JOINT UKRAINE- POLAND R&D PROJECT
FOR THE PERIOD OF 2018 – 2019

Lodz University of Technology
Institute of Electrical Power Engineering

Pryazovskyi State Technical University
Electrical Engineering Department

International Ukraine-Poland Seminar

Power quality in distribution networks with distributed generation

Kiev, July 4-5, 2019

DOI: 10.32073/iepl.2019.22

OBJECT-ORIENTED APPROACH TO INDUCTION MOTOR DESIGN

Vladyslav Pliuhin[1], Vasylyi Gariazha[1], Volodymyr Korobka[1], Tatyana Baranenko[2]

[1] O.M. Beketov National University of Urban Economy in Kharkiv, Ukraine
[2] Pryazovskyi State Technical University, Ukraine

Abstract: It was proposed to introduce the principles of object-oriented design into the structural and systemic organization of electromechanical energy converters, which makes it possible to build easily modifiable projects that lack such traditional design disadvantages as the presence of hard links between individual project phases and make it difficult to coordinate input and output parameters of calculation blocks.

1. INTRODUCTION

Object-oriented design is an approach to solving problems, consisting in the consideration of the subject area and the logical solution of the problem from the point of view of objects. The fundamental element is the object that combines the data structure with the calculated blocks. An electromechanical energy converter (EEC), as an object of design, is characterized by a set of design data defining its assembly: geometrical dimensions, winding data, technical and economic parameters. All these data should satisfy many internal mutual relations in the form of analytical ratios, determined by physical laws, peculiarities of structural design and technological factors. The idea is to accompany the inheritance of the constructive characteristics of the derived classes of EEC by simultaneous creation their mathematical representation.

The proposed design method performs consistent steps, at each of which they solve strictly defined tasks. Each stage in its internal implementation is independent of the other stages, representing a design module with a set of input and output parameters. Input parameters come from previous ones from the current calculation modules, and output parameters are passed to the next one from the current calculation module.

2. ASSEMBLY OF COMPOUND OBJECT MODEL

When constructing a composite model of objects, the operations of generalization and inheritance are used [1, 2]. In the generalization process, the main class is characterized by general attributes (parameters), operations (calculated blocks) and associations (logical connections in the model), which it can delegate to its descendant classes.

In the inheritance hierarchy, an object of a subclass is simultaneously an object of all its ancestor classes. Due to inheritance, the object has the values of all attributes of all ancestor classes. In this case, the object can also invoke any operation specified by any of its ancestors.

In a widely branched hierarchical tree of a large project, in which subclasses are subordinated to the operations of generalization and inheritance, it is necessary to group the individual subclasses into a single data structure. Often it is important not only for the holistic perception of the project, but also for the realization of the possibility of its subsequent modification with minimal expenses for the synthesis of a new project. As an example, the grouping of subclasses “stator”, “rotor”, “slot” into the class “electric motor” can be given. For the formation of such a composite model of objects, the operations of aggregation and composition [3] are used (Fig. 1).

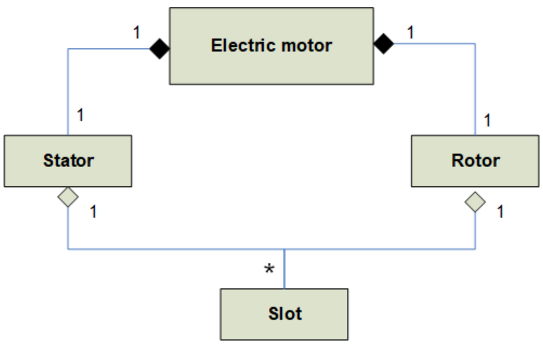

Figure 1. An example of an induction motor class composite model with aggregation and composition operations

Aggregation is characterized by an additional semantic meaning [4–6]. If two objects are related by a part-to-whole relationship, they should be modeled using aggregation. Aggregation is represented by a small rhombus, which is placed near the pole, which is an aggregate.

Composition is a special case of aggregation, characterized by two additional limitations:

- component can belong to no more than one unit;
- the component attributed to the unit automatically receives programmatic lifetime, which coincides with the lifetime of the unit.

Thus, the composition implies that the parts belong to the whole. To indicate the composition, a filled rhombus is used, which is placed with the class-unit.

Class models are not only useful for defining data structures. Tracing class models makes it possible to express some kinds of behavior. Various models of classes allow us to describe a complex system, which was shown in the example.

In the well-known methods of EEC designing [7, 8], the order of calculations in design project is as follows:

- selection of main sizes;
- calculation of stator winding;
- calculation of rotor winding;
- calculation of the magnetic circuit;

– determination of parameters of stator and rotor windings;
– calculation of losses, operating and start-up characteristics;
– ventilation, thermal calculations;
– mechanical shaft calculation.

The listed design stages are rigidly interconnected by a hierarchy. Each new type or modification of EEC requires the processing of calculated formulas, the introduction of new variables, or the exclusion of already existing ones. Data and calculation blocks that are at the lower level of the hierarchy cannot be obtained if the values of the parameters that are located higher in the hierarchy were not calculated.

Unlike traditional (cascading) design stages, the formation of an object-oriented project is performed differently [9, 10]. First, the creation of the project begins with the initialization of the base class. For example, in induction motor (IM) with a squirrel cage rotor (SCR), the base abstract class “Stator” contains local variables and calculation blocks, which are characteristic of all machines of an asynchronous type (Fig. 2).

The term "abstract" means that a given class cannot spawn objects, i.e. is an idealized data structure that summarizes the characteristics of various types of electric machines.

The main local variables of the stator class are:
– number of slots;
– external / internal diameters;
– length of the stator core;
– steel grade mark;
– wire section;
– current, power, voltage, number of pole pairs, network frequency;
– magnetic flux, induction, active and inductive resistance of the winding.

The main calculation blocks of the stator class:
– calculation of the main geometric dimensions;
– calculation of parameters;
– calculation of magnetic flux, current density.

A feature of the “Stator” class is that the variables and calculation blocks are invariant to the type of IM and in each specific implementation will be replaced by actual numerical values.

The refined “Stator with Slots” class inherits the variables and calculation blocks of the base abstract class “Stator” and introduces its own variables and calculation blocks, which are inherent only for this kind of induction motor. This class already belongs to a specific variety of EEC and, therefore, can generate objects.

The example shows a stator with trapezoid slots and round wire. This class was obtained as a result of inheritance from the “Stator”, “Round wire” and “Trapezoidal” classes. Local variables and calculated blocks of the resulting refined class will no longer be common to different machines, but will be tied to the realized subspecies:

“Slot” class local variables:
– wire diameter;
– the height of the slot;
– the width of the top / bottom of the slot;
– the fill factor of the slot;
– coefficients of slot, frontal, differential scattering;

“Slot” class calculating blocks:
– calculation of the geometry of the slot;
– calculation of coefficients of slot, frontal, differential scattering;

inherited computational units of the class "Slot":

– calculation of active and inductive resistances.

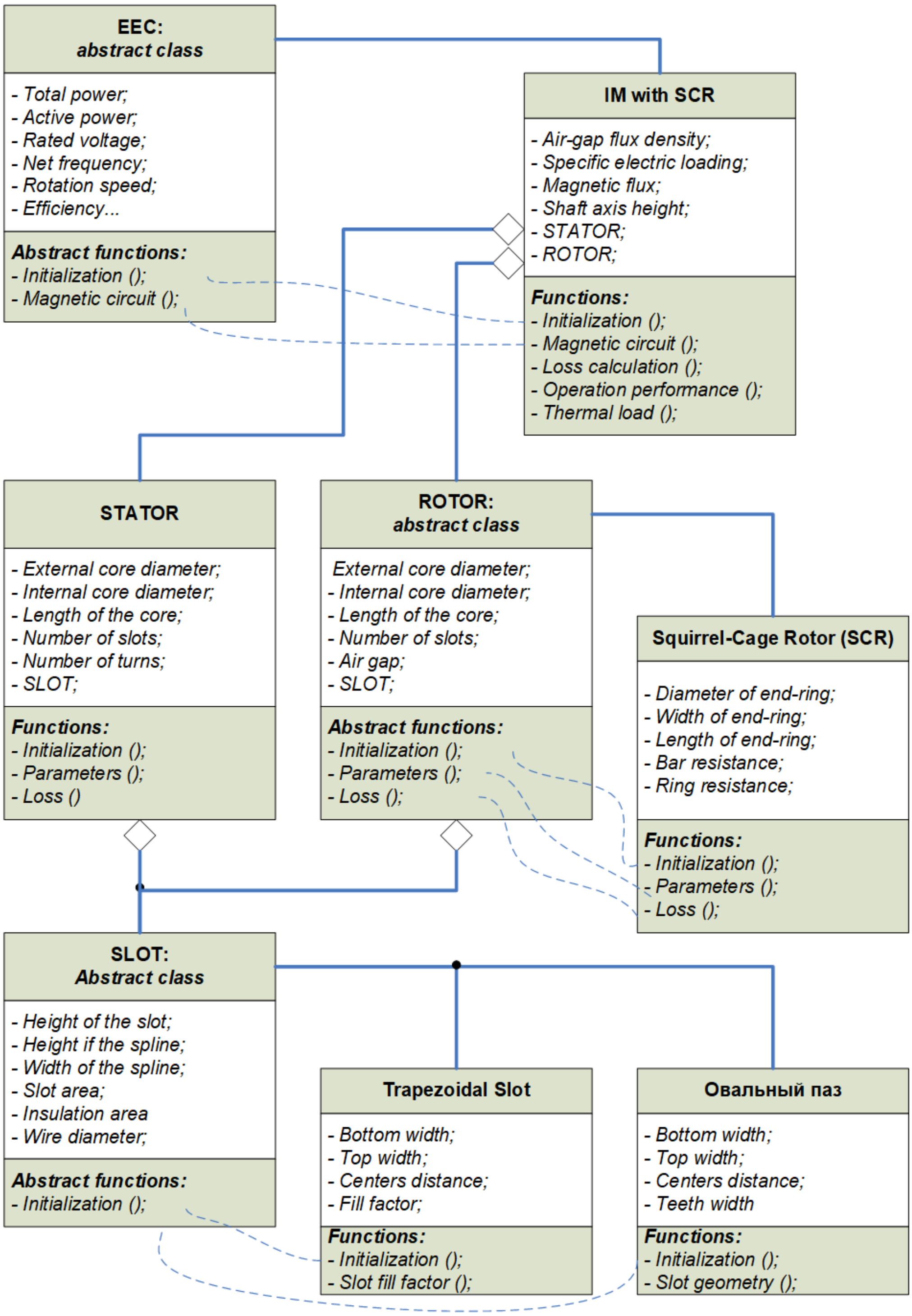

Figure 2. Class diagram of object-oriented design project for induction motor

As can be seen, the refined "Stator with Slots" class inherits from the base class methods for calculating parameters using its own calculation blocks to obtain the auxiliary coefficients included in the formulas for active and inductive resistances. When changing the slot type, the inheritance logic remains unchanged, but in the final formulas, the blocks for obtaining parameters are dynamically replaced, which is realized due to the principles of inheritance and polymorphism.

The base abstract class “Rotor”, similarly to the abstract class “Stator”, also contains local variables and computational blocks characteristic of different types of rotors.

The refined class “Squirrel Cage Rotor” is a descendant of several classes: “Rotor”, “Rotor’s Slot” and is also obtained as a result of inheritance.

The final class “Induction motor with a squirrel-cage rotor” located on the top level of the hierarchical tree. It inherits all the properties of the lower parent classes and introduces its own data and calculation blocks. In particular, it is the calculation of operating and start-up characteristics, ventilation, thermal and mechanical calculations, simulation of transient characteristic only for an induction motor with a squirrel-cage rotor. It should be noted that in each specific implementation of different types of EEC, the listed calculations will be adapted to the accepted chain in the inheritance tree. In addition to the electromagnetic calculation incorporated in the class tree inheritance, the project is complemented by independent classes: the ventilation system, the thermal system, the dynamics, the mechanics.

3. IMPLEMENTATION OF OBJECT-ORIENTED PROJECT

At the stage of the technical task statement the initial data to the project is forming. As the initial parameters for IM, are accepted: rated power, rated voltage, winding circuit diagram, number of poles, operating temperature. These data are used to initialize stator and rotor classes of generalized electric machine (EM) and IM.

1. According to the requirements of the standards (DSTU, GOST, ISO) for EM the basic geometric dimensions of the IM are chooses, performed by the degree of protection and cooling, parametric constraints governing the multiplicity of starting current, over-loading capacity, multiplicity of the starting torque, temperature of the IM nodes, etc.

2. The algorithm of an object-oriented design includes:

– the stage of creating a design object, which distinguishes the base class, searches for keywords, compiles a class diagram, group variables, and calculation blocks in the corresponding classes, execute the filling of the calculation blocks by formulas [11].
– basic function of calculating geometric sizes, containing data common to stator and rotor;
– a function of calculating the geometric sizes of the stator, which is the heir of the basic function of calculating the geometry;
– control of the correct choice of geometric sizes and stator parameters;
– a function of calculating the geometric dimensions of the rotor, which is the heir of the basic function of calculating the geometry;
– control of the correct choice of geometric dimensions and rotor parameters;
– function of calculation of working, start-up characteristics, modeling of IM in dynamic modes, field calculation;
– function of calculation of thermal and mechanical parameters.

3. Forming the classes of the project, consisting of the base class of the generalized EM and the descendants, which extend the functional capabilities of the base class and differ in the features of the projected EM [3, 9–10]. Classes make up the UML language and represent graphically in the form of diagrams that establish links between the base class and the descendant classes.

In this case, the adoption of initial data, the accounting of technical requirements are delegated from the base class to the descendants. The logical connections between the calculation blocks are formed. When creating an object, the IM with SCR the sequence of calling calculation blocks in accordance with the design stages.

To get a class representation of EM, the keywords are assigned:

– electric motor, IM, IM with SQR, stator, rotor, SQR, slot, rectangular slot, trapezoid slot, slot, winding, conductor, steel grade.

The base class includes calculation blocks, which with the use of polymorphism call the calculation blocks of the descendants.

At the current stage, distinguish the parameters of EM and group them in the corresponding classes, fill out the calculation blocks formulas to determine the parameters determined by the appointment of the calculation block.

For an IM with a SQR stator parameters and its windings are included in the Stator class; the parameters of the rotor and its windings are grouped in the abstract class Rotor and the class SQR. The slot parameters are grouped in the abstract slot class, rectangular slot, trapezium slot, oval slot.

4. At the stage of entering the initial data, initial initialization in a computer program, compiled in the object-oriented language in accordance with the UML class diagram is performed. Such common for generalized EM parameters as nominal power, frequency, and voltage are transmitted by the polymorphic function in the main EM class. In the calculation block of the designed EM class, derived from the base class, the parameters that are available at the stage of inputting the initial data for the EM are determined.

5. At the stage of calculating the geometric sizes (Fig. 3), the choice of the number of stator slots, which is included in the separate object in the IM class, inherited from the base class, its internal and external diameters for given initial data. A choice of stator slot type is performing.

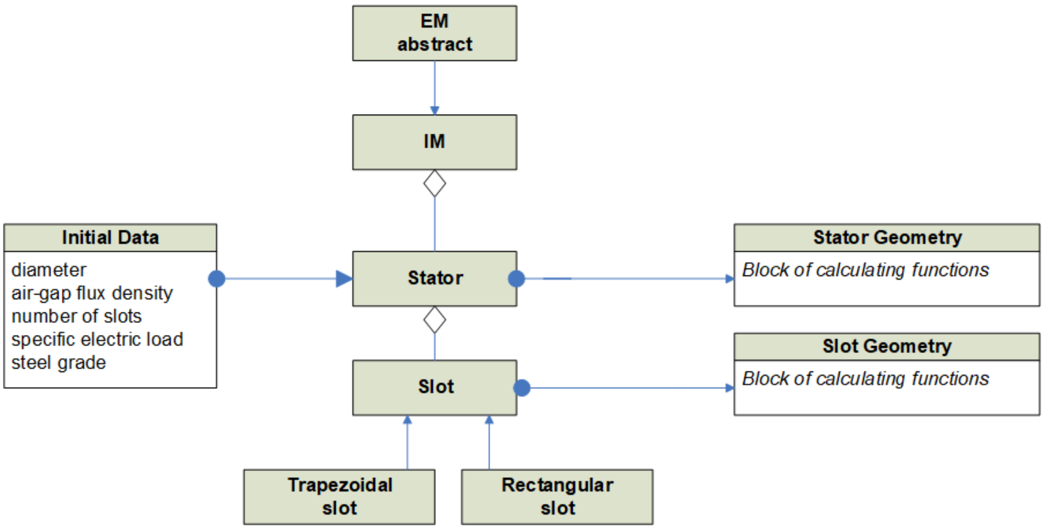

Figure 3. Class diagram of the stator of induction motor

After that, the stator geometry is calculating taking into account the type of slot. In the same way, the calculation of the rotor, the class of which, as well as the stator class, is a separate object in the IM class of the.

The rotor class includes the same slots classes as the stator, however, due to polymorphism, the parameters of the grooves of the stator and rotor classes do not conflict with each other, since they are dynamically redirected to different objects. For the rotor, select the number of grooves, the outer and inner diameters for given initial data. Make a choice of type of rotor slots. Perform calculation of the geometry of the rotor, considering the type of groove.

6. Performing the calculation of the magnetic field. Calculation of the magnetic field of the IM includes the parts of the stator's circuit and the circuit of the rotor. The calculation is made from an abstract unit of calculation of the main IM class, which defines the sequence of the call of the calculation blocks of the stator's circuit, and then the circuit of the rotor. Then the total ampere-turns and the magnetization current are determined.

7. Performing calculation of active and inductive resistances (resistance and reactance). Calculation of active and inductive resistance of stator and rotor windings performed by a single polymorphic command, which called from the IM class calculation blocks for determination of stator parameters, and then the rotor. On Fig. 4 SQR class as the only descendant of the parent class Rotor is considered.

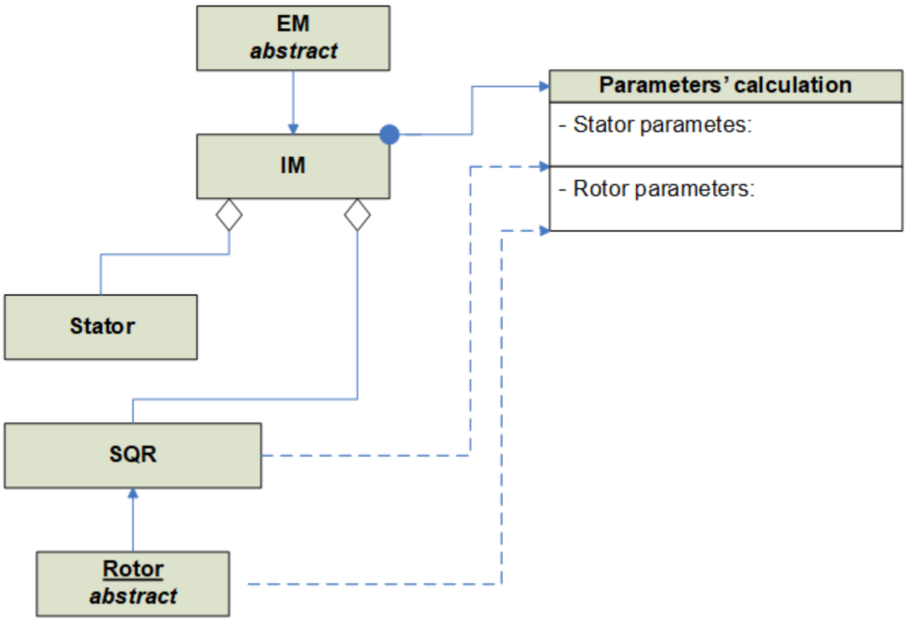

Figure 4. Class diagram of the rotor of induction motor

In this class diagram the SQR of the abstract calculating blocks of the class Rotor acquires a specific implementation but retains the same name for the program call.

8. Performing the losses calculation. Calculation of the IM losses includes calculation of losses in stator and in the rotor, realizing by the calling of the corresponding calculation blocks. Polymorphic block, being global and having full access to the obtained results, completes the calculation of losses by determining the efficiency of the IM.

9. Performing the calculation of non-loading, loading and starting characteristics. The calculation of the IM characteristics, taking into account the parameters of the stator and rotor classes, are performed in a loop when changing the slippery in a given range with the definition of working and starting characteristics.

10. Performing heat calculation. During the thermal calculation, formed on the basis of the structure of IM class diagram, the excess of the temperature of the stator winding under the open air temperature is determined using equivalent thermal schemes, the nodes of which inherit known at this stage parameters of the IM classes.

11. Performing the calculation of mechanical strength and reliability. At this stage, the mechanical calculation of the shaft and bearings of the IM with the definition of reliability indicators, including the data of stator winding temperature and mechanical calculation.

4. CONCLUSIONS

In the proposed method of induction machines design an object-oriented approach in the organization of calculation procedures is implemented. The flexibility of the presented object-oriented design model of an induction motor allow to make the follows obvious conclusions:

- maximally approximates the formal representation of the project to object-oriented programming languages;
- increasing the efficiency of design, reduces the time and material costs; on it
- simplifies modification of the project;
- makes it possible to reuse the existing developments in new projects.

The modifications made to the project will not require a complete reworking of the calculated sequences, even if the designations of variables and data types are changed, as if it were in a procedural implementation with subroutine blocks.

The formal representation of the computational algorithms within the classes allows to perform various hereditary combinations, and the final object is already completely defined and calculated.

5. LITERATURE

1. Weisfeld M.: The Object-Oriented Thought Process. Addison-Wesley Professional, London, 2013.
2. Snyder A.: Encapsulation and Inheritance in Object-Oriented Programming Languages / Alan Snyder // SIG- PLAN Notices, November 1986. – Warsaw, 1986. – pp. 38 – 45.
3. Zablodskii N.N.: Using object-oriented design principles in electric machines development / N.N. Zablodskii, V.E. Pliugin, A.N. Petrenko // Electrical Engineering & Electromechanics, Kharkiv. – 2016. – no. 1. – pp. 17–20.
4. Booch G.: Modeling Language user guide / Grady Booch, James Rumbaugh, Ivar Jacobson. – New Jersey: Addison-Wesley, 2005. – 432 p.
5. Flower M.: UML Distilled. A Brief Guide To The Standard Object Modeling Language / M. Flower. – New York: Pearson Education, 2007. – 208 p.
6. Scott W.: Ambler. The Elements of UML(TM) 2.0 Style. Cambridge University Press, New York, 2005.
7. Nagarajan V.S., Rajini V.: Electrical machine design. Pearson, Uttar Pradesh, 2018.
8. Sen S.K.: Principles of electrical machine design with computer programs. IBH Publishing Company Pvt. Limited, Oxford, 2006.
9. Pliugin V.: Using of object-oriented design principles in mathematic modeling of electric machines / V. Pliugin, V. Milykh, A. Polivianchuk N. Zablodskij // TEKA, Lu-blin-Rzeszow. – Vol. 15, No. 2. – 2015. – P. 25 – 3.
10. Pliugin V.: Analysis of the Electromagnetic Field of Electric Machines Based on Object-oriented Design Principles / V. Pliugin, L. Shilkova, J. Letl, K. Buhr, R. Fajtl // PIERS 2015, Prague. – 2015. – P. 2522 - 2527.
11. Gamma E., Helm R., Johnson R., Vlissides J.: Design Patterns: Elements of Reusable Object-Oriented Software. Addison-Wesley, New York, 2000.

Professor, Dr.Sc. Vladyslav Pliuhin
O.M. Beketov National University of Urban Economy in Kharkiv
17, Marshala Bazhanova street, Kharkiv, 61002 Ukraine
e-mail: vladyslav.pliuhin@kname.edu.ua

Assistant Professor, Vasylyi Gariazha
O.M. Beketov National University of Urban Economy in Kharkiv
17, Marshala Bazhanova street, Kharkiv, 61002 Ukraine
e-mail: garany49@ukr.net

Assistant Professor, Volodymyr Korobka
O.M. Beketov National University of Urban Economy in Kharkiv
17, Marshala Bazhanova street, Kharkiv, 61002 Ukraine
e-mail: korobkat62@ukr.net

Associate Prof., Ph.D Tatyana Baranenko
Pryazovskyi State Technical University
str. Universytets'ka, 7, Mariupol, 87555, Donets'ka oblast, Ukraine
e-mail: tetianabaranenko@ukr.net