *". The software may consist of one or more neural networks [7-8] and only one expert system.

The relationships between the Software class and the Image and Neural Network classes are different from each other, although both are aggregation relationships. The relationship between the class "Software" and the classes "Neural network", "Expert system" is a composite aggregation, because these classes are an integral part of the whole - the "Software" class, and the "Image" class is a separate part.

The "Characteristics" class includes quantitative and qualitative characteristics of the recognized image of the metal microstructure, which is reflected in the form of the operation of switching on the corresponding classes.

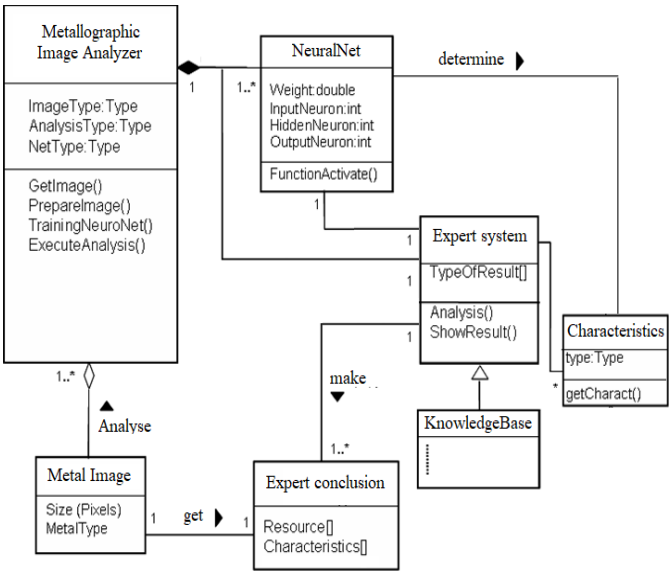

Fig. 2. Class diagram of the software for metallographic analysis

Previously created models reflected the conceptual aspects of developing an object model of the system and related to the logical level of representation. The main purpose of the logical representation is to analyze the structural and functional relationships between the elements of a system model. However, to create a specific physical system, it is required in some way to realize all the elements of a logical representation into specific material entities [9-12]. Another aspect of the model representation, namely the physical representation of the object model, is designed to describe such real entities. For the physical representation, an implementation model is developed as shown in figure 3.

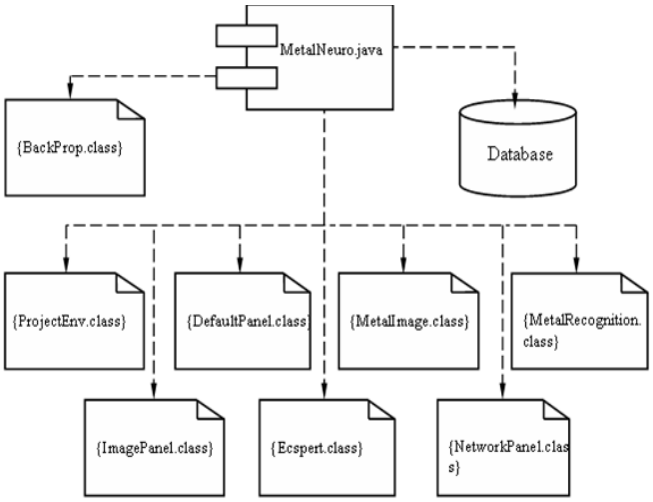

Fig. 3. Component diagram of the software for metallographic analysis

The model describes the features of the physical representation of the system, as well as the relationships between software components are established.

The MetalNeuro.java component implements a certain set of interfaces for interconnection with the database and modules, and also serves for the general designation of elements of the physical representation of the model.

The module of the BackProp.class implements the properties and methods of the neural network, as well as the error back propagation algorithm.

The module of the ProjectEnv.class contains information about the up-to-date state of the system.

Module of the class DefaultPanel.class describes the main panel of the graphical user interface (GraphicUserInterface)

The module of the MetalImage.class implements properties and methods for accessing the current image being processed.

The module of the MetalRecognition.class connects all modules, ensuring the functioning of information flows.

The ImagePanel.class class module contains class methods for visualizing recognition results.

The module of the Ecspert.class class contains the methods, properties and functions required for the expert system.

NetworkPanel.class class module contains a class that inherits class methods for visual operation within a neural network.

Based on the proposed object model, software was developed, which allows automated image processing for metal microstructures (figure 4).

Since the system includes data mining tools, the operating modes of the software are divided into “analysis” and “training”.

The functions of the software in the analysis mode are as follows:

- Acquisition and input of primary data on metal microstructures required for the implementation of the metallographic analysis.
- Recognition the metal microstructure images based on the primary data.
- Determining the metal characteristics.
- Keeping the experience gained in metallographic analysis in the database.

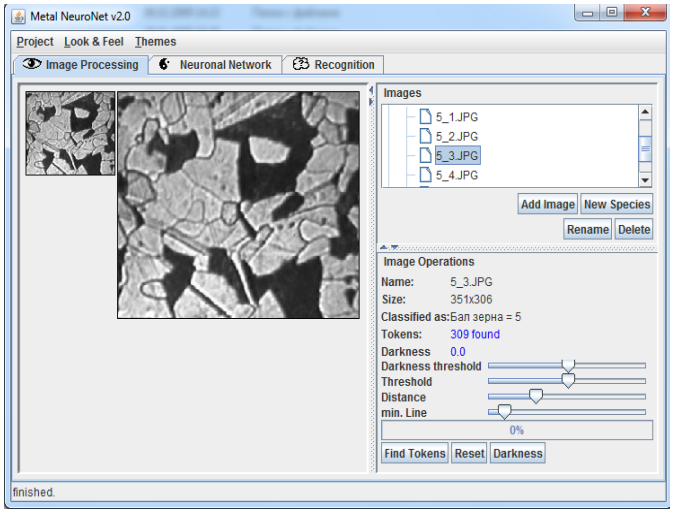

Fig. 4. Developed software for metallographic analysis

In training mode, the software supports:

- Creation of a neural network and setting its main parameters (number of layers, neurons, learning rule, etc.).
- Loading of data on the metallographic analysis for training the neural network.
- Training a neural network with the task of metal microstructure image recognition.
- Determining by graph the value of the optimal training epochs of a neural network.

III. Conclusion

Thus, the developed object model displays the main abstractions of the subject area, the options for using the software, its physical representation, as well as the information flows functioning in the software required for the metallographic analysis process.

Based on the proposed object model, software was developed, which allows automated image processing for metal microstructures. In its turn, it also enables decision supporting regarding the metal sample under study.

References

[1] N. Martyushev and V. Skeeba “The method of quantitative automatic metallographic analysis” in Journal of Physics: Conference Series. 803. 012094, 2017, 10.1088/1742-6596/803/1/012094.

[2] B. L. DeCost and E. A. Holm “A computer vision approach for automated analysis and classification of microstructural image data” in Computational Materials Science, vol. 110, 2015, pp. 126-133.

[3] T. M. Nunes, V. H. C. de Albuquerque, J. P. Papa, C. C. Silva, P. G. Normando, E. P. Moura, J. Manuel and R.S. Tavares “Automatic microstructural characterization and classification using artificial intelligence techniques on ultrasound signals”in Expert Systems with Applications, vol. 40, issue 8, 2013, pp. 3096-3105.

[4] S.R. Niezgoda, A.K. Kanjarla and S.R. Kalidindi “Novel microstructure quantification framework for databasing, visualization, and analysis of microstructure data” in Integr. Mater. Manuf. Innov, vol. 2, issue 1, pp. 1-27.

[5] P. Krutchen “The Rational Unified Process – An Introduction (3rd edition)” Addison-Wesley, 2003.

[6] G. Booch; J. Rumbaugh and I. Jacobson “The Unified Modeling Language User Guide”, Addison-Wesley, 2005.

[7] S. Kenji “Artificial Neural Networks: Architectures and Applications”, InTech, 2013.

[8] P. Danny, K. Derek and A. Swarnalatha “Object-Oriented Programming and Java” Springer, 2008.

[9] S. Chernyi, "Methods for ensuring fault tolerance of equipment based on reliability theory", 2018 IEEE Conference of Russian Young Researchers in Electrical and Electronic Engineering (EIConRus), 2018. DOI: 10.1109/EIConRus.2018.8316860

[10] S. Chernyi, S. Kolesnichenko, B. Novak, A. Degtyarev and P. Erofeev, "Reliability and Fault-tolerance Control of Automation Systems on Maritime Vessels", 2018 IEEE International Conference "Quality Management, Transport and Information Security, Information Technologies" (IT&QM&IS), 2018. DOI: 10.1109/ITMQIS.2018.8524934

[11] V. Veselkov, N. Vikhrov, A. Nyrkov, S. Chernyi and I. Titov, "Development of methods to identify risks to build up the automated diagnosis systems", 2017 IEEE Conference of Russian Young Researchers in Electrical and Electronic Engineering (EIConRus), 2017. DOI: 10.1109/EIConRus.2017.7910625

[12] N. Logunova, L. Aleksahina and S. Chernyi, "Region-specific determiners of marine transport system development as basis for strategic focus formation", 2017 International Conference "Quality Management,Transport and Information Security, Information Technologies" (IT&QM&IS), 2017. DOI: 10.1109/ITMQIS.2017.8085781