THE MINISTRY OF EDUCATION AND SCIENCE OF UKRAINE

NATIONAL TECHNICAL UNIVERSITY
«KHARKIV POLYTECHNICAL INSTITUTE»

I. P. Gamayun, O. Yu. Cherednichenko

MODELING OF SYSTEMS

Handbook for students of directions
6.050103 “Software engineering”, 6.050101 “Computer science”

Recommended by The Ministry of education and science of Ukraine
as a handbook for students of higher education institutions

Kharkiv
«Fakt»
2015

UDC 004.94
BBC 32.817в6
G 18

Recommended by The Ministry of education and science of Ukraine
as a handbook for students of higher education institutions
(letter № 1/11–16122 from 13.10.2014)

Reviewers:

S. G. Udovychenko, Doctor of Engineering Science, professor, Kharkiv national university of radioelectronics;

O. E. Fedorovych, Doctor of Engineering Science, professor, National aerospace university "Kharkiv Aviation Institute".

У посібнику викладені фундаментальні засади моделювання складних систем. Наведена класифікація математичних моделей та методи їхньої побудови. Розглянуто аналітичне моделювання основних видів процесів, що мають місце у складних системах. Описані принципи імітаційного та статистичного моделювання, а також моделювання бізнес-процесів.

Призначено для студентів вищих навчальних закладів, що навчаються за спеціальностями "Програмна інженерія" та "Комп'ютерні науки", викладачів та спеціалістів у галузі комп'ютерного моделювання.

Gamayun I. P., Cherednichenko O. Yu.

G 18 Modeling od systems: handbook for students of directions 6.050103 "Software engineering", 6.050101 "Computer Science" / I. P. Gamayun, O. Yu. Cherednichenko. – Kharkiv : Fakt, 2015. – 224 p.

ISBN 978-966-637-800-5

The handbook contains the fundamentals of modeling of complex systems. The classification of mathematical models is represented and the methods of their construction are given. The analytical modeling of the basic types of processes in the complex systems is considered. The principles of simulation, statistical and business processes modeling are described.

The handbook is oriented on students of higher education establishments that obtain a degree in directions of "Software engineering" and "Computer science" as well as on lecturers and specialists in the domain of computer modeling.

UDC 004.94
BBC 32.817в6

ISBN 978-966-637-800-5

TABLE OF CONTENTS

INTRODUCTION

The extremely fast development and sophistication of technologies, the growth of scales and costs for creation of new systems, the spreading of computerization caused the necessity of introduction of the notion of complex system.

A *complex system* is a set of interconnected and cooperating elements and subsystems of different physical nature, which form the indivisible unit, provide the fulfillment of the given complex function and can be described with the help of complex enough mathematical models.

The initial design stages are the most important in creation of the complex system. The overview of the researched system is made, the main requirements to tactical and technical characteristics are formulated, the generalized model of system structure is created and the laws of its functioning are described at initial stages. The errors on these design steps may lead to unjustified expenditures of intelligence, time and money or to irrational solutions and finally to unrealized projects.

The basic features of complex systems include the following:

- multidimensionality connected with existence of a big number of elements and relations between them;
- hierarchical multilevel character of the system and management structure;
- big number of structural and component solutions;
- multidimensionality of representation;
- distribution in space and time;
- multi-purpose functioning;
- probabilistic functioning and system behavior caused by complex interactions with changing external environment.

One of the most important instruments of system analysis is modeling. It provides the description of relations hierarchy, mechanisms and structures of complex multicomponent systems. It allows to:

- visualize the system in its current or expected state for research;
- define the system structure and behavior;

- obtain the patterns that may help to create and model the complex systems.

Modeling method of analysis is applied for formalized description of complex systems that contain presentation of elementary system objects, their features and relations. Nowadays together with the widespread traditional methods of analytical modeling attention is paid to methods of simulation modeling which allow to obtain the results of high quality during the analysis of complex system dynamics. They allow to reflect processes that take place in complex systems in the flexible and obvious way.

Today when the software industry provides various tools for modeling, every highly qualified engineer and manager must be able to model complex systems with the help of modern technologies realized in the form of graphical integrated environments and packages of visual modeling.

The advantages of simulation modeling include:

- dynamical character of system expression;
- possibility of taking into account random factors and complex dependencies;
- simplicity of introduction of any modifications in the model;
- possibility of exploring the system on the set of model realizations of its functioning, i.e. conducting statistical experiments;
- almost infinite possibilities of model detalization depending on the researcher's knowledge.

Depending on the character of the processes the simulation models (SM) are divided into three large groups: SM of continuous dynamic systems, SM of discrete systems, combined or general SM. Additionally there is another classification of SM according to the principle of dynamic arrangement, which is based on the possibility to represent an object as a collection of actions or as a sequence of actions, or as a set of processes. At that the most obvious formalization of system-related approach to research of complex system is done on the basis of the notion of "an object as a basic element of a system". Therefore most of the modern methods and information technologies of modeling are built according to object-oriented approach.

Nowadays, when the computer industry introduces various modeling tools, any well-qualified engineer and manager must have skills of modeling of complex systems by means of modern technologies implemented in the form of integrated environments and packages of visual modeling.

This handbook was designed for study of the course "Modeling of systems". In this handbook we consider the fundamentals of modeling. The focus was made to mathematical models of dynamical systems. The content reflects the main points of the course “Modeling of systems” and can help you to acquire basic theoretical knowledge and practical skills of this course.

This handbook is organized in the following way. The first chapter is devoted to general aspects of modeling. The second chapter considers main mathematical schemes for dynamical system modeling. The third chapter describes principles and methods of statistical modeling. The simulation modeling is the subject of the fourth chapter. And the last fifth chapter is devoted to business processes modeling. Each chapter contains examples, tasks for case-studying and control questions.

1. THE FUNDAMENTALS AND DEFINITIONS OF THE MATHEMATICAL MODELING THEORY

1.1. The concept of complex system and its model

The concept of "system" is widely used in science and practice to designate the objects of study by man. In scientific and technical literature, Internet, mass media the objects of the most various nature are called as corresponding kinds of systems - technical, economic, political, social, operational etc.

The width of usage of the concept “system” has also evoked a range of definitions of this concept in which two groups [1] are usually allocated.

The first group of definitions reflects the principles of studying developed in natural sciences for the surrounding world. Researchers divide the system under study into elements, study the elements properties, interactions (connection) of elements taking into account their properties and, thus, receive the idea about the system as a whole as a set of interconnected elements.

The *first group* is represented by the definition by Hall and Feige [1]:

«The system is a set of objects together with the relations between the objects and between their attributes (properties)».

Quite exact formalization of system S, defined in this way, was offered by Messarovich [2] through introduction of family of sets $\tilde{X} = \{X_i \mid i \in \overline{1,n}\}$, where X_i – the totality of manifestations of the attribute $i \in \overline{1,n}$. Then system S is such a relation over $\tilde{X}$ that $S \subset X_1 \times X_2 \times ... \times X_n$. In other words, system S represents the totality of manifestations of a real-life object which is called “system”.

The drawback of the first group definition is that it fails to take into account the system property as a complete object as well as the properties which are not reduced to the properties of the system elements.

In the definitions of the *second* group the mentioned drawback is eliminated. They realize the duality principle according to which the system on the one hand is a complete object and is included into a higher level system as the element and, on the other hand, it is split into the component elements making it.

The definitions of the second group «redefine» the system in relation to the definitions of the first group and, thus, express the following features of complex systems more fully:

- a big number of the elements forming a variable hierarchical structure, and sometimes stochastic character;
- actions (functioning) of the elements have, as a rule, uncertain and non-stationary character;
- the presence of various purposes in the elements which often have inconsistent character;
- the absence of mathematical description without which it is, first of all, impossible to solve management problems.

To be guided in a huge variety of objects which are defined as a system, classifications of systems have been developed. There are many classifications, each of which is based on certain classification attributes - the number of elements, the character and type of relations, the ways of interaction with the environment etc. Comprehensive enough classification is given in the following works [1, 3, 4]. According to this classification systems are divided into natural ones, which are created by nature, and artificial ones, which are created by man. Further natural systems are divided into live and lifeless systems, and artificial ones - into formal and informal systems. The importance of this classification is that (in it) the formal system or, otherwise, the system model is accurately singled out in it.

The emergence of the formal system is the result of the researcher's interaction with the system as a part of environment according to the objectives. The interaction scheme is shown in figure 1.1 [1].

Researchers are the people who study systems, create their descriptions, explain the interaction mechanisms of their parts and predict the system behaviour in time on the basis of the given objectives [1].

The goals for the researcher are formulated by decision making person (DMP) who solves problems connected with management, functioning and system development. Thus development is understood as goal setting and actions planning for their achieving, while functioning is understood as performance of actions directed onto goals achievement.

The environment or supersystem (metasystem) for system *S* is formed by the set of the objects interacting with *S*.

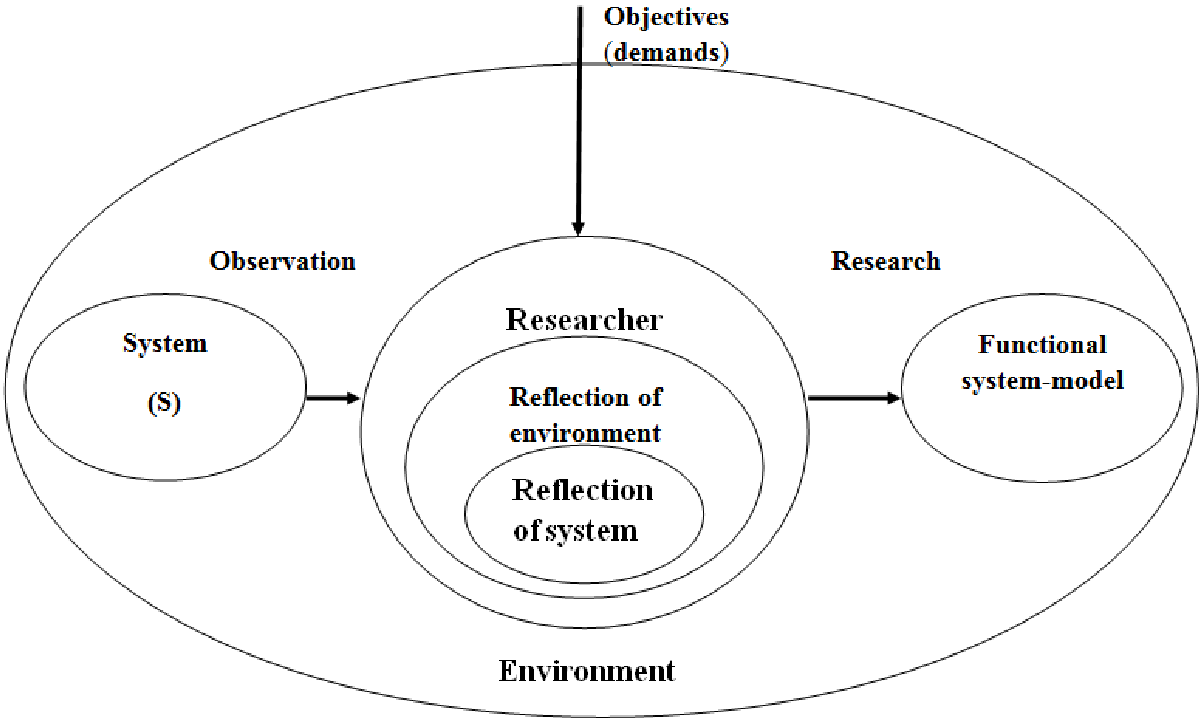


Figure 1.1 – The interaction scheme "system-researcher"

In the "system-researcher" interaction one can distinguish actions, observation and research.

Observation is understood as the effect the system has on sense organs (analyzers) of the researcher. Observation allows researcher to create an image of the system and the environment with which further analysis and synthesis operations are carried out. The images fixed in the researcher's memory are defined as representation.

Research is the process of receiving new knowledge about the systems reflected in images and representations. Research is realized by carrying out the stages of description, explanation and prediction.

Description is representations fixation in the sign system.

Explanation consists in formulating empirical laws and revealing cause-and-effect relations.

Explanation allows to order the facts in time that enables predicting the system behaviour.

Prediction is the description of the system future, its elemental composition, structure and behaviour in relation to the environment.

The result of the research that is carrying out the stages of description, explanation and prediction, is the creation of a formal system which is otherwise called a model.

Model construction is an interactive process the realization of which is directed onto eliminating errors by the researcher at the stages of description, explanation and prediction. The interactive process provides for the performance of images and representations updating, as it is shown in figure 1.1.

Thus, the model is an analogue of the studied system, process or phenomenon which is created for defining the properties of the system, the process, the phenomenon and analyzing their behaviour.

Modeling is understood as the process of model construction on the basis of "researcher-system" interaction scheme specified in fig 1.1.

The *modeling theory* is the interconnected set of regulations, definitions, methods as well as the means of creating and studying these models. These regulations, definitions, methods and means, as well as models, are the subject of the modeling theory.

The modeling theory is a component of the general theory of systems – systematology: the system can be represented by a finite set of models, each of which reflects a certain part (aspect) of its essence.

A model as a formal system is always simpler than the real system which is the object of the researcher's study. Only the basic properties of the investigated system are retained in the model, as they represent the greatest interest for the researcher proceeding from the research objectives.

A model simplification in comparison with the studied model allows to break an information complexity barrier and to conduct the necessary experiments with the model. The information complexity barrier means that the researcher is unable to capture mentally all the set of elements, connections between them, the events occurring in the modeled system.

It is obvious that models are not fewer than modeled systems. To be guided in a range of models it is necessary to classify them. One of the full classifications of model types is given in the work [6] in which the character of processes in the modeled system is used as a classification criterion. The scheme of model types is presented in fig. 1.2.

The scheme gives the basic kinds of models from which other, more difficult kinds of models can be made up. For example, it can be deterministic dynamic discrete model.

Deterministic model does not reflect random factors. *Stochastic* model, on the contrary, is intended for describing processes taking into account random factors. *Static* models reflect the system state at a certain fixed moment of time, and dynamic models show changing of the system state in time. It is obvious, that *dynamic* model is represented as a set of static ones, each of which reflects the system state at certain time points arranged in ascending order.

Discrete models describe the processes, which parameters or time of change, or both have discrete, i.e. discontinuous character.

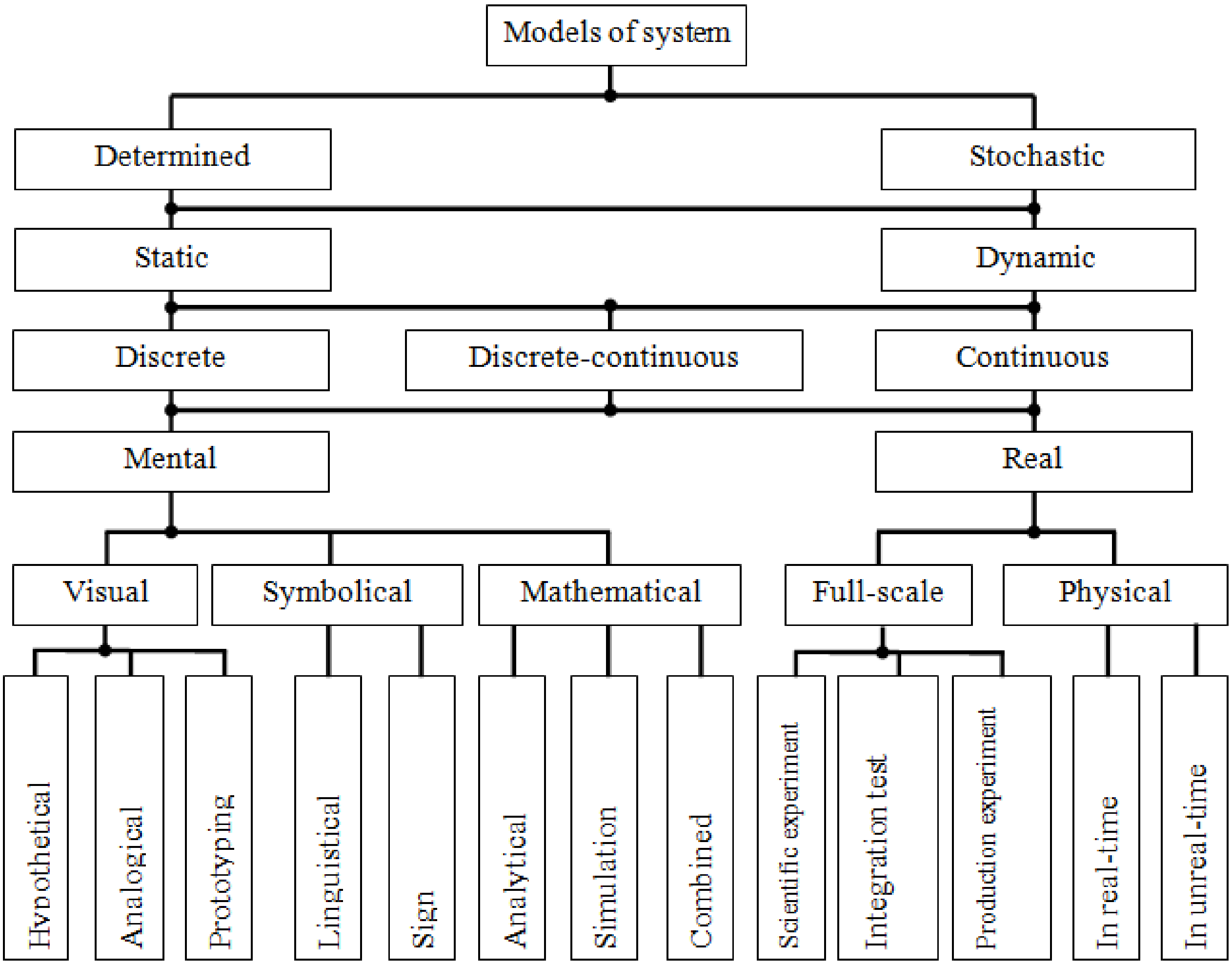


Figure 1.2 – Basic types of models

Continuous models describe the processes which parameters and time of their change are continuous.

The examples of knowledge changes of parameter *x* and time *t* – *x(t)* of a discrete and a continuous processes are shown on fig. 1.3.

Discrete-continuous models describe the systems in which one part of the processes has discrete, and the other one – continuous character.

In case of a discrete process (*a*) the values of the parameter x and time *t* are chosen out of the corresponding final sets $\{x_0, x_1, x_2, x_3, x_4, x_5\}$, $\{t_0, t_1, t_2, t_3, t_4, t_5\}$. In case of a continuous process (*b*) the values of the parameter *x* and time *t* belong to the corresponding sets, represented by segments $[x_0, x_T]$, $[0, T]$, i.e. $x \in [x_0, x_T]$, $t \in [0, T]$.

Depending on the representation form of the modeled system the *mental* (virtual) and *real* models can be distinguished.

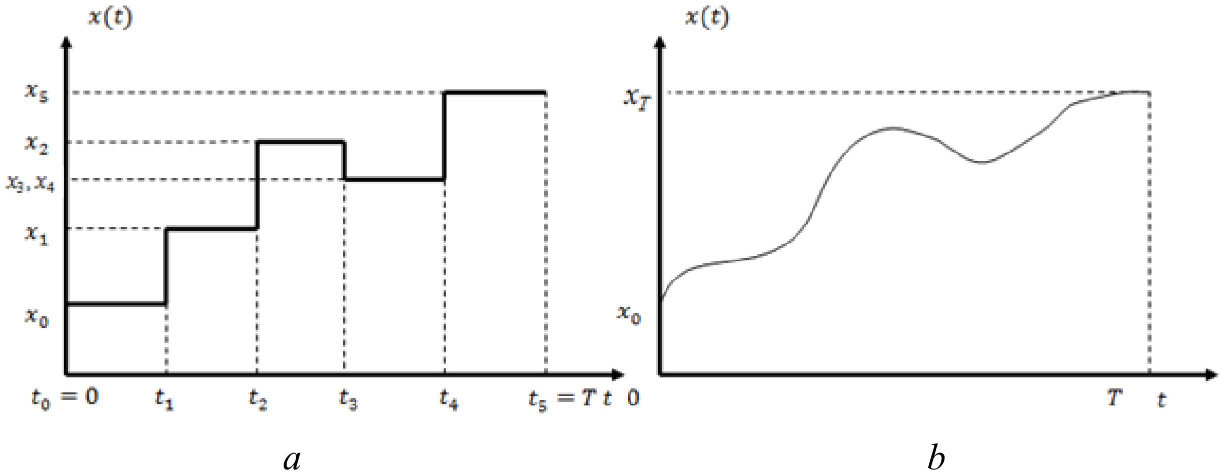


Figure 1.3 – Examples of *x(t)* change of a discrete (*a*) and a continuous (*b*) processes

The necessity of creating mental models arises every time the researcher has difficulties with fixing of real systems in space and time by his sense organs. Difficulties of fixing in space are mainly caused by the sizes of the systems under study, which can be the objects of a microcosm (an atom, a molecule etc.) or objects of a macrocosm (a planet, the Universe etc.). Difficulties of fixing in time are mainly caused by random flow of processes in the investigated system. These processes can occur in fractions of seconds or can occupy centuries.

Main types of mental modeling are visual, symbolical and mathematical ones.

The visual model is created on the basis of the researcher's conceptions about the real system in the form of some visual image – a drawing, a diagram, a prototype model etc.

Visual modeling is subdivided into hypothetical, analogue modeling and prototyping.

In a basis of the *hypothetical* modeling there is some hypothesis about laws of processes in the real system. Thus the hypothesis is understood as a prediction based on low number of experimental data, observations, and guesses.

Analogical modeling is based on analogies of various levels. The highest level is full analogy which can occur in the case of modeling simple objects. The more complex the objects are, the lower the level of the analogies is used. In this case the analogue model reflects several or even one aspect of the modeled complex system.

Mental prototyping is applied when the processes of a real system are not reproduced on installations which retain the nature of the phenomena and possess physical similarity. Mental or imaginable prototype models are based on analogies and cause-effect relationships between the phenomena and the processes in the modeled system.

Symbolical models express the basic properties of the modeled system by means of signs or symbols.

To designate concepts in the *sign* model certain signs connected by operations of algebra of logics are used. By means of signs and operations certain words, sentences and statements describing the modeled system are constructed.

In *linguistical* modeling the thesaurus is used. It differs from a usual dictionary by the absence of ambiguities, i.e. each concept is defined in one word.

The basic means for studying complex systems are *mathematical* models, which are understood as formal systems with elements in the form of mathematical objects (numbers, variables, matrixes, sets etc.) and relations between them, reflecting the properties of the modeled real system.

The type of the mathematical model depends on the nature of the modeled system, the purposes of modeling, the necessary accuracy and solution reliability of the research problems etc. In the classification in figure 1.2 mathematical models are divided into analytical, simulation and combined.

Analytical modeling is applied, when it is possible to represent functioning of the modeled system elements as a functional relationship of some kind – algebraic, integral, differential, finite-difference.

The analytical model allows to carry out the study of the modeled system by three basic methods – analytical, numerical, qualitative.

The analytical method consists in receiving the dependence between parameters of the investigated system in an explicit form.

For example, if an analytical model is represented by differential equation

$$\frac{d^2y}{dt^2} + \omega^2 y(t) = 0,$$

the realization of the analytical research method is reduced to the solution of the given differential equation:

$$y(t) = y_0 \cos \omega t + \frac{\dot{y}_0}{\omega} \sin \omega t,$$

where $y_0 = y(0)$; $\dot{y}_0 = \dot{y}(0)$ – the initial conditions of the differential equation solution.

The numerical method is used, when application of the analytical method is impossible or is accompanied by considerable difficulties. The numerical method consists in receiving some partial solution for the concrete initial data.

The qualitative method is used in case of inapplicability of analytical or numerical methods. It is not directed onto receiving the solution, but enables the possibility of determining some solution properties. For instance, by means of limiting it is sometimes possible to establish the value to which the parameter of the investigated system tends to.

The *simulation model* is an algorithm or a set of algorithms reflecting the functioning process of the modeled system in time.

The basic advantage of the simulation model is the possibility of adequate reflection of more complex systems. This possibility is provided by taking into account such features of complex systems as discreteness, linearity, ambiguity, nonstationarity, etc. which it is difficult to consider in analytical models.

Combined modeling assumes usage of analytical and simulation modeling. In the complex system there can be distinguished the processes that are described by analytical models and those ones that are describe by simulation models. Thus the main thing is the most complete usage of the advantages of each modeling method.

Real modeling provides the possibility of studying various characteristics with the help of either the real system, or its part. The research is conducted both under the normal functioning conditions, and under suggested special conditions, for example, creation of extreme conditions (high or low temperature, high pressure or vacuum etc.). Real modeling is the most adequate, but its usage for studying complex systems is very limited, and for the systems which are only projected, is not probable.

Real modeling, according to the diagram in figure 1.2, is divided into natural and physical.

Full-scale modeling provides conducting research on a real object or its copy with the subsequent processing of the experiments results on the basis of the similarity theory. A well-known example of full-scale modeling is creation of the reduced copy of a plane or a car which is placed in a wind tunnel where aerodynamic properties of the copy are studied. The received results are transformed on the basis of the similarity theory and thus aerodynamic characteristics of a real object are obtained.

The definition of full-scale modeling emphasizes the important role of the experiment either with the real object or with its copy. Therefore, the full-scale modeling has types which correspond to different types of experiment. According to the classification in fig. 1.2 one can single out such types, as scientific experiment, integration test, production experiment.

The main aim of the *scientific experiment* is to understand the essence of the processes by means of experiment automation and information processing. Experiment differs from a usual, normal process by that fact that new factors are introduced, enabling to study the behavior of the real object under extreme conditions and to define stability limits.

One variety of the experiment is *integration test* during which it is possible to collect and process enough statistical data, to reveal regularities of such qualitative characteristics as accuracy, reliability, performance speed etc.

The *production experiment* consists in generalization of implantation and operation experience of new equipment samples or technologies. Meetings, seminars, etc. are the forms of conducting the industrial experiment.

The last type of modeling specified on the diagram in fig. 1.2, is physical modeling. This modeling type differs from the natural one by the fact that experiments are conducted not on real objects, but on installations which retain and reproduce the nature of the phenomena the researchers are interested in. Physical modeling can be conducted in real time and in a special time scale which is called modeling time.

1.2. Requirements to models of complex systems

The simplicity of the model in comparison with the real simulated system, as it was mentioned, enables to break the informational barrier of complexity and to conduct all necessary experiments with the model. The purpose of the planned experiments is determining the model properties and the properties of the system on their basis.

Model simplification in comparison with the simulated system is one of the basic requirements of modeling. However this requirement should come together with other basic requirements obligatory to models - universality, adequacy, accuracy and profitability.

By the model universality degree we mean the completeness of those features of the studied system which are determined by the research objective.

To define other requirements exterior, interior and output parameters are used. The vector of exterior parameters *Z* reflects the properties of the environment. The vector of interior parameters *X* reflects the properties of the system elements, or otherwise reflects the interior content of the system. And, at last, the vector of output parameters *Y* reflects the properties of the system in relation to other systems and the environment.

There is a certain functional relationship between *X, Y, Z*:

$$Y = F(X, Z), \tag{1.1}$$

which represents a mathematical model of the system under corresponding definition *F*.

Let Y_s be a vector of output parameters at the system input, and Y_m – a vector of output parameters at the output of the system model. Then the relative error is defined as

$$\varepsilon = \frac{|Y_m - Y_s|}{Y_s}. \qquad (1.2)$$

To reduce the obtained vector estimation of the relative error to scalar value, a certain norm of vector ε is used, the examples of which can be:

$$\|\varepsilon\| = \sqrt{\sum_{j\in\overline{1,n}} \varepsilon_j^2}\ ; \ \|\varepsilon\| = \max_{j\in 1,n} \varepsilon_j,$$

where ε_j ($j \in \overline{1,n}$) – components of vector ε .

The adequacy of the model is understood as its ability to reflect the given properties of the studied system with the given error, which limiting value can be set by constant $\delta > 0$.

From the expression (1.1), (1.2) follows, that the value of the norm $\|\varepsilon\|$ depends on vectors *X* and *Z*. An important engineering problem is determining *X* minimizing $\|\varepsilon\|$ at the given vector *Z*. If we denote $\|\varepsilon\| = f(X,Y)$ the specified problem can be formalized by the statement

$$\min_{\{x\}} f(X,\bar{Z}) = \varepsilon_m,$$

where $\bar{Z}$ – the given vector of *Z*, i.e. a concrete set of parameters values of the environment.

Obviously that at modification of vector $\bar{Z}$ various values of ε_m are gained. The set $\bar{Z}$ and the corresponding range of values ε_m are used for constructing the area of adequacy which is defined as

$$\{\bar{Z} \mid \varepsilon_m < \delta\}.$$

The profitability of the model is characterized by expenditures of computer resources on its implementation. Resources, first of all, include machine time and random-access memory. Indirect expenditure indicator of these resources is the number of the operations necessary to turn to the model, dimensions and the number of internal parameters of the model.

The demands of accuracy, universality, adequacy, as a rule, contradict the profitability requirement. This inconsistency is solved by the compromise solution which is attained by the implementation of coordinated, mutual concessions in requirements.

1.3. Classification of mathematical models

In the classification diagram in fig. 1.2 mathematical models are divided into analytical, simulation and combined. This is division according to the classification criterion «a mode of representation of the modeling system properties».

There are many other classification schemes, each of which is formed on the basis of a certain classification criterion. Some of them are considered below.

So according to the classification criterion «the character of the reflected properties of the modeling system» mathematical models are divided into structural and functional.

Structural models reflect corresponding structural properties of the modeling system and are divided into topological and geometrical.

By means of topological models the composition and interconnections of elements of the modeling system are reflected. This type of models is often used for solving the problems where the space connection of elements to certain positions and/or time connection of elements to certain instants of time are essential. The examples of such tasks are the arrangement problems of the equipment on industrial areas, location of elements in products of engineering and instrument industry, tracing of junctions in various schemes, schedules conditions of technological processes execution, distributions of powers of decision making between individuals or groups of individuals (structural subdivisions) and many other tasks. All these tasks are the tasks of structural synthesis. The alternative of system structure is the solution of such tasks. The alternative of system structure is understood as the organization of the system from separate elements with their correlations which are defined by the distribution of functions and purposes fulfilled by the system [7].

General enough is the following formal statement of the problem of structural synthesis [7]:

$P = \{\pi\}$ – a set of collections of possible principles for the system construction or its elements;

$F(\pi) = \{f\}$ – a set of interconnected functions (operations) fulfilled by the system which depends on the collection of principles π of the system construction;

$ST = \{str\}$ – a set of the structure variants as the set of possible interconnected elements of the system which can be represented by engineering tools, service stations, certain executors or structural subdivisions of organizations etc.

Let Φ be mapping of the set $F(\pi)$ to the set ST, then Φ: $F(\pi) \to ST$. The required alternative can be $str = \Phi(f)$. The alternative of the structure is optimal if the extreme value of some criterion function is attained at the given restrictions.

There are several levels of complexity of the structural synthesis problem [8]. Most of the problems belong to the third level of complexity and the class of so-called NP-complete problems. It is the class of combinatorial problems for the exact solution of which there exist only algorithms of exponential complexity, i.e. the algorithms in which the number of the required operations is proportional to the exponent from the problem dimension.

Topological models often are presented in the form of graphs, tables, matrixes, lists, etc.

Another type of structural models – geometrical models – is used for reflecting geometrical properties of the modeling object. In addition to the information about positional relationship of the elements in geometrical models there is some information about the form of the elements making up the system.

Geometrical models are represented by equations which describe lines and the surface of the elements; algebraic and logical relations which describe the areas making up the body of the element; graphs and lists reflecting the constructions from typical constructive elements [9].

The theoretical basis of geometrical models creation is analytic geometry, set theory, differential geometry, graph theory, algebra of logics.

Geometrical models have their own classification [10]. We will single out the well-known and widely used types in it – analytical and algebrological geometrical models.

An example of an analytical geometrical model can be general control of the second-order curve on the plain in the Cartesian rectangular coordinates

$$F(x,y) = ax^2 + 2bxy + cy^2 + 2dx + 2ey + g = 0,$$

where a, b, c, d, e, g – the constant coefficients.

The curve $F(x,y)$ is used for describing the contour of the element. For describing the surface of revolution the following equation can be used

$$F(x,y,z) = a(x^2 + y^2) + bz^2 + 2cz + d = 0,$$

where x, y, z – the axes of coordinates.

Analytical models, as a rule, serve for describing elementary geometrical objects on the basis of which some composite geometrical objects can be received.

By means of algebralogical geometrical models flat figures and bodies in three-dimensional space are presented. The model represents a logic function of the conditions of the points membership to the given area.

For example, it is necessary to design a model describing a plain figure in the form of triangle D in fig. 1.4.

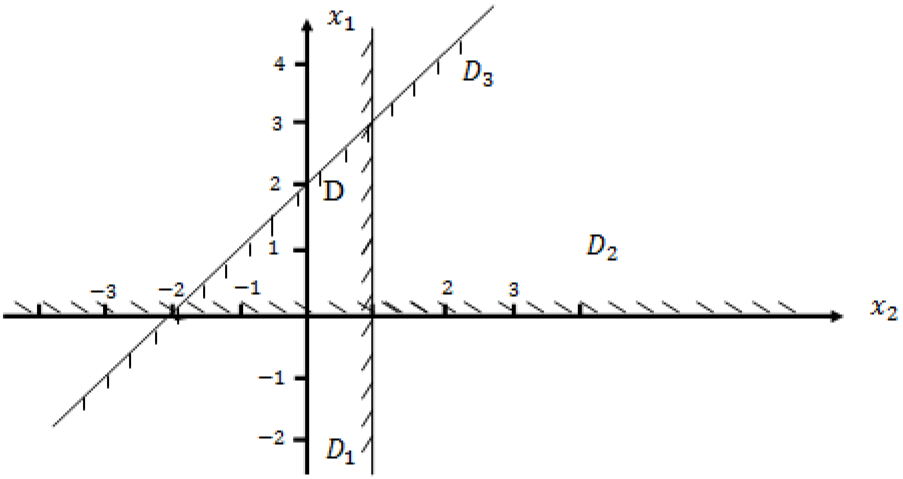


Figure 1.4 – Plain figure *D* in the space $x_1 0 x_2$

The model of figure D is the logic expression

$$D = D_1 \cap D_2 \cap D_3,$$

where domains D_1, D_2 and D_3 are defined by inequalities:

$$D_1 : x_2 \le 1,$$

$$D_2 : x_1 \ge 0,$$

$$D_3 : x_1 - x_2 \le 2.$$

Other types of geometrical models – canonical, frame, receptor, mathematical etc. are considered in details in special literature [10].

Unlike structural models, functional ones reflect physical or informational processes in the modeling system during its functioning or creation.

Since functional models reflect the processes taking part in the operating system one can use the classification of the models by the character of the processes given in fig. 1.2. That is, as well as processes, functional models are subdivided into discrete, continuous, stochastic etc.

Defining the model type, proceeding from the character of the process, enables to choose certain mathematical apparatus or scheme for model construction. For instance, models of continuous-determined processes are created on the basis of mathematical apparatus of differential equations, models of discrete-determined processes are created on the basis of the finite automata theory etc.

Besides the character of the process, physical nature of the process or the aspect of the description is important for the definition of the functional mathematical model. The definition of the description aspects leads to allocation of electric, mechanical, hydraulic, optical, chemical models etc. The specified aspects enable to choose laws and regularities of processes. These laws form the basis for the functional model construction. For instance, very often the basis of mechanical systems models are Newton's and Hook's laws etc., for electric systems – Ohm, Kirchhoff's laws etc.

For example, it is necessary to develop a functional model of such a system, as a spring pendulum in fig. 1.5

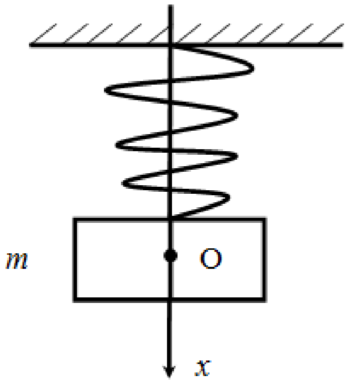


Figure 1.5 – The view of a spring pendulum in the equilibrium position

For the considered system the most important is the process of its movement which has continuous-determined character. Hence, for the construction of a mathematical functional model of the considered system's movement the mathematical apparatus or the scheme of ordinary differential equations can be used.

To more precisely define the mathematical model expression it is necessary to establish the physical nature (aspect) of the movement process and the laws of this movement. The considered system is mechanical by its nature. The law defining the system dynamics is Newton's 2nd law, according to which

$$a \cdot m = F, \tag{1.3}$$

where a – acceleration, F – force of the spring stretching and compression, affecting mass m.

According to the Hook's law $F = -kx$, where x is deviation of mass m from the equilibrium position.

It was earlier noticed, that for the model construction it is necessary to use apparatus (scheme) of differential equations. Therefore acceleration a is represented as $a = \frac{d^2x}{dt^2}$. Taking this into account, expression (1.3) takes the form:

$$\frac{d^2x}{dt^2}m = -kx.$$

Introducing the known denotation $\omega^2 = \frac{k}{m}$ (ω- oscillation frequency), we receive the final version of the movement (dynamics) functional mathematical model of the considered system in the form of a spring pendulum:

$$\frac{d^2x}{dt^2} + \omega^2 x(t) = 0.$$

Thus, the model is ordinary, homogeneous, differential equation of the second order.

The hierarchy structure of existing complex systems and the block-hierarchical principle of their designing predetermine the hierarchy of the corresponding mathematical models.

Depending on the place in the description hierarchy functional and structural mathematical models are usually divided into three types, each of which corresponds to one of the three generalized levels – micro-, macro-, metalevel. As a rule, the model of a higher level is aggregated.

Formally aggregation consists in replacing a certain group of variables reflecting the system condition with one variable. Physically aggregation corresponds to the integrated space discretization according to a functional indicator. At a metalevel the elements can be represented by the collection of elements of lower levels.

In a sufficiently general overview the construction problem of the aggregated model can be formulated as follows [8].

Let the initial multidimensional model represent a system of equations

$$y_i = f_i(x_1, x_2,..., x_n),\ i \in \overline{1,n}, \tag{1.4}$$

where $x = (x_1, x_2,..., x_n)$ are inputs, and $y = (y_1, y_2,..., y_n)$ are the model outputs.

Aggregation consists in the introduction of the aggregated variables and the system of equations connecting them.

The aggregated variables are expressed through variables of the initial model:

$$z_k = \psi_k(x_1, x_2,..., x_n);$$

$$u_k = \varphi_k(y_1, y_2,..., y_n),\ (k \in \overline{1,m}),\ (m < n). \tag{1.5}$$

The system of equations connecting aggregated variables looks like:

$$u_k = s_k(z_1, z_2,..., z_k),\ k \in \overline{1,m}, \tag{1.6}$$

where z_k, u_k are considered as inputs and outputs of the aggregated model accordingly.

If we designate $x = (x_1, x_2,..., x_n)$, then the condition of adequate aggregation is

$$\forall x \in X \quad \varphi_k\left(f_1(x), f_2(x),..., f_n(x)\right) = s_k\left(\psi_1(x), \psi_2(x),..., \psi_m(x)\right), \quad (1.7)$$

where X – is the set of inputs of the initial model (1.4).

As a matter of fact, the condition of adequate aggregation is the condition of the aggregation system compatibility (1.5), (1.6) for the initial model (1.4), and the aggregation system satisfying this condition is called a joint aggregation system.

The construction of the joint aggregation system requires defining certain additional conditions imposed on functions f, ψ, φ, s. These additional conditions lead to big computational expenditures and often have no physical interpretation.

Refusal to satisfy the compatibility conditions generates a big number of aggregation criteria and modes among which there exist methods of linear models aggregation, methods of descriptive aggregation, methods of function minimization from an aggregation error and from a simplification error [8].

Methods of linear models aggregation are based on the analysis of the coefficient matrixes of the initial model (1.4) which enables to select the variables boundaries, each of which can be represented by one aggregated variable.

For descriptive aggregation the initial model f and the aggregation functions ψ, φ are fixed (1.5), and the aggregated model s (1.6) should satisfy the condition (1.7) not for all $x \in X$, but only for a certain base vector $x^0 \in X$.

For realization of the aggregation error minimization methods the error is usually understood as the difference between the right and left parts of the compatibility condition (1.7).

The use of the simplified model as the bases of the function minimization method from a simplification error is expedient in cases when the initial model (1.4) is complicated enough from the computing point of view. The simplification error is defined as the difference between outputs of the simplified and the initial models.

1.4. Methods and technique of the process of mathematical model construction

Certain informal and formal methods are used for the construction of the mathematical model [8].

Informal methods are used for obtaining mathematical models of the system elements arranged at the most different hierarchical levels.

The realization of informal methods includes studying the processes and phenomena regularity in the modeling system or in its elements, allocating certain essential factors, admitting various sorts of assumptions and their substantiation, mathematical interpretation of the available information etc.

All the specified operations of the mathematical model construction are realized by qualified experts. Indicators of the mathematical model efficiency depend on successful realization of these operations – universality degree, accuracy, profitability.

The purpose of studying regularities of processes and phenomena is to determine the nature of the processes and phenomena as well as the laws to which they submit. The example of this operation realization has already been considered while designing the model of the spring pendulum which view is shown in fig. 1.5.

Allocation of essential factors is an important informal operation the realization of which is possible with the help of experts in the subject domain. From experience of mathematical models construction it is known, that 20 % of factors determine 80 % of the modeling system properties, and the remaining 80 % of factors determine 20 % of properties. The problem consists in defining these 20 % of essential factors. Their definition is realized by means of expert methods which presuppose obtaining expert estimations and their processing by statistics methods with introduction of new factors taking into consideration the importance of this or that expert's estimation [12].

Acceptance of various assumptions and their substantiation are mainly used for model simplification. Simplification consisting in replacing the nonlinear model of the real process with some set of linear models is often used. The essence of this replacement is shown in fig. 1.6.

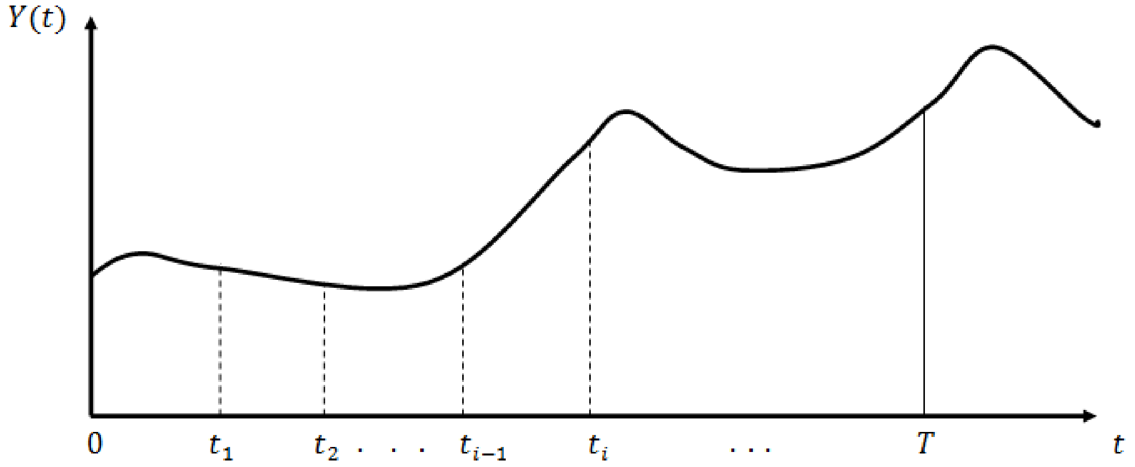


Figure 1.6 – The example of the characteristic $y(t)$ modification

The output characteristic $y(t)$ on the interval $[0,T]$ has explicitly expressed nonlinear character. Thus regularity $y(t)$ is complicated enough and consequently there arises the necessity for its simplification. For this purpose interval $[0,T]$ is divide into smaller intervals $[t_{i-1},t_i]$ $\left(i\in\overline{1,n},\, t_0=0, t_n=T\right)$ and the assumption is made about the linearity of the modification law of y on the interval $[t_{i-1},t_i]$, i.e. $\tilde{y}_i(t)=a_i+b_i t$, $t\in[t_{i-1},t_i]$. The substantiation of such assumption is the fulfillment of the condition

$$\forall t\in[t_{i-1},t_i]\quad |y(t)-\tilde{y}_i(t)|<\varepsilon, \tag{1.8}$$

where ε – is a permissible deviation error of the nonlinear characteristic $y(t)$ from linear $\tilde{y}_i(t)$ on the interval $[t_{i-1},t_i]$.

If the condition (1.8) is fulfilled for all intervals $i\in\overline{1,n}$ then one nonlinear characteristic of the real process $y(t)$ can be replaced with the set of n linear models $\{\tilde{y}_i(t)=a_i+b_i t\mid i\in\overline{1,n}\}$.

Under mathematical interpretation of the available information we consider empirical interpretation as explanation of objects of formal and substantial theories by means of the real world categories [12, 13]. In the process of empirical interpretation, which is typical for applied mathematics, the concepts of the theory are connected with the objects and relations of physical space. For example, one of the widely used theories of applied mathematics is the graph theory which basic notions are a node, an edge, an arc. Connecting these concepts with settlements and roads or with the elements of electric schemes (resistors, capacitors etc.) and junctions between them, one can create graph models for studying transport systems or electric circuits accordingly.

Informal methods are used for developing both theoretical and empirical mathematical models. Theoretical models are the research result of the processes and their regularities which are typical for the class of objects and phenomena under the study.

Empirical models are created to study the system properties which it shows in relation to the environment, i.e. in relation to its systems. These models often belong to «a black box» type

$$Y=F(Z), \tag{1.9}$$

which basic difference from models (1.1) consists in failing to take into account the interior system structure expressed by the vector of the interior parameters of system *X*.

To define *F* in model (1.9) the experiments are conducted which consist in modification of vector *Z* according to the previously developed plan and fixing the corresponding values of vector *Y*. As a result we obtain so-called «statistics of modeling» $\{Y_j, Z_j\mid j\in\overline{1,m}\}$, where *j* – is the experiment index, *m* – the general number of experiments, Y_j, Z_j – vectors *Y*, *Z* in *j*-th experiment.

Procession of modeling statistics enables to define *F*. While processing, first, a hypothesis about the model structure (about the model type) is put forward, and then the model parameters are determined, often on the basis of least-squares method [11].

Thus, on the whole, the technique of deriving the mathematical model of the system elements on the basis of informal methods presupposes sequence of the following operations.

1. The properties to be reflected in the model are defined. The choice of properties depends on the goals and tasks of modeling and defines the universality degree of the mathematical model.

2. Information preparation on each property chosen in art. 1. Information sources are database and knowledge bases, scientific and technical literature with the description of the previously fulfilled working outs of similar mathematical models, experiments results, experience and knowledge of experts etc.

3. Elaboration of the structure of the mathematical model which is understood as the general view of mathematical relations without indicating numerical values of the model parameters. Structure *St* and the vector of parameters *C* are the pair of the model *F* in (1.1), i.e.

$$F = \langle St, C \rangle . \tag{1.10}$$

It is known that the structure is understood as a certain set of elements which compose the object under the study with the relations between the elements.

Structures of many real objects are represented by the graph which nodes correspond to the object elements, and arcs correspond to relations between the elements.

For the representation of the mathematical model structure such categories as dimension, linearity, nonlinearity, determinacy, stochasticity, discreteness, continuity etc. are used. The model structure is set by enumeration of the specified categories. For example, a two-dimensional, static, determined, linear, continuous, stationary model (1.1) can be presented as

$$y = C_1 + C_2 x + C_3 z , \tag{1.11}$$

where x, y, z – are the scalars, C_1, C_2, C_3 – are the constant factors, which values are undefined.

If substitute the category “static” for the category “dynamic” (1.11) can be presented as

$$y(t) = C_1 + C_2 x(t) + C_3 z(t) .$$

Besides, if substitute the category “stationary” for “nonstationary” (1.11) becomes

$$y(t) = C_1(t) + C_2(t)x(t) + C_3(t)z(t),$$

i.e. both the condition parameters and the parameters defining properties of the system or elements are functions of time.

4. The calculation of parameters' numerical values *C* of the mathematical model in the pair (1.10). The problem of defining parameters $C = (c_1, c_2, ..., c_k)$ under the condition of the modeled object normal functioning, i.e. without feeding the object with special disturbances (operating impacts) is called the problem of the identification of the model parameters.

To solve the identification problem the information about the structure St is used as well as observation over the input and output of the modeled object at its interaction with the environment. If to fix the properties of the systems elements expressed by vector *X*, the necessary information for defining parameters *C* of model (1.1) is represented in the pair

$$I = \langle Z_i, Y_i \rangle, \; (i \in \overline{1,n}),$$

where Z_i, Y_i – are the vectors at the moment of observation $t_i \in [0,T]$ from interval $[0,T]$, *n* – number of observations.

Determination of *C* is reduced to the realization of algorithm φ on input data in the form of structure St and observations *I*, i.e.

$$C = \varphi(St, I). \tag{1.12}$$

Algorithms φ of the solution of the identification problem are divided into two big classes – adaptive and nonadaptive.

Adaptive algorithm φ_a is understood as the algorithm enabling to obtain value of parameters C_{i+1} on $(i+1)$ algorithm step with the help of the values of parameters C_i on the previous *i*-th step and the information about the values of inputs and outputs of modeling objects on $(i+1)$ step $I_{i+1} = \langle Y_{i+1}, Z_{i+1} \rangle$, i.e.

$$C_{i+1} = \varphi_a(C_i, I_{i+1}).$$

Thus, the adaptive algorithm enables to receive the parameters value by realization of sequence from n steps. Thus at each step we use only the information about the state of inputs and outputs at the given step and the parameters value at the previous step of the algorithm.

Unlike an adaptive algorithm, a nonadaptive one assumes the use of all the information about the state of inputs, outputs of the modeling object $I = \langle Z_i, Y_i \rangle$ $i \in \overline{1,n}$, i.e. a nonadaptive algorithm is, as a matter of fact, an algorithm φ in (1.12). An example of a nonadaptive algorithm is the algorithm based on the least-squares methods [13].

5. An evaluation of difficulty and adequacy of the mathematical model on the basis of (1.2) with the construction of the adequacy area and their various approximations [12].

Formal methods are mainly used for the development of the system mathematical models on the basis of available models of system elements and relations between the elements. Very frequently the elements' models are represented by component equations, and the relations between elements are considered by means of topological equations [8].

1.5. Body of knowledge of modeling and simulation

A *body of knowledge* (BOK) of the modeling and simulation (M&S) is being developed under the auspices of the National Training Systems Association (NTSA). The M&SBOK is open to the simulation professionals. A review committee is already formed for this purpose. Membership of the review committee is open to concerned professionals. To benefit from a proactive approach, a mechanism is in place to receive the recommendations of the colleagues.

There are several interpretations of M&SBOK. Prom a pragmatic perspective, body of knowledge is:

1) Structured knowledge that is used by members of a discipline to guide their practice or work.

2) The prescribed aggregation of knowledge in a particular area an individual is expected to have mastered to be considered or certified as a practitioner.

3) BOK is a stepping stone to unifying community.

The development of M&SBOK is substantiated by the existence of BOKs in other disciplines. Some disciplines have several BOKs to provide their different aspects. In the domain of quality management:

- Body of Quality Knowledge;
- BOK in Quality and Performance Improvement;
- Body of Knowledge for National Board Inspector Commission Examination;
- (American Society for Quality) Six Sigma Body of Knowledge;
- (American Society for Quality) Quality Technician Certification BOK.

There are BOKs in the domain of safety:

- BOK for Safety Practitioners;
- BOK of the American Society of Safety Engineers Council on Practices and Standards.

Various BOKs in computer science and software engineering are available for professionals in those areas:

- BOK of Software Engineering (SWEBOK);
- Personal Software Process BOK;
- BOK for Software Quality Measurement;
- Software Assurance BOK;
- BOK for the Certified Software Tester;
- BOK About Software Reading Techniques;
- Computer Science BOK.

In system engineering we can name the following body of knowledge:

- Systems Engineering BOK;
- Information Systems Engineering BOK;
- Information Systems Security Engineering Professionals BOK.

The fact that some disciplines have several BOKs is indicative that M&S, which has several aspects, may also need more than one BOK.

The concept of modeling has already been considered in this chapter. The notion of simulation has two directions of definition. As a process, the term simulation has a non-technical and a technical meaning. As a non-technical term, simulation means imitation.

As a technical term simulation has two aspects, i.e., gaining experience and performing experimentation. From the point of view of experimentation, simulation is goal-directed experimentation using dynamic models. From the point of view of experience, simulation is a technique to gain experience under controlled conditions.

The technical meanings are used since 1950s. They cover any type of simulation regardless whether it is computerized or not and whether it is carried out on pure software or hardware/software. Furthermore, both of the technical definitions ease the top down decomposition of the entities and activities involved.

M&S is used in different application areas in training, education, decision making, and entertainment. From the point of view of experimentation, application domain includes all types of engineering areas as well as most nonengineering areas such as arts, biology, chemistry, management, medicine, and physics as well as education and entertainment.

In military applications, most recommended topics are: war gaming, missile defense, communication and computers, scene generation, visualization, animation, and seeker, sensor, and signal processing.

There are two types of simulation activities, depending whether or not simulation program runs independently from the system it represents. Hence, there are standalone and integrated simulation activities

Standalone simulation activities are those ones where the simulation program runs independently from the system of interest. Simulation is used for the following purposes:

- training to improve motor skills, decision and communication skills, operational skills by getting real-life-like experience in a controlled environment;
- education;
- decision making;
- entertainment (simulation games, animation of dynamic systems).

In particular we can say that use of simulation for decision making is done for the following activities:

- prediction of behavior or system performance within the constraints inherent in the simulation model;
- evaluation of alternative models, parameters, experimental and operating conditions on model behavior or performance;
- sensitivity analysis;
- engineering design;
- prototyping;
- planning;
- acquisition;
- proof of concept.

Integrated simulation activities imply that simulation program operates together with the system. Two main purposes are to support and to enrich real system operation. To support real system operation, the system of interest and the simulation program operate alternately to provide predictive displays. To enrich the real system operation, the system of interest and the simulation program operate simultaneously to assure on-line diagnosis and augmented reality operation.

M&S can be perceived from different aspects such as:

- infrastructure to support real-world activities;
- computational activity;
- systemic activity;
- model-based activity;
- knowledge-generation activity;
- knowledge-processing activity.

M&S as an infrastructure supports real-world activities is the black box perception by practitioners, for whom simulation is a tool to achieve other goals. This view allows to concentrate to the original problems they face. From this perspective, simulation is perceived as not being the real thing.

M&S as a computational activity emphasizes at different levels: from the generation of model behavior to simulation-based problem solving environments. This point of view is reflected in some definitions of simulation, e.g., "Simulation is the execution over time of models representing the attributes of one or more entities or processes."

M&S as a systemic activity is used to find the values of output, input, or state variables of a system; provided that the values of the two other types of variables are known. System sciences provide the basis for modeling formalisms as well as for symbolic processing of models, for a large variety of dynamic systems, including goal-directed systems, variable structure systems, evolutionary systems.

M&S as a model-based activity allows construction of simulation-based computer-aided problem solving environments.

M&S as a knowledge-generation activity which implies that simulation is a goal-directed activity with dynamic models. This view allows advanced methodologists and technologists to integrate simulation with several other knowledge generation techniques.

M&S as a knowledge-processing activity allows advanced methodologists and technologists to integrate simulation with several other knowledge processing techniques.

There are different stakeholders of M&SBOK. It is natural that different people may be interested in different aspects of M&SBOK. From a pragmatic aspect, a good M&SBOK may highlight the salient features knowledge of M&S in all application areas. This knowledge can be used, for several categories of usages of M&S:

- to prepare the workforce, by providing guidelines for curriculum development for undergraduate and graduate education as well as for professional development;
- as a basis for professional certifications;
- as a guideline for the current and future simulation professionals for career planning and self-assessment.

Anyone may use M&SBOK to explore the discipline and determine whether it is useful for himself. Practitioners may manage some specific problem solving with the help of M&SBOK. Students and any learners should expand their knowledge based on the guidelines provided by M&SBOK. Academicians should elaborate the curricula and the courseware of engineering specialties based on M&SBOK. Enterprises and industry can make employers certification with respect to knowledge presented in M&SBOK. Licensing and certification agencies may be guided by M&SBOK in their activities connected with determination of professional standards for academic curricula and commercial organizations.

Basically a practitioner of M&S, needs to know three types of knowledge which are:

1) knowledge of the application area;
2) knowledge of M&S that will be referred to as the core elements;
3) knowledge of supporting areas.

The structure of M&SBOK includes four parts. The first one is devoted to M&S background, its terminology and comprehensive overview. The second part describes the core areas of M&S. The third part is related to supporting domains. The final part represents useful references in M&S domain, which includes M&S portals, societies, blogs and so on.

The core areas have to be elaborated separately, benefiting from critical view of other M&SBOK studies as well as M&S taxonomies and ontologies. The core areas include the following:

- science / methodology;
- types of simulation;
- life cycles of M&S;
- technology;
- infrastructure;
- reliability;
- ethics;
- history;
- trends, challenges, and desirable features;
- enterprise;
- maturity.

Let's describe some of them in more details. The sub-areas of the science / methodology area present the fundamentals of M&S (Tables 1.1).

Table 1.1 – Sub-areas of science / methodology area

№	Sub-area	Description

1	Data	Issues;
		types of variables;
		types of values.

2	Models	Issues connected with reusability, interoperability, composability, dynamic composability; conceptual models and conceptual modeling; taxonomy of simulation models; modeling formalisms; modeling physical systems; modeling qualitative systems.
3	Model building	Modeling; Model composition.

Continuation of table 1.1

№	Sub-area	Description

4	Model-base management	Model search;
		semantic model search;
		model integrity.

5	Model Parameters and Parameter-base Management	Parameters;
		auxiliary parameters;
		deterministic parameters;
		stochastic parameters

6	Model characterization	Model documentation;
		model usability;
		model referability.

7	Model evaluation	Evaluation of static structure; evaluation of dynamic structure
8	Model comparison	Structural model comparison: model verification (types of and techniques and tools for model verification); model checking (for homomorphism, isomorphism, endomorphism); model equivalencing. Behavioral model comparison (under same or different scenarios)
9	Technical System Specifications	Model qualification (model realism, model adequacy, model correctness analysis); model validity
10	Experimentation	Statistical design of experiments; computer-aided systems for design of experiments; computer-aided systems for execution of experiments data compression techniques (deterministic, stochastic); analysis of simulation data
11	Model behavior	Types of model behavior; generation of model behavior; processing of model behavior visualization of model behavior

The area of technology represents the M&S languages, tools and environments. Reliability area covers questions related to quality assurance of M&S and types of errors, validation and verification.

M&SBOK defines supporting domains for M&S, which include computers and computation, supporting science areas, supporting engineering areas, supporting management areas and education.

The computation supporting domain takes into account the digital, hybrid, analog, mobile, cloud computing; fuzzy logic and neural networks; agent-based models, agent-supported and agent-monitored simulation. Scientific areas include system analysis, physics, mathematics and queuing theory. Among engineering areas system engineering and visualization are mentioned. Enterprise, project and product management refer to management areas.

M&SBOK represents the guidelines for the professionals in software engineering. Since modeling and particularly simulation are the essential part of any applied engineering problem solving, knowledge of fundamentals of M&S are relevant for professionals in the information technologies domain. A BOK for M&S can be very useful to provide a comprehensive and integrative view of the discipline, for self-assessment and for assessment of professionals and organizations, as well as for curriculum and courseware elaboration in corresponding higher education institutions.

Summary

Modeling is an essential part of any engineering activities. It allows to research complex objects, determine their characteristics and predict their behavior. Specialists in information technologies often face with these tasks. This chapter introduces the concepts of model and modeling. The classification of mathematical models represents those models that will be described in details in the next chapters. To provide the professionals in engineering disciplines with the common understanding of modeling fundamentals the body of knowledge of modeling and simulation should be used. Its overview is given in the last section.

Tasks for case-study

1. Let’s imagine that we have a local network switch as a research object. Describe the functioning of this object. Specify the peculiarities of functioning (continuance or discreteness of state transition, presence or absence of random factors). Define the goals of modeling. Identify the type of modeling for this object.
2. Let’s imagine that we have the arithmetic and logic unit of computer as a research object. Describe the functioning of this object. Specify the peculiarities of functioning (continuance or discreteness of state transition, presence or absence of random factors). Define the goals of modeling. Identify the type of modeling for this object.
3. Let’s consider the infection of population as a research object. Describe the functioning of this object. Specify the peculiarities of functioning (continuance or discreteness of state transition, presence or absence of random factors). Define the goals of modeling. Identify the type of modeling for this object.

4. Let’s imagine that we have the memory cells of computer as research object. Describe the functioning of this object. Specify the peculiarities of functioning (continuance or discreteness of state transition, presence or absence of random factors). Define the goals of modeling. Identify the type of modeling for this object.
5. Let’s imagine that we have the public transport system as research object. Describe the functioning of this object. Specify the peculiarities of functioning (continuance or discreteness of state transition, presence or absence of random factors). Define the goals of modeling. Identify the type of modeling for this object.
6. Let’s imagine that we have the retail business with some sales outlets as research object. Describe the functioning of this object. Specify the peculiarities of functioning (continuance or discreteness of state transition, presence or absence of random factors). Define the goals of modeling. Identify the type of modeling for this object.
7. Let’s consider the struggle for existence between predators and their prey as research object. Describe the functioning of this object. Specify the peculiarities of functioning (continuance or discreteness of state transition, presence or absence of random factors). Define the goals of modeling. Identify the type of modeling for this object.
8. Let’s imagine that we have a call center of a service center as research object. Describe the functioning of this object. Specify the peculiarities of functioning (continuance or discreteness of state transition, presence or absence of random factors). Define the goals of modeling. Identify the type of modeling for this object.
9. Let’s imagine that we have the customers service at the bank as research object. Describe the functioning of this object. Specify the peculiarities of functioning (continuance or discreteness of state transition, presence or absence of random factors). Define the goals of modeling. Identify the type of modeling for this object.
10. Let’s imagine that we have a simple telephone system as research object. Describe the functioning of this object. Specify the peculiarities of functioning (continuance or discreteness of state transition, presence or absence of random factors). Define the goals of modeling. Identify the type of modeling for this object.
11. Let’s imagine that we have a turnstile at a football stadium as researched object. Describe the functioning of this object. Specify the peculiarities of functioning (continuance or discreteness of state transition, presence or absence of random factors). Define the goals of modeling. Identify the type of modeling for this object.

Check questions

1. Why do we need the modeling?
2. Give the definition of the notion “model”.
3. Give the definition of the notion “modeling”.
4. Provide some examples of models surrounding us in our ordinary life.
5. Why do we use models in the everyday life?
6. What kinds of models do you face in your life?
7. What kinds of models do you know?
8. What is the main goal of modeling?
9. Why do people use different models?
10. What targets of modeling can you underline?
11. Give your own definition of the notion “modeling”.
12. Provide some examples of modeling application for the study of a new object or a phenomenon.
13. Provide some examples of modeling application in ordinary life.
14. Provide some examples of modeling application in your educational process.
15. What are the main types of modeling?
16. Give the definition of the static modeling.
17. Give the definition of the dynamic modeling.
18. Give the definition of the determined modeling.
19. Give the definition of the stochastic modeling.
20. Give the definition of the discrete modeling.
21. Give the definition of the continuous modeling.
22. Give the definition of a dynamical system.
23. What types of dynamical systems do you know?
24. What is a mathematical scheme?
25. What types of schemes did we emphasize?
26. In what cases do you need to use dynamical models?
27. What requirements can be formulated to the models of complex systems?
28. What are the steps of construction of mathematical models of systems elements based on non-functional methods?
29. What are the goals of M&SBOK creation?
30. Who can use knowledge represented in M&SBOK?
31. What are the core areas modeling described in M&SBOK?

2. ANALYTICAL MODELING OF THE MAIN TYPES OF PROCESSES IN THE COMPLEX SYSTEMS

2.1. Typical mathematical schemas of analytical models of the main types of processes

The development of the analytical model is realized on the basis of the mathematical scheme which is understood as the apparatus, or a mathematical tool which allows to represent the character of the investigated process in the most comprehensive way [6]. Differential equations, finite-difference equations etc., can serve as the examples of the mathematical schemes.

The main types of processes in the complex systems in accordance with their character include: continuous, discrete, determined, and stochastic. Most often the objects of research are combinations of the specified main types: *continuous-determined, discrete-determined, discrete-stochastic and continuous-stochastic* processes. For developing analytical models of these processes the corresponding mathematical schemes are used which are defined as typical mathematical schemes.

Differential equations are used as the typical mathematical scheme for developing the analytical model of the continuous-determined processes; the finite automata – for the discrete-determined, the probabilistic automata – for the discrete-stochastic and the queue systems – for the continuous-stochastic processes respectively.

No conceptual problems appear when all the processes in the system are of the single type, that is, when all processes are for example continuous-determined or discrete-determined. In this case the system model is represented by the system of models, each of which reflects the corresponding process and is developed on the basis of a certain typical mathematical scheme, common for all models. Thus, if several continuous-determined processes run in the system, their model can be represented by the system of differential equations.

Additional difficulties arise, when several processes of different type take place in the system. Difficulties are caused by the fact that, as a rule, models of different type processes do not form the system of models.

Overcoming of the mentioned difficulty required to develop a unified mathematical scheme which would enable to reflect various types of processes uniformly. In the nineteen sixties of the last century the group of scientists supervised by professor Buslenko N.P. suggested the unified mathematical scheme called “the aggregation” [14]. All above mentioned typical mathematical schemes are special cases of the aggregation and can be derived from it by introducing certain limitations or conditions.

2.2. Continuous-determined modeling

For developing continuous-determined models such typical mathematical scheme as differential equations is used.

It is well known that equations are called differential if their structure includes unknown function of one or several variables and its derivatives of various orders.

If the unknown function is the function of the single variable, then the differential equation is the ordinary one. If the unknown function is the function of several variables (two and more) the differential equation is the partial one.

In the English-language literature the continuous-determined models are often called D-schemas (Dynamic), thereby emphasizing, that this type of models is intended for studying the dynamics of the continuous-determined processes, that is, for studying the temporal variation of characteristics of these processes.

For deriving a continuous-determined model in the form of a differential equation or a system of differential equations, firstly, it is necessary to identify the nature of the studied processes which can be mechanical, electric, thermodynamic etc., and, secondly, to know and to be able to use the laws explaining the process behavior.

It should be noted that the specified operations, related to identification of the processes nature and of the laws they are subject to, are necessary for deriving not only continuous-determined, but also the other types of models.

Let's consider the application of these operations for the construction of continuous-determined models of the functioning processes of some particular systems.

Let there be the spring pendulum of mass *m* which moves, touching the cylinder walls as shown in fig. 2.1. Like in the case of the spring pendulum in fig. 1.5, the process of mass *m* movement has a continuous-determined character and is explained by the laws of mechanics: the Newton's second law, the Hooke’s law and the laws of mass *m* friction against the cylinder walls.

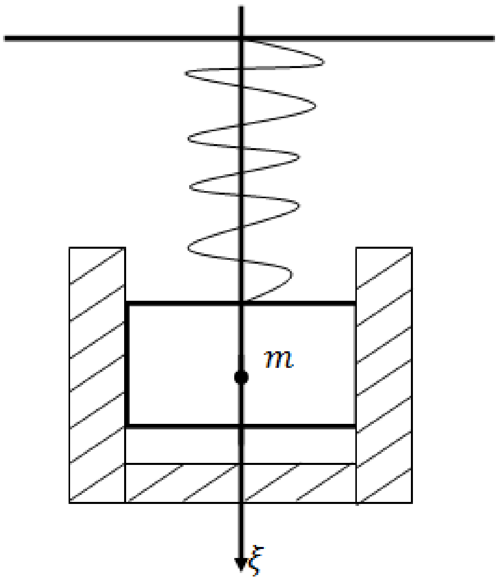


Figure 2.1 – The mechanical system «a spring pendulum in a cylinder»

According to the second Newton's law

$$a \cdot m = F,$$

where *a* is mass *m* acceleration, *F* is the force influencing mass *m* consisting of two forces –

$$F = F_1 + F_2.$$

F_1 is the force of the spring stretching-compression which, according to the Hooke’s law, depends on the deviation value ξ of mass *m* from the equilibrium position and is directed against the movement, that is

$$F_1 = -k_1 \xi.$$

The sign «–» in the expression of force F_1 reflects the opposition of the force direction to the movement direction, and k_1 is the spring stiffness coefficient.

F_2 is the force of friction which is proportional to the coefficient k_2 ,to the movement velocity along the axis $\xi - \frac{d\xi}{dt}$ and is directed against the movement. Therefore

$$F_2 = -k_2 \frac{d\xi}{dt}.$$

Taking into consideration, that the acceleration $a = \frac{d^2\xi}{dt^2}$ we receive a resultant expression of the analytical model of a spring pendulum movement in the cylinder

$$m\frac{d^2\xi}{dt^2} = -k_2\frac{d\xi}{dt} - k_1\xi .$$

This model is often presented as:

$$\frac{d^2\xi}{dt^2} + 2r\frac{d\xi}{dt} + \omega^2\xi = 0, \qquad (2.1)$$

where $2r = \frac{k_2}{m}$; $\omega^2 = \frac{k_1}{m}$.

In the more generalized view model (2.1) is expressed as:

$$\frac{d^2\xi}{dt^2} = f\left(\xi, \dot{\xi}\right).$$

Analytical models in the form of equations

$$\frac{d^2\xi}{dt^2} = f(\xi); \quad \frac{d^2\xi}{dt^2} = f\left(\xi, \dot{\xi}\right),$$

right members of which are not explicitly dependent on time *t*, reflect the movement of stationary dynamic systems with the single degree of freedom.

The characteristics of the stationary dynamic systems are not explicitly dependent on time *t*. In particular, for the given example of the dynamic system, mass *m*, stiffness coefficient of spring k_1 and friction coefficient k_2 are the constants.

As a rule, the real dynamic systems are nonstationary and are described by analytical models in the form of differential equations right members of which are explicitly dependent on time *t*:

$$\frac{d^2\xi}{dt^2} = f\left(t, \xi, \dot{\xi}\right).$$

The characteristics of nonstationary dynamic systems are time functions. In the example under consideration $m = m(t); k_1 = k_1(t); k_2 = k_2(t)$. The variable-mass body is presented by the rocket. Its mass changes as fuel is consumed during the flight. The dependence $k_1 = k_1(t)$ means that eventually the spring stiffness changes. Obviously, the friction coefficient $k_2 = k_2(t)$ changes in time. The concept "degree of freedom" also has to be explained. The system possesses one degree of freedom if it can be presented it in the form of a mass point which moves along one of the space co-ordinates. If the point moves in a plain and its position is set by two co-ordinates ξ_1, ξ_2 the system possesses two degrees of freedoms. Further, if the point position is defined by three co-ordinates ξ_1, ξ_2, ξ_3 the system possesses three degree of freedoms. In case the system is presented as n mass points, which move in 3-dimensional space the system possesses 3n degrees of freedom.

For setting the position or, in a wider sense, the state of dynamic systems the concept of space is essential.

The space in which the relocation of moving points representing the system is described is called a configuration space. Its dimension is equal to the number of degrees of freedom of the dynamic system. The instant position (the state) of the moving points is called the configuration of the dynamic system at the given time point.

Mass behavior dynamics of the spring pendulum in the cylinder, considered as an example, is the system dynamics represented by one point in which all mass m is concentrated. This point moves along axis 0ξ , therefore the given system is the system with one degree of freedom. The position of such system is fully determined in one-dimensional configuration space with axis 0ξ. Adding to this axis the time axis and bringing each time point t in correspondence with value ξ we receive the well-known graph of function $\xi(t)$ which is shown in fig. 2.2.

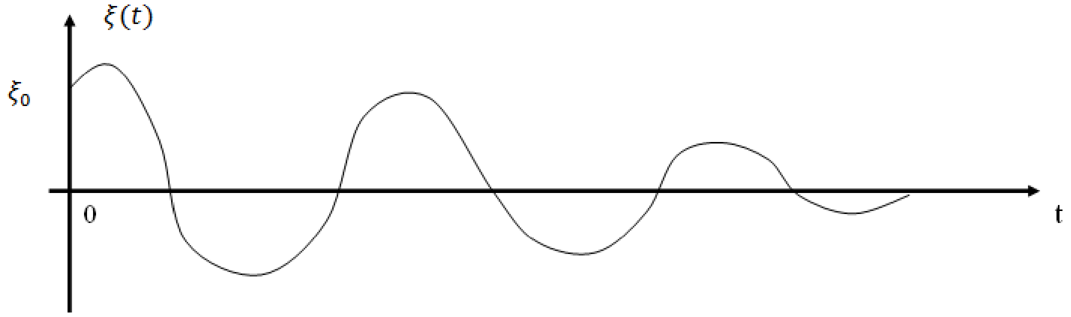


Figure 2.2 – Graph of function $\xi(t)$

Function $\xi(t)$ is the solution of differential equation (2.1) and reflects damped oscillation with the frequency ω. On the graph ξ_0 is initial deviation of mass m from the equilibrium position at the moment $t = 0$.

For studying the dynamic systems, besides the configuration space, the space state is often used. Most frequently the space state is used when studying the dynamic systems by methods of the automatic control theory [15].

In the space state each of the points $i \in \overline{1,n}$, which represent the investigated system, has two co-ordinates – the position co-ordinate (dimension) $z_{1i} = \xi$ and the repositioning velocity coordinate $z_{2i} = \dot{\xi}$.

When *n* points are moving in three-dimensional space, that is when the dynamic system has 3n degrees of freedom, the corresponding space state will be 6n-dimensional. The state vector of such system has a 6n components –

$$z = \left(z_{11}^{1}, z_{11}^{2}, z_{11}^{3}, z_{21}^{1}, z_{21}^{2}, z_{21}^{3}, \ldots, z_{1i}^{1}, z_{1i}^{2}, z_{1i}^{3}, z_{2i}^{1}, z_{2i}^{2}, z_{2i}^{3}, \ldots, z_{1n}^{1}, z_{1n}^{2}, z_{1n}^{3}, z_{2n}^{1}, z_{2n}^{2}, z_{2n}^{3}\right).$$

Components $z_{1i}^{1}, z_{1i}^{2}, z_{1i}^{3}$ $\left(i \in \overline{1,n}\right)$ define the position of the *i*-th point respectively by the first, second, third co-ordinate of three-dimensional space, and components $z_{2i}^{1}, z_{2i}^{2}, z_{2i}^{3}$ define the repositioning velocity of the *i*-th point accordingly by the first, second and third co-ordinate.

The whole set of vectors *z* or otherwise of the points in 6*n*-dimensional space for all values of time points of time $t \in [0, T]$ defines the dynamic system movement trajectory in the space state.

The ordered pair (t, z) is called the event or the phase of the system movement process, and the space which points are various events, – the phase space of the dynamic system.

The dynamic system trajectories in the space state form the system image in the space state or otherwise the phase portrait.

Let's make the model of the spring pendulum in fig. 1.5 in the space state and form its phase portrait.

The spring pendulum model in fig. 1.5 at its movement along axis 0*X* is the previously obtained ordinary differential equation of the second order, expressed as (discounting the friction):

$$\frac{d^2\xi}{dt^2} + \omega^2 \xi(t) = 0; \quad (k_2 = 0), \quad (\xi = x). \qquad (2.2)$$

The spring pendulum, provided that its mass is represented by the point moving along axis 0ξ, is the dynamic system with one degree of freedom. The position of such system is defined by coordinate ξ, and the repositioning velocity by the first derivative – $\frac{d\xi}{dt}$. Therefore

$$z_1 = \xi, \; z_2 = \frac{d\xi}{dt} = \dot{\xi}. \qquad (2.3)$$

Based on the introduced co-ordinates of the state space z_1 and z_2, model (2.2) transforms into the model in the form of the system of two ordinary differential equations of the first order:

$$\frac{dz_1}{dt} = z_2,$$
$$\frac{dz_2}{dt} = -\omega^2 z_1. \tag{2.4}$$

The received model (2.4) is the model of the spring pendulum in state space.

To form the phase portrait of the spring pendulum it is necessary to determine $z_1(t)$ and $z_2(t)$. For this purpose it is necessary to solve the system (2.4) or the equation (2.2). The solution of the equation (2.2) is

$$\xi(t) = \xi_0 \cos \omega t + \frac{\dot{\xi}_0}{\omega} \sin \omega t,$$

where $\xi_0 = \xi(t_0)$, $\dot{\xi}_0 = \frac{d\xi}{dt}\Big|_{t=t_0}$ are the initial conditions for solving (2.2).

Taking into consideration the designation of (2.3), we obtain:

$$z_1(t) = \xi(t) = \xi_0 \cos \omega t + \frac{\dot{\xi}_0}{\omega} \sin \omega t;$$
$$z_2(t) = \frac{d\xi}{dt} = -\xi_0 \omega \sin \omega t + \dot{\xi}_0 \cos \omega t. \tag{2.5}$$

Having squared the left and the right members of correlations (2.5), and then having added them at $\omega = 1$, we obtain:

$$z_1^2(t) + z_2^2(t) = \xi_0^2 + \dot{\xi}_0^2. \tag{2.6}$$

It is obvious, that (2.6) is the equation of the circle with the radius $r_0 = \sqrt{\xi_0^2 + \dot{\xi}_0^2}$. Changing the values of initial conditions ξ_0, ξ_0, we receive a set of circles. This set is the phase portrait of the spring pendulum which is presented in fig. 2.3.

Fig. 2.3 shows the point $z = (z_1, z_2)$ which, at increasing t> 0 moves on the circle clockwise with the angular velocity ω=1. The origin of co-ordinates corresponds to the state of rest and is the degenerate circle at initial conditions $\xi_0 = 0, \dot{\xi}_0 = 0$.

The phase portrait permits to determine the properties of the investigated system. Indeed, in the system considered $z_1^2(t)$ is proportional to the potential energy, and $z_2^2(t)$ to the kinetic energy. The sum $z_1^2(t) + z_2^2(t)$ is obviously proportional to the total energy of the system, which in this case is constant. In the process of movement one type of energy increases, and the other decreases so, that their sum is constant.

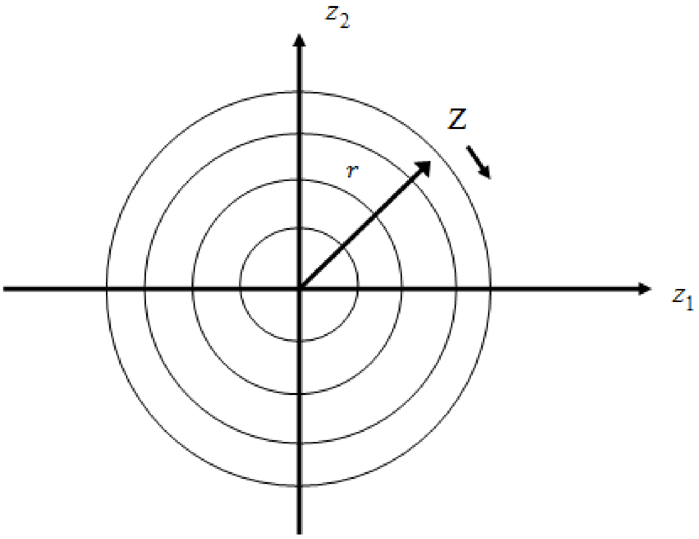


Figure 2.3 – The spring pendulum phase portrait

Another example of the modeling object can be another object of a different physical nature – the electrical oscillatory circuit (*LC*-chain) with an external voltage source which is shown in fig. 2.4.

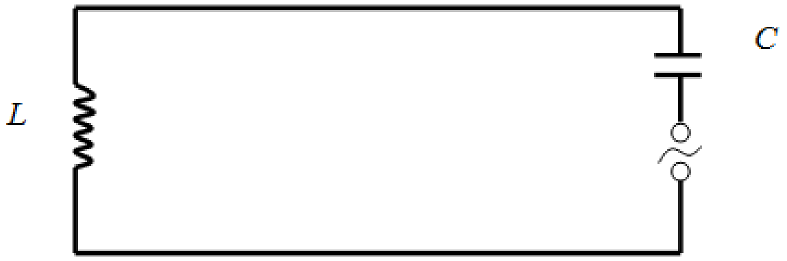


Figure 2.4 – *LC*-chain schema with an external voltage source

The processes in the object of interest are classified as the continuous-determined processes. Therefore, the mathematical scheme for the analytical model construction can use the differential equations.

Based on the electric nature of the processes and the Kirchhoff's laws the processes are subject to, we obtain the following analytical model of the processes dynamics in the form of the system of ordinary differential equations:

$$C\frac{dV_C}{dt} = -i_L;$$

$$L\frac{di_L}{dt} = -V_C + V_\sim,$$

where V_C is the voltage on the capacitor with capacity C, i_L is current in the coil L, $V_\sim$ is voltage of the external power source.

If to use designation for presenting the modeling results in the state space – $z_1 = V_C$, $z_2 = i_L$, $x(t) = \frac{V_\sim}{L}$ the received model takes the following form:

$$\begin{aligned} \frac{dz_1}{dt} &= -\frac{1}{C} z_2; \\ \frac{dz_2}{dt} &= -\frac{1}{L} z_1 + x(t). \end{aligned} \qquad (2.5)$$

If to exclude the impact of environment on the LC-chain, that is to suppose that $x(t) = 0$ it becomes obvious that, accurate within certain designations, model (2.4) describing the dynamics of the spring pendulum, and model (2.5) describing the dynamics of an electric oscillating circuit, coincide. Hence, we can make an important conclusion – despite different physical nature of the processes in the objects under study, the dynamics of their behavior is identical. This fact is used for deeper studying and understanding of investigated processes. Thus, the dynamics of well-known processes in the hydrodynamic systems can be used for understanding the dynamics of processes in the bank system.

Now let us consider the case when $x(t) \neq 0$, that is the LC-chain is exposed to the external environment. Providing that $x(t) = u \sin pt$, the solution of system (2.5) is

$$z_1 = V_C(t) = V_0 \cos \omega t + \frac{i_0}{\omega} \sin \omega t + \frac{up}{\omega(\omega^2 - p^2)} \sin \omega t + \frac{u}{\omega(\omega^2 - p^2)} \sin pt, \quad (2.6)$$

where $\omega = \frac{1}{\sqrt{LC}}$.

The solution of (2.6) contain members which depend on parameters of the external impact u, p. It is obvious, that these parameters can be used for modification of solution (2.6), and therefore, in a wider sense, for modification of the state of the object of interest.

Function $x(t)$ in the right part of the differential equation is called the input process, and its instantaneous value at $t = t'$ is called the input signal $x_{t'} = x(t')$.

Along with the input process, the output processes $y(t)$ and the output signals y_t are considered and they are defined as

$$y_t = g(t, z, x(t)).$$

In general case the input and the output processes are the vectors of the following kind

$$x(t) = (x_1(t), x_2(t), \ldots, x_m(t));$$
$$y(t) = (y_1(t), y_2(t), \ldots, y_r(t)).$$

The analytical model of the dynamics of the system processes is set by the following groups of correlations:

a) differential equations of motion in the state space:

$$\begin{aligned}
\frac{dz_1}{dt} &= f_1(t, z_1, z_2, \ldots, z_n, x_1, x_2, \ldots, x_m);\\
\frac{dz_2}{dt} &= f_2(t, z_1, z_2, \ldots, z_n, x_1, x_2, \ldots, x_m);\\
&\ldots\\
\frac{dz_n}{dt} &= f_n(t, z_1, z_2, \ldots, z_n, x_1, x_2, \ldots, x_m);
\end{aligned} \tag{2.7}$$

b) initial conditions: at $t = t_{0,}\ z_1 = z_1^0,\ z_2 = z_2^0, \ldots, z_n = z_n^0;$

c) values of the input process on the half-interval $(t_0, t]$:

$$x_1(t)\big|_{t_0}^{t},\ x_2(t)\big|_{t_0}^{t}, \ldots, x_m(t)\big|_{t_0}^{t},$$

which are also called fragments of the input process,

d) correlations for the output processes:

$$\begin{aligned}
y_1(t) &= g_1(t, z_1, z_2, \ldots, z_n, x_1, x_2, \ldots, x_m);\\
y_2(t) &= g_2(t, z_1, z_2, \ldots, z_n, x_1, x_2, \ldots, x_m);\\
&\ldots\\
y_r(t) &= g_r(t, z_1, z_2, \ldots, z_n, x_1, x_2, \ldots, x_m).
\end{aligned}$$

If the conditions of existence and uniqueness of solutions of the specified system of ordinary differential equations (2.7) are met, these solutions can be presented as follows:

$$\begin{aligned}
z_1(t) &= \varphi_1\left(t, t_0, z^0, x(t)\big|_{t_0}^{t}\right),\\
z_2(t) &= \varphi_2\left(t, t_0, z^0, x(t)\big|_{t_0}^{t}\right),\\
&\ldots\\
z_n(t) &= \varphi_n\left(t, t_0, z^0, x(t)\big|_{t_0}^{t}\right),
\end{aligned}$$

where $z^0 = \left(z_1^0, z_2^0, ..., z_n^0\right)$ is the vector of initial conditions, $x(t)\Big|_{t_0}^{t} = \left(x_1(t)\Big|_{t_0}^{t}, x_2(t)\Big|_{t_0}^{t}, ..., x_m(t)\Big|_{t_0}^{t}\right)$ is the vector of values of the input process on the half-interval $(t_0, t]$.

In terms of vectors the solution can be presented as

$$z(t) = \Phi\left(t, t_0, z^0, x(t)\Big|_{t_0}^{t}\right).$$

Such representation is also called the next-state function of the dynamic system, and the function $y(t) = G\left(t, t_0, z^0, x(t)\Big|_{t_0}^{t}\right)$ is called an output function of the dynamic system.

Besides the specified general view of the analytical model of the dynamic system as the system of ordinary differential equations, the model in the form of ordinary differential equations or their systems with a retarded argument is frequently used. Such type of the model can result from the following problem of one-lane traffic control.

Let *x* be direction of the one-lane traffic flow, as shown in fig. 2.4.

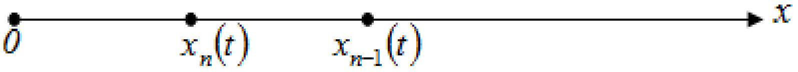


Figure 2.4 – The scheme of one-lane traffic flow

Let's designate the position of the n-th transport vehicle (TV) at the instant *t*, as $x_n(t)$ and the position of the (*n*-1) TV which moves directly before the *n*-th as $x_{n-1}(t)$.

The travel speeds of TVs n and (*n*-1) are defined as $\frac{dx_n(t)}{dt}, \frac{dx_{n-1}(t)}{dt}$.

It is obvious, that the driver of the *n*-th TV with the delay τ reacts to the change of the speed of the TV which moves before it, that is of the (*n*-1) TV. The acceleration of the n-th TV is presented by the following expression

$$\frac{d^2 x_n(t+\tau)}{dt^2} = \alpha\left(\frac{dx_{n-1}(t)}{dt} - \frac{dx_n(t)}{dt}\right), \tag{2.8}$$

where α is the sensitivity factor which is assumed to be a constant with the dimension of 1/second

Designating $t + \tau = \xi$, we obtain $t = \xi - \tau, dt = d\xi$, as τ is a constant magnitude. Taking into account these designations, the equation (2.8) takes the form:

$$\frac{d^2x_n(\xi)}{d\xi^2}=\alpha\left(\frac{dx_{n-1}(\xi-\tau)}{d\xi}-\frac{dx_n(\xi-\tau)}{d\xi}\right).$$

Reverting to the usual designation of time $t=\xi$, we obtain the final variant of the analytical model of the TV dynamics, moving in one lane:

$$\frac{d^2x_n(t)}{dt^2}=\alpha\left(\frac{dx_{n-1}(t-\tau)}{dt}-\frac{dx_n(t-\tau)}{dt}\right). \tag{2.9}$$

Model (2.9) is the ordinary second-order differential equation with the retarded argument $(t-\tau)$. Model (2.9) can be transformed into the model in the state space by introducing the following notations:

$$x_n(t)=z_1;\quad \frac{dx_n}{dt}=z_2;\quad \frac{dx_{n-1}(t)}{dt}=x(t).$$

The first two notations are usual for the state space, and notation of speed of the (*n*-1) TV as an input process means, that (*n*-1)-th TV is viewed as the “external environment” for the *n*-th TV, which change of the state is an input process for the considered *n*-th TV as the system under study.

Taking into account the introduced notations, the model of dynamics of TVs, moving in the single lane, looks as follows:

$$\frac{dz_1}{dt}=z_2;$$
$$\frac{dz_2}{dt}=\alpha[x(t-\tau)-z_2(t-\tau)].$$

The elementary differential equation with a retarded argument is expressed as:

$$\frac{dz(t)}{dt}=f[t,z(t),z(t-\tau)].$$

The solution of this equation is expressed as

$$z(t)=z_0+\int_{t_0}^{t}f(\Theta,z(\Theta),z(\Theta-\tau))d\Theta,$$

that is for obtaining the solution, apart from the initial conditions z_0, it is necessary and sufficient to specify the function $z(t)$ in the half-interval $[t_0-\tau,t_0)$ or otherwise to specify the initial function.

Analytical models in the form of differential equations with a retarded argument are used for describing dynamic systems with an aftereffect, that is for the systems, which for determining of their state $z(t)$ at $t>t_0$, apart from specifying the initial state $z_0 = z(t_0)$ require specification of the previous history of the state which is determined by the initial function $z(t)\big|_{t_0-\tau}^{t_0}$, that is by function $z(t)$ in the half-interval $[t_0 - \tau, t_0)$.

2.3. Models in the form of finite-difference equations

Finite-difference equations can be used as the mathematical scheme for developing analytical models either of continuous-determined or discrete-determined processes. Indeed, on the one hand, the finite-difference equations are transformations of differential equations which, as shown in the previous subsection, describe the dynamics of the continuous-determined processes. The transformation is based on replacement of derivatives of the following form:

$$\frac{dz_i}{dt} \approx \frac{z_i(t+h) - z_i(t)}{h},$$

where h is a sufficiently small quantity which in the numerical methods is considered as the integration step.

The specified transformation enables to present the system of differential equations (2.7) in the form of the following system of finite-difference equations:

$$\begin{aligned} & z_1(t+h) = z_1(t) + hf_1(t, z_1, z_2, \dots, z_n, x_1, x_2, \dots, x_m); \\ & z_2(t+h) = z_2(t) + hf_2(t, z_1, z_2, \dots, z_n, x_1, x_2, \dots, x_m); \\ & \dots \\ & z_n(t+h) = z_n(t) + hf_n(t, z_1, z_2, \dots, z_n, x_1, x_2, \dots, x_m). \end{aligned} \tag{2.10}$$

The system of finite-difference equations can be considered as the analytical model of the continuous-determined processes which state is defined by the system of differential equations (2.7).

On the other hand, the finite-difference equations are the independent mathematical scheme as the basis for developing analytical models of the discrete-determined processes. An example of such model is the model for solving the macroeconomic problem of relations «expenditures - output» within the framework of corporation of enterprises [16].

Let z_i be the total yield of the i-th product per the time unit; a_{ij} the amount of the i-th product consumed for manufacturing of a unit of the j-th product. Then the model of manufacture self-sufficiency within the framework of the corporation is expressed as:

$$\forall j \in \overline{1,n}; \quad z_j(t+1) = \sum_{i \in \overline{1,n}} a_{ij} z_i(t); \quad z(0) = z_0. \qquad (2.11)$$

It is obvious, that (2.11) is the system of the finite-difference equations of the type (2.10) on the assumption of the linear character of function $f_i \left(i \in \overline{1,n}\right)$ h = 1. Otherwise (2.11) is called a one-step transition function of the dynamic system described by the finite-difference equations.

2.4. Discrete-determined modeling

The basic mathematical schemes for developing the analytical models of the discrete-determined processes, along with the finite-difference equations mentioned in the previous subsection, can also be various types of finite automata.

One of the basic concepts of the theory of finite automata is the concept of the alphabet as the final set of letters, each of which designates a certain object. Objects are represented as signs, figures, letters, drawings, phrases, algorithms etc.

The finite ordered set of letters in the given alphabet is called a word.

The finite Mealy or Moore automaton ensures transformation of the signals of the input alphabet *X* into the signals of the output alphabet *Y*.

The Mealy finite automaton functions at the moment of time $t_0 < t_1 < t_2 ...$ in such a way that at each moment $t_j \in T$ (T is the set of the moments of time) it is in one of the possible states $z_j = z(t_j) \in Z$ (Z is the final set of states) and after arrival of the input signal $x \in X$ at the moment of time $t_j \in T$, starting from the moment t_1, the automaton passes into the next state from the set *Z* and produces the output signal $y \in Y$ [17, 18].

Three basic forms of the automaton representation are used – analytical, tabular and graphic. All these forms ensure the transformation as a result of which the new state of the automaton and the output signal can be defined.

The analytical form of representation of the Mealy finite automaton presupposes representation of the one-step next-state function φ for defining the new state of the finite automaton, and the output function ψ for defining the output signal:

$$\begin{aligned} z(t) &= \varphi[z(t-1), x(t)]; \\ y(t) &= \psi[z(t-1), x(t)]. \end{aligned} \qquad (2.11)$$

From (2.11) it follows, that for defining the new state of the automaton $z(t)$ and the output signal $y(t)$ at the moment of time $t \in T$ it is necessary to know the state of the automaton at the previous moment $z(t-1)$ and the input signal at the moment $t \in T$ – $x(t)$.

The tabular form of representation of the Mealy finite automaton presupposes representation of two tables – the transition table which corresponds to the function φ, and the output table which corresponds to the function ψ in (2.11).

Let, for example, the input alphabet *X*, the set of states *Z*, the output alphabet *Y* have the form:

$$X = \{x_1, x_2, x_3\}, \; Z = \{z_1, z_2, z_3, z_4\}, \; Y = \{y_1, y_2\}.$$

Then the transition table "φ" and the output table "ψ" can be presented according to tab. 2.1 and tab. 2.2. The structure of the tables is identical – their lines correspond to the values of input signal $x(t)$ at the moments $t \in T$, and columns correspond to the states of the automaton at the moment of time $z(t-1)$ prior to (*t*–1).

The states of the automaton in which it transfers at the moment $t \in T$, that is $z(t)$, are recorded at the intersection of lines and columns of tab. 2.1. Thus, for example, if the automaton at the moment $t-1 \in T$ was in the state $z(t-1) = z_1$ (the first column), and at the moment $t \in T$ the input signal $x(t) = x_3$ arrived (the third line), then at the moment $t \in T$ the automaton would pass into the new state $z(t) = z_2$ defined by the intersection of the first column and the third line.

Table 2.1 – Table of transitions "φ"

$x(t)$	$z(t-1)$			
	z_1	z_2	z_3	z_4
x_1	z_2	z_3	z_1	z_2
x_2	z_1	z_1	z_4	z_4
x_3	z_2	z_3	z_2	z_1

Table 2.2 – Table of outputs "ψ"

$x(t)$	$z(t-1)$			
	z_1	z_2	z_3	z_4
x_1	y_1	y_2	y_1	y_2
x_2	y_2	y_1	y_2	y_1
x_3	y_1	y_1	y_2	y_2

At the intersection of lines and columns of tab. 2.2 the values of the output signal $y(t) \in Y$, which are produced at the moment $t \in T$ are recorded. Thus, if the automaton at the moment (t-1) was in the state $z(t-1)=z_1$, and at the moment t the input signal $x(t)=x_3$ arrived, then the automaton produces the output signal $y(t)=y_1$ which is set in the table by intersection of the first column and the third line.

The representation of the Mealy finite automaton in the form of the oriented graph presupposes representation of the states of the automaton $z_j = z(t_j) \in Z$ by the tops of the graph and representation of transitions from the state $z_i = z(t-1)$ into the state $z_j = z(t)$ by the oriented arcs (arrows) (z_i, z_j). At the same time usually at the beginning of an arrow the input signal $x(t)$ is written, which causes the transaction, and at the end of an arrow the output signal $y(t)$ is written, which accompanies the transition.

For the considered example of the Mealy automaton, the transitions and the output signals being presented in tab. 2.1 and tab. 2.2, the oriented graph has the form given on figure 2.5.

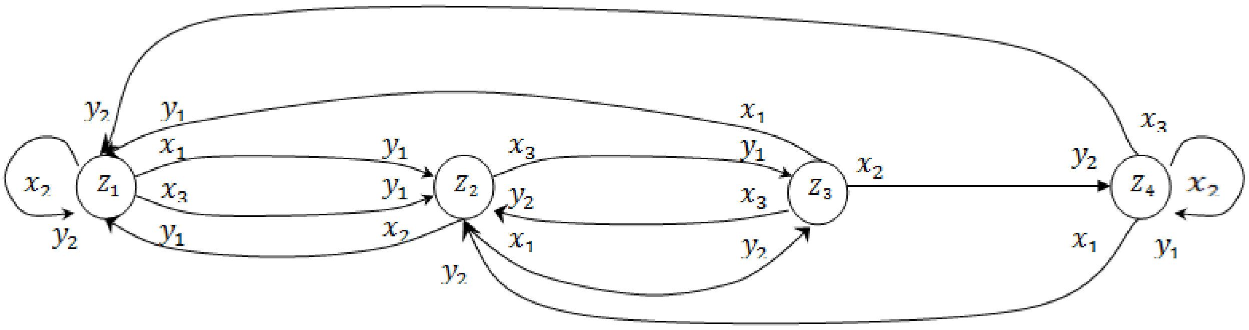


Figure 2.5 – The Mealy automaton oriented graph

Another type of the finite automata, often used for modeling is the Moore's automaton. This automaton functions similarly to the Mealy automaton, has the same forms of representation and differs only in the output function which is defined as

$$y(t) = \psi'[z(t)],$$

where ψ' is called the shifted function of outputs.

In a shorter way the Mealy automaton is often designated as $A(X, Z, Y, \varphi, \psi)$, and the Moore automaton as $A'(X, Z, Y, \varphi, \psi')$.

The finite automaton is synchronous if the arrival times of the input signal $t_1, t_2 ...$ are previously fixed and satisfy the correlation $t_1 < t_2 < ...$.

The finite automaton is asynchronous if the arrival times of the input signal $t_1, t_2 ...$ are not previously fixed, and only satisfy the correlation $t_1 < t_2 < ...$ Since in this case the specified moments can be sufficiently close to each other, the asynchronous automaton can be used as the mathematical scheme for developing the analytical models of the dynamic systems functioning in the continuous time. Let's consider some examples of analytical models based on the finite automata.

Example 1. It is required to develop the model of the ticket machine functioning. The device accepts tokens or coins of denominations 1,2,3,5 units and delivers tickets in the value of 5 units. Having delivered the ticket the device nullifies its state and gives no change.

In order to make the model based on the Mealy or Moore finite automaton it is necessary to determine the set of the input signals *X*, the set of the states *Z*, the set of the output signals *Y*, and also, in one or another form, functions φ and $\psi\,(\psi'$ for the Moore automaton).

The elements of the set *X* in the object of interest are tokens or coins of different denomination, which buyers can use for purchasing tickets. Therefore it is obvious that $X = \{1,2,3,5\}$.

The possible states of the device can be: «0» when there are no coins in the device; «1» when there is one coin of denomination 1 in the device; «2», when there is one coin of denomination 2, or two coins of denomination 1 each in the device; «3», when in the device there is one coin of denomination 3 or two coins, one of which is of denomination 1, and the other is of denomination 2; «4», when in the device there are two coins of denomination 1 and one of denomination 2 or two coins of denomination 2, or, finally, two coins, one of which is of denomination 1, and the other is of denomination 3.

If in the device there are coins of the total denomination of 5 and more, then, according to the conditions of the device functioning, it delivers a ticket to the buyer and nullifies the state.

Thus, the set of possible states of the device looks like $Z = \{0,1,2,3,4\}$. The elements of the set of the output signals in the object of interest can be «ticket is to be delivered» and «ticket is not to be delivered». If element «ticket is to be delivered» is brought in correspondence with «1», and the element «the ticket is not to be delivered» with «0», the set *Y* takes the form $Y = \{0,1\}$.

It is recommended to start determination of functions φ and ψ from the tabular form of their representation. The tables of transitions "φ" and outputs "ψ" for the model under development have the form of tab. 2.3 and tab. 2.4 accordingly.

Table 2.3 – Table of transitions "φ" in the model of functioning of the ticket selling devices

x	$z(t-1)$				
	0	1	2	3	4
1	1	2	3	4	0
2	2	3	4	0	0
3	3	4	0	0	0
5	0	0	0	0	0

In the cells on the intersection of rows and columns of tab. 2.3 the elements of the set Z are recorded as a result of summation of values $z(t-1)$ and $x(t)$, providing that the result is not greater than 4. If the result is greater than 4, then the value of $z(t)$ is assumed to be equal to zero. Hence the analytical form of the function φ representation looks as:

$$z(t)=\begin{cases} z(t-1)+x(t), \text{ if } z(t-1)+x(t)\le 4; \\ 0, \qquad\qquad\quad \text{ if } z(t-1)+x(t)>4. \end{cases} \tag{2.12}$$

Table 2.4 – Table of outputs "ψ"in the model of functioning of the ticket selling devices

x	$z(t-1)$				
	0	1	2	3	4
1	0	0	0	0	1
2	0	0	0	1	1
3	0	0	1	1	1
5	1	1	1	1	1

In the cells of tab. 2.4 the elements of the set Y are recorded. Thus, it is obvious that y= 0, when the state of the device is defined by values within the range from 1 to 4, and y = 1 when the state of the device is defined by values 5 and greater, to which correspond zeros in tab. 2.3. Thus, the analytical form of the function ψ representation looks as:

$$y(t)=\begin{cases} 0, \text{ if } z(t-1)+x(t)\le 4; \\ 1, \text{ if } z(t-1)+x(t)>4. \end{cases} \tag{2.13}$$

The graph form of the Mealy automaton for the modeling object of interest is presented on fig. 2.6.

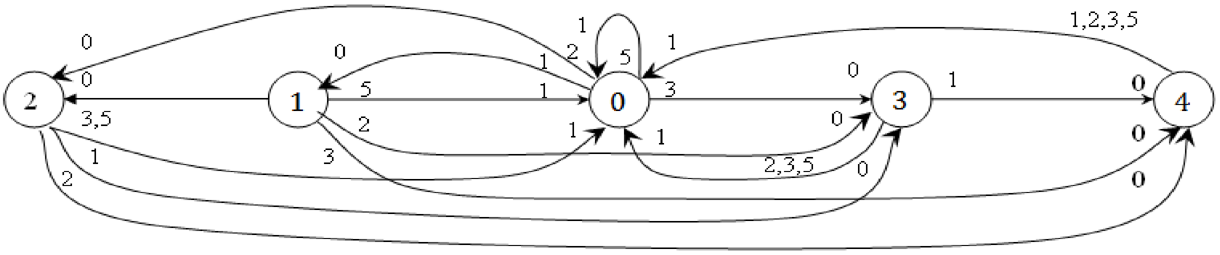


Figure 2.6 – The Mealy automaton graph, modeling the functioning of the ticket selling device

Thus, the model of the functioning process of the ticket machine based on the Mealy automaton includes the specified sets *X*, *Y*, *Z*, as well as functions φ and ψ, set either in the form of tab. 2.3. tab. 2.4, or in the form of correlations (2.12), (2.13), or, finally, in the form of graph in fig. 2.6.

Example 2. It is necessary to develop the model of functioning of the lift in the three-story house.

Like in the previous example at the beginning the set of the input signals *X*, the set of states *Z* and the set of the output signals *Y* are to be determined.

The elements of the set *X* are signals $x_i\left(i\in\overline{1,3}\right)$ of calling the lift to the *i*-th floor, that is $X=\left\{x_i\mid i\in\overline{1,3}\right\}$.

The set $Z=\left\{z_i\mid i\in\overline{1,3}\right\}$, where z_i corresponds to the state «the lift is on the *i*-th floor».

Set *Y* can consist of the elements which correspond to the direction of the lift movement. For example, $Y=\left\{y_i\mid i\in\overline{1,3}\right\}$, where y_1 - «the lift is moving upwards», y_2 - «the lift stands still», y_3 - «the lift is moving downwards».

The tabular form of functions $\varphi\, and\, \psi$ is represented in tab. 2.5, 2.6.

Table 2.5 – Table of “φ” transition in the lift functioning model

x	$z(t-1)$		
	z_1	z_2	z_3
x_1	z_1	z_1	z_1
x_2	z_2	z_2	z_2
x_3	z_3	z_3	z_3

Table 2.6 – Table of outputs “ψ” in the lift functioning model

x	$z(t-1)$		
	z_1	z_2	z_3
x_1	y_2	y_3	y_3
x_2	y_1	y_2	y_3
x_3	y_1	y_1	y_2

Values $z(t)$ in table 2.5 are such that they meet the requirements of the user, that is signals $x(t)$, irrespective of the fact where the lift was situated in the previous $(t-1)$ moment.

Values $y(t)$ in table 2.6 are determined by comparison of values $x(t)$ and $z(t-1)$, that is the call signal at the given instant and the lift state at the previous instant. For example, if at the moment $(t-1)$ the lift is in the state z_1 (on the ground floor), and at the moment t arrives the input signal $x(t)=x_1$ (call to the ground floor), it is obvious that the output signal will be $y(t)=y_2$«the lift stands still».

Analytically functions φ and ψ are defined by expressions:

$$z(t)=x(t)\ ; \tag{2.14}$$

$$y(t)=\begin{cases} y_1,\text{ if } x_i > z_i; \\ y_2,\text{ if } x_i = z_i; & \left(i\in\overline{1,3}\right) \\ y_3,\text{ if } x_i < z_i; \end{cases} \tag{2.15}$$

The graph form of φ and ψ representation is given in fig. 2.7.

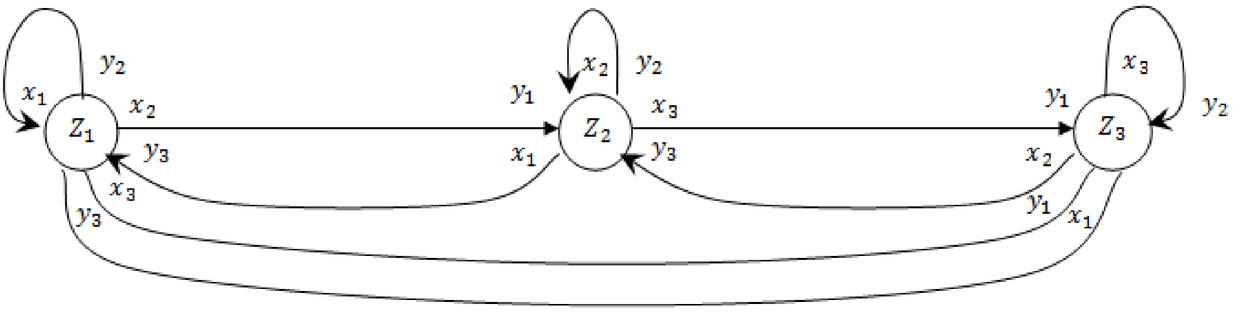


Figure 2.7 – The Mealy automaton graph, modeling the lift functioning

Besides the specified Mealy and Moore finite automatons, finite automatons with an aftereffect and nonstationary finite automatons are often used for constructing analytical models of discrete-determined processes

The finite automaton with an aftereffect while constructing discrete-determined models enables to take into account the previous history of the investigated process and in this sense it is an analogue of differential equations with retarded argument which are used for constructing continuous-determined models.

The finite automaton with an aftereffect is defined as

$$\begin{aligned} z(t) &= \varphi[z(t-k), z(t-k+1),..., z(t-1), x(t)]; \\ y(t) &= \psi[z(t-1), x(t)], \end{aligned} \qquad (2.16)$$

where k – a natural number called the order of the initial set.

The set of states $z(t-k), z(t-k+1),..., z(t-1)$ is called an automaton prehistory, and the set of moments $t-k, t-k+1, ..., t-1$ is called the initial set relative to the moment $t-1$.

It is obvious, that with $k = 1$ the automaton (2.16) turns into a usual Mealy finite automaton.

The aftereffect automaton as a mathematical scheme is used for developing models of systems for which the delay factor in the course of their functioning is typical. For example, the return on investment does not appear immediately, but after some period of time, which length is taken into account either by value τ in differential equations with a retarded argument, or by value k in the studied mathematical schema of the aftereffect automaton.

The nonstationary automaton enables to consider dependence of the environment changes and of the structure and parameters of the modeled system on time. Formally this dependence is expressed in the form of the explicit dependence of functions φ and ψ on time. Thus, the nonstationary automaton is defined as

$$\begin{aligned} z(t) &= \varphi[t-1, z(t-1), x(t)]; \\ y(t) &= \psi[t-1, z(t-1), x(t)]. \end{aligned} \qquad (2.17)$$

If we consider time t as an additional co-ordinate of the automaton state, that is to introduce a new designation of the state as $\tilde{z}(t) = (t, z(t))$, then in the correlations it is possible to get rid of the explicit dependence of the transition function φ and the output function ψ on time t.

Indeed, assuming, that $\tilde{z}(t-1) = (t-1, z(t-1))$, we obtain

$$\begin{aligned} \tilde{z}(t) &= \varphi[\tilde{z}(t-1), x(t)]; \\ \tilde{y}(t) &= \psi[\tilde{z}(t-1), x(t)]. \end{aligned} \qquad (2.18)$$

By its form the obtained automaton (2.18) is the stationary, but not no longer the finite one, as there are as many states of $\tilde{z}(t-1)$ type, as the number of moments $t-1$. However, in modeling practice often the finite set of moments $\{t\}$ is taken on modeling interval $[0,T]$, that is why the functioning of the automaton

(2.18) will be similar to the functioning of a usual Mealy or Moore finite automaton.

2.5. Discrete-stochastic modeling

Functioning of the stochastic dynamic systems in discrete time is often described by means of the probabilistic automata.

The concept of the probabilistic automata is based on the unity of three concepts – the random transition automaton, the automaton with a random choice of the initial state and the automaton with a random choice of the output signal.

The random transition automaton or with the random transition function φ is defined by the population of matrixes $\{\|P_{ij}(x)\|\}$ the elements of which $P_{ij}(x)=P\{z(t)=z_j / z(t-1)=z_i; x(t)=x\}$ represent the conditional probabilities that at the moment t the automaton will be in the state z_j, that is $z(t)=z_j$, provided that during the previous moment it was in the state z_i that is $z(t-1)=z_i$ and at the moment t the external signal $x(t)=x$ arrived.

The random transition automaton functions as follows. Let at the moment t arrives the input signal$x(t)=\hat{x}$. According to value $\hat{x}$ the matrix $\{\|P_{ij}(x)\|\}$ is selected from the population of matrixes$\{\|P_{ij}(x)\|\}$ $\|P_{ij}(\hat{x})\|$.

If in the previous *t*-1 moment the automaton was in the state $z(t-1)=z_i$, then in the matrix $\|P_{ij}(\hat{x})\|$ the i-th line with the values of conditional probabilities p_{ij} is chosen . This line looks like

$$p_{i1}\ p_{i2}\ \cdots\ p_{ij}\ \cdots\ p_{in},$$

and its elements satisfy the conditions:

$$\forall i,j \quad 0 \le p_{ij} \le 1, \quad \sum_{j\in\overline{1,n}} p_{ij} = 1.$$

The specified conditions ensure the possibility of unconditional transition from the state z_i into one of the states z_j.

To choose the next state $z(t)=z_j$ the toss procedure is applied [19]. According to this procedure, from the sequence of random, regularly distributed in the range of (0,1) numbers $\{\xi_k \mid k \in 1,2,...\}$ one of $\overline{\xi}_k$ numbers is chosen in a random way.

If $0<\overline{\xi}_k \le p_{i1}$, then $z(t)=z_1$

If $p_{i1} < \bar{\xi}_k \le p_{i1} + p_{i2}$, then $z(t) = z_2$. And so on.

In general, if $\sum\limits_{\nu \in \overline{1, j-1}} p_{i\nu} < \bar{\xi}_k \le \sum\limits_{\nu \in \overline{1, j}} p_{i\nu}$, then $z(t) = z_j$.

Since $\bar{\xi}_k \in (0,1)$, the number of states n is finite and $\sum\limits_{j \in \overline{1,n}} p_{ij} = 1$, then the specified procedure necessarily ensures the choice of the specific state z_j.

At the next step of the automaton functioning the state z_j will be considered as the previous one, that is relating to the moment $(t-1)$ – $z(t-1) = z_j$. To define the next state $z(t)$ it is necessary to repeat all the described step of the automaton functioning.

It is possible to show, that multiple realization of the random transition automaton functioning leads to the condition when the frequencies of choice of states $z_1, z_2, \ldots z_n$, provided that $z(t-1) = z_i$ will be close to the corresponding probabilities $p_{i1}\ p_{i2} \ldots p_{in}$.

The realization of each step of the random transition automaton is predetermined by knowledge of the state at the previous moment. It is obvious, that for the first step realization it is necessary to have knowledge of the initial state or the automaton state at the moment $t = 0$, that is $z(0) = z_0$.

If the state $z(0)$ is not fixed and is chosen on a toss based on distribution of probabilities $p_0 = (p_{01}, p_{02}, \ldots, p_{0n})$, where $0 \le p_{0i} \le 1 \left(i \in \overline{1,n}\right)$ is the probability that the initial state is z_i and $\sum\limits_{i \in \overline{1,n}} p_{0i} = 1$, then such stochastic system is called the automaton with a random choice of the initial condition.

Finally, the automaton with a random choice of the output signal or the automaton with a random function of the outputs ψ is defined by the population of matrixes $\{\|Q_{ir}(x)\|\}$ the elements of which

$$q_{ip}(x) = P\{y(t) = y_p \,/\, z(t-1) = z_i; x(t) = x\}$$

represent conditional probabilities of producing by the automaton the output signal $y(t) = y_p$ at the moment t provided that $z(t-1) = z_i,\ x(t) = x$.

The automaton with a random choice of the output signal functions similarly to the random transition automaton.

On the whole, the probabilistic automaton is the object

$$A\left(X, Z, Y, \left\{M\left(y/_x\right)\right\}\right),$$

where $X = \left\{x_r \mid r \in \overline{1,l}\right\}$ is the input alphabet (the set of input signals); $Z = \left\{z_i \mid i \in \overline{1,n}\right\}$ is the set of states; $Y = \left\{y_p \mid p \in \overline{1,m}\right\}$ is the output alphabet (the set of output signals); $\{M(y/x)\}$ is the set of matrixes $l\times m$ each having dimension $n\times n$.

In the matrix $M(y_p/x_r)$ the element $\mu_{ij}(y_p/x_r)$ is defined as the conditional probability

$$\mu_{ij}(y_p/x_r)=P\{z(t)=z_j,y(t)=y_p/z(t-1)=z_i,x(t)=x_r\},$$

that is, as the probability that the automaton will pass into the state $z(t)=z_j$, produce the output signal $y(t)=y_p$, provided that at the previous moment (t-1) the automaton was in the state $z(t-1)=z_i$, and at the moment t the signal $x(t)=x_r$ arrived at the input.

If the elements $\mu_{ij}(y_p/x_r)$ possess the values of "zero" or "one", the probability automaton turns into the previously described finite automaton. It will be the Mealy finite automaton, provided that

$$P\{z(t)=z_j,y(t)=y_p/z(t-1)=z_i,x(t)=x_r\}=P\{z(t)=z_j/z(t-1)=z_i,x(t)=x_r\}\times$$
$$\times P\{y(t)=y_p/z(t-1)=z_i,x(t)=x_r\},$$

that is random quantities z_j,y_p are conditionally independent.

If

$$P\{y(t)=y_p/z(t-1)=z_i,x(t)=x_r\}=P\{y(t)=y_p/z(t)=z_j\}$$

everywhere, where $P\{z(t)=z_j/z(t-1)=z_i,x(t)=x_r\}\neq 0$, the finite automaton is the Moore automaton.

The example of the functioning model of the discrete-stochastic system on the basis of the random transition automaton is the functioning model of a trade enterprise selling piece-goods which initial volume is known in advance [20].

The trade enterprise functioning in many respects is connected with the turnover control which is realized on the basis of the function of the temporal sales volume change. The construction of this function is possible by performing operations with the discrete-stochastic model which describes the processes of buyers' appearing and purchasing piece-goods by them, that leads to a certain change of goods volume in the trade enterprise stock.

The model on the basis of the random transition automaton is constructed based on the basis of the following preconditions and assumptions.

The trade enterprise sells piece-goods of *n* descriptions. Its state at every discrete moment $t\in\{0,1,2,...\}$ is defined by the population of goods volumes of each description $\{z_j(t)\,|\,j\in\overline{1,n}\}$. Thus:

$$\forall j \in \overline{1,n} \quad \max_t z_j(t) = z_j(0) = N_j ;$$
$$\forall j \in \overline{1,n} \quad \min_t z_j(t) = z_j(T_j) = 0,$$

where N_j is the initial volume of the goods of the *j*-th description, and T_j is the moment of the complete sale of the initial goods volume.

Further it is assumed, that the change of the trade enterprise state takes place discretely for each separate coordinate $z_j(t)$. Therefore the state change modeling is made for one co-ordinate $z_j(t)$ at the same time the index j is omitted, that is the state co-ordinate is $z(t)$, which satisfies the condition $0 \le z(t) \le N$. Thus $z(0) = N, z(T) = 0$.

The number of the buyers visiting the trade enterprise at the moment *t*, starting from *t* = 1, is a random quantity $m(t)$ which varies in the known limits - $m(t) \in \overline{0, M(t)}$. The probability distribution of occurrence of one or another number of buyers for each moment is assumed to be known, that is

$$\forall t \in \{1,2,3,...\} \quad \left\{P^t_{m(t)} \mid m(t) \in \overline{0, M(t)}\right\}, \tag{2.19}$$

where $P^t_{m(t)}$is the probability of arrival of buyers $m(t)$ at the moment *t*.

Thus the values $P^t_{m(t)}$ satisfy the known relations:

$$\forall t \in \{1,2,3,...\}, \forall m(t) \in \overline{0, M(t)} \quad 0 \le P^t_{m(t)} \le 1,$$
$$\forall t \in \{1,2,3,...\} \quad \sum_{m(t) \in \overline{0, M(t)}} P^t_{m(t)} = 1.$$

Each buyer with the probability *p* buys goods and with the probability (1-*p*) does not buy goods. If at the moment *t* $m(t)$ buyers happened to arrive in the trade enterprise, the volume of goods bought by them is $r(t) \in \overline{0, m(t)}$, and the probability of buying $r(t)$ units of piece-goods by $m(t)$ buyers is defined as

$$P[r(t)] = C^{r(t)}_{m(t)} p^{r(t)} \cdot (1-p)^{m(t)-r(t)}, \tag{2.20}$$

where $C^{r(t)}_{m(t)} = \dfrac{m(t)!}{r(t)!(m(t) - r(t))!}$.

The discrete-stochastic model for constructing the change function of the piece-goods volume, taking into account the assumed preconditions and assumptions is represented by the population of matrixes

$$\left\{\left\| P_{ij}(x) \right\| \mid x \in X\right\}. \tag{2.21}$$

The input signal *x* in the offered model is the number of buyers $m(t)$. Therefore, the model represents the population of $M(t)+1$ matrixes of (2.21) type for each moment $t \in \{1,2,3,...\}$, that is

$$\forall t \in \{1,2,3,\ldots\} \quad \{P_{ij}(m(t)) \mid m(t) \in \overline{0, M(t)}\}. \tag{2.22}$$

The structure of matrixes (2.22), determined by the number of lines and columns, and the values of the elements $P_{ij}(m(t))$, the sum of which in every line of any matrix is equal to “one”, depends on $m(t)$. Below is the look of matrixes (2.22) for various values of $m(t)$.

$$P(0)=\begin{array}{c||ccccccc||}
 & N & N-1 & N-2 & & 2 & 1 & 0 \\
N & 1 & 0 & 0 & \ldots & 0 & 0 & 0 \\
N-1 & 0 & 1 & 0 & \ldots & 0 & 0 & 0 \\
N-2 & 0 & 0 & 1 & \ldots & 0 & 0 & 0 \\
 & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots \\
2 & 0 & 0 & 0 & \ldots & 1 & 0 & 0 \\
1 & 0 & 0 & 0 & \ldots & 0 & 1 & 0 \\
0 & 0 & 0 & 0 & \ldots & 0 & 0 & 1
\end{array},$$

$$P(1)=\begin{array}{c||ccccccc||}
 & N & N-1 & N-2 & & 2 & 1 & 0 \\
N & 1-p & p & 0 & \ldots & 0 & 0 & 0 \\
N-1 & 0 & 1-p & 0 & \ldots & 0 & 0 & 0 \\
N-2 & 0 & 0 & 1-p & \ldots & 0 & 0 & 0 \\
 & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots \\
2 & 0 & 0 & 0 & \ldots & 1-p & p & 0 \\
1 & 0 & 0 & 0 & \ldots & 0 & 1-p & p
\end{array},$$

…

$$P(m(t))=\begin{array}{c||cccccccc||}
 & N & N-1 & \ldots & N-m(t) & \ldots & 1 & 0 \\
N & P(0) & P(1) & \ldots & P(m(t)) & \ldots & 0 & 0 \\
N-1 & 0 & P(0) & \ldots & P(m(t)-1) & \ldots & 0 & 0 \\
N-2 & 0 & 0 & \ldots & P(m(t)-2) & \ldots & 0 & 0 \\
\ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots \\
m(t) & 0 & 0 & \ldots & 0 & \ldots & P(m(t)-1) & P(m(t))
\end{array}.$$

The structure of matrixes $P(m(t))$ assumes satisfying the condition $N > 2M(t)$. At that the lines and columns of the specified matrixes are identified by the corresponding volume of the goods - $N, N-1,...,N-m(t),...,2,1,0$.

The elements of matrix $P(m(t))$ are defined according to the expression (2.20), that is

$$P(0) = C^0_{m(t)} p^0 \cdot (1-p)^{m(t)-0} = 1 - p^{m(t)};$$

$$P(1) = C^1_{m(t)} p^1 \cdot (1-p)^{m(t)-1} = m(t) p (1-p)^{m(t)-1};$$

$$P(2) = C^2_{m(t)} p^2 \cdot (1-p)^{m(t)-2} = \frac{m(t)(m(t)-1)}{2!} p^2 (1-p)^{m(t)-2};$$

$$\ldots$$

$$P(m(t)-1) = C^{m(t)-1}_{m(t)} p^{m(t)-1} \cdot (1-p)^1 = m(t) p^{m(t)-1} (1-p);$$

$$P(m(t)) = C^{m(t)}_{m(t)} p^{m(t)} \cdot (1-p)^{m(t)-m(t)} = m(t) p^{m(t)}.$$

It is obvious, that the sum of elements in each line of all specified matrixes $\|P_{ij}(m(t))\|$ is equal to “one”, that is

$$\sum_{r(t) \in \overline{0, m(t)}} P(r(t)) = \sum_{r(t) \in \overline{0, m(t)}} C^{r(t)}_{m(t)} p^{r(t)} (1-p)^{m(t)-r(t)} = 1,$$

that corresponds to the conditions of defining the random transition automaton.

The procedure of construction of the sold volume change function is realized on the basis of two operations – defining the number of buyers $m(t)$ on the basis of (2.19), and then the state $z(t)$ on the basis of model (2.22), provided that the state $z(t-1)$ is known. These operations are performed by means of the sequence of random equally distributed numbers in the range of (0,1) and the toss scheme method.

The method presupposes selection of a certain number from the sequence of random numbers and its comparison with the values of the sums of probabilities from the lines of matrixes (2.22), that enables to determine the state $z(t)$ into which the modeled object will pass from the state $z(t-1)$. In other words, the given technique completely corresponds to the functioning process of the random transition automaton.

The ordinates of the function under definition are the values of the state of the trade enterprise at various moments $z(t)$ $t \in \{1,2,3...\}$. Each new value of $z(t)$ is defined based on the known previous value of $z(t-1)$. At the same time $z(0) = N$ and $z(t-1) \geq z(t)$, that corresponds to the non-increasing character of the defined function.

2.6. Continuous-stochastic modeling

The basis for constructing analytical models of continuous-stochastic processes is the queuing system theory. Functioning of the queuing systems (QS) has the character of servicing of incoming requests or clients. For execution of the service operations the system has available necessary equipment which is called channels or service lines [21].

The requests can be individual or such that organize a flow the main characteristic of which is the distribution law of the requests inflow in the QS.

If the next request enters the system and finds all channels engaged, it stands in queue for service. The basic characteristics of the queue is its length and the order of a request selection from the queue.

The service process can be structured by separation such its parts as stages. Thus, the service process can be both a one-phase and a multiphase one.

For realization of the service stage one or several service channels are allocated. In the latter case the QS is the multi-channel one.

On the whole, the schema of the multi-channel, multiphase QS with queues is shown in fig. 2.8.

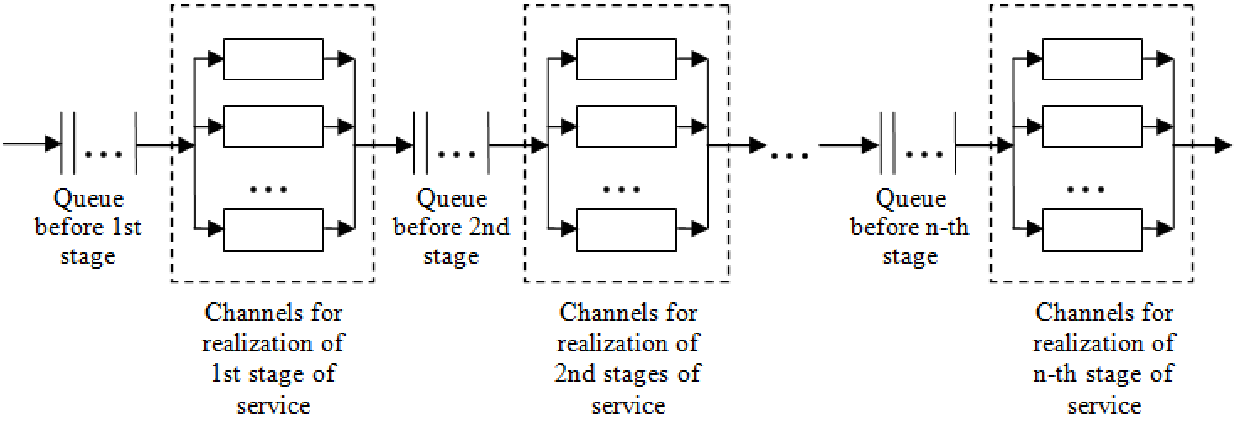

Queue before 1st stage
Channels for realization of 1st stage of service
Queue before 2nd stage
Channels for realization of 2nd stages of service
Queue before n-th stage
Channels for realization of n-th stage of service

Figure 2.8 – The schema of the multi-channel, multiphase QS with queues

Studying an individual request servicing consists in defining if the service was completed and in defining the service duration, as well as in the estimation of the service quality.

While studying the servicing of the requests flow there appear additional tasks – defining the part of the served requests and the part of the rejected requests; defining the busy time and the idle time of channels, etc.

Analytical modeling of the continuous-stochastic processes on the basis of QS consists of the request flow modeling and modeling of the serving channels functioning.

The requests flow modeling assumes defining or setting the character of this flow.

The flow is called a homogeneous events flow if from the servicing point of view all requests are equal and the only thing that matters is the fact of entering or absence of the request in the service system at the present moment. In this case each request is characterized by the arrival moment, and the flow of requests – by the sequence of moments $t_1, t_2, ..., t_k$ or by the law determining the alternation of the moments $t_j \left(j \in \overline{1,k} \right)$.

If requests are not homogeneous, then each of them, except the arrival moment t_j, is characterized by the set of parameters $\alpha_{1j}, \alpha_{2j}, ..., \alpha_{mj}$ expressing the properties of the request. In this case the flow of requests is seen as the flow of vectors $\left(t_j, \alpha_{1j}, \alpha_{2j}, ..., \alpha_{mj} \right) \left(j \in \overline{1,k} \right)$. For example, for a flow of cars arriving at the gas-filling station, besides the arrival moment, the important role is played by such properties of cars as their tanks capacity, the type and grade of fuel, the necessity of rendering additional services. All these properties are expressed by corresponding parameters. Another example is the flow of blanks arriving at an industrial site or an individual machine tool for processing. In this case, besides the arrival moment of certain bars, an important role is played by such parameters as the type, the class of processing, the number of workpiece surfaces, their complexity etc.

For the mathematical description of a random flow of homogeneous requests usually are used the laws of distribution of time intervals between adjacent moments $\xi_j \left(j \in \overline{1,k} \right)$, whose interrelations are shown in fig. 2.9, rather than laws of distribution of moments t_j.

From the specified distribution $t_j, \xi_j \left(j \in \overline{1,k} \right)$ along the time axis it is obvious, that

$$t_1 = 0 + \xi_1,\ t_2 = \xi_1 + \xi_2, ..., t_k = \xi_1 + \xi_2 + ... + \xi_k,$$

that is the representation of the distribution law of intervals $\xi_j \left(j \in \overline{1,k} \right)$ enables to determine the value $t_j \left(j \in \overline{1,k} \right)$ as well.

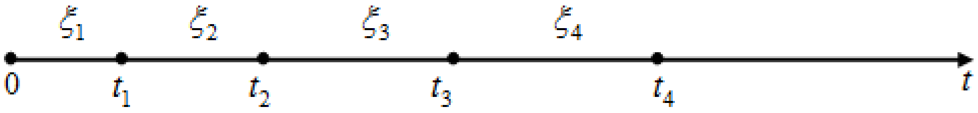


Figure 2.9 – Distribution of moments $t_j\left(j \in \overline{1,k}\right)$ and intervals $\xi_j\left(j \in \overline{1,k}\right)$ between them along the time axis $0t$

Application of the distribution law of intervals $\xi_j\left(j \in \overline{1,k}\right)$ instead of the distribution law of moments $t_j\left(j \in \overline{1,k}\right)$ is explained by continuity of quantities ξ_j and their simple connection with quantities $t_j\left(j \in \overline{1,k}\right)$. Considered are the integral distribution laws of random variables $\xi_j\left(j \in \overline{1,k}\right)$

$$F(z_1, z_2,..., z_k) = P(\xi_1 < z_1, \xi_2 < z_2,..., \xi_k < z_k)$$

and differential distribution laws or the distribution function of probability density which are usually designated as $f(z_1, z_2,..., z_k)$ [21].

If $f(z_1, z_2,..., z_k) = f_1(z_1)\cdot f_2(z_2)\cdot\ldots\cdot f_k(z_k)$, that is random variables are independent, then the flow is called a restricted aftereffect flow. When, besides, the distribution laws are identical $f_j(z_j)$, the flow is called a recurrent flow.

The flow is called ordinary if at any t_0 on the axis of time the probability of occurrence within the range of $(t_0, t_0 + \Delta t)$ of two or more requests tends to zero at $\Delta t \to 0$.

If the probability of entering *k* requests in the interval $(t_0, t_0 + \Delta t)$ does not depend on t_0, but is determined by the values *k* and Δt the flow is called a stationary flow.

For the recurrent flow the average length value of the interval between the arrival moments of requests is determined by the mathematical expectation value

$$m_z = \int_0^\infty z f(z) dz.$$

Magnitude $\lambda = {}^1\!/_{m_z}$ is called the requests flow intensity which determines the average number of requests, arriving per a time unit.

If the flow possesses the properties of stationarity, ordinariness and absence of aftereffect, such flow is called an elementary one or a Poisson flow. The probability of occurrence of *k* requests in the Poisson flow within the interval Δt is determined by the expression

$$P_k(\Delta t) = \frac{(\lambda \Delta t)^k}{k!} e^{-\lambda \Delta t}.$$

The probability distribution density function in this case looks like

$$f(\Delta t) = \lambda e^{-\lambda \Delta t}.$$

This function has a special name – the demonstrative distribution law.

Thus, on the whole, the requests flow modeling is reduced to the representation or definition of the integral or differential distribution law (a probability distribution density function) of intervals between the moments of requests arrival.

Another aspect of analytical modeling of the continuous-stochastic processes is modeling of the servicing process which consists in defining the character of the servicing process, that means more particularly:

- setting the queuing order of a request, in case of the service channels engagement, and choosing a request from the queue in the case of a channel or several channels deallocation;
- setting the assignment order of the relinquished channels for servicing the next request;
- determining the request service time by a channel or several channels.

It is known, that a channel can be found in two states – "free" and “engaged”. The request which entered the service system at the moment t_j, is accepted for service if there is a free channel, or it is put in queue if there are no free channels.

It is often assumed that a request can be in queue during some time Δt. If the request is not accepted for service by the moment $t_j + \Delta t$, it leaves the system. Taking into account the specified assumption, there can be: queuing system (QS) with refusals, when $\Delta t = 0$, QS with expectation, when $\Delta t = \infty$, and finally, mixed QSs, when Δt is a positive random variable.

Examples of requests queue ordering and requests selection can be: receipt of requests on the first-come-first-served basis, and selection is done on the first-come basis; use of various priorities; random orders.

The assignment order of the relinquished channels for servicing the next request can be: attraction of channels in the ascending order of channel numbers; in the order of deallocation; in random order, etc.

An important quantity characterizing the channel functioning, is the request service time or otherwise the channel holding time of the request service. As a rule, this is a random quantity with the set distribution law. For example, the random holding time variable τ is characterized by the exponential law with a distribution density $f(\tau) = \lambda e^{-\lambda \tau}$ $(\tau > 0)$, where λ is the parameter expressing the intensity of service.

The distribution function

$$F(\tau)=\int_0^\tau \lambda e^{-\lambda t}dt=1-e^{-\lambda\tau}$$

represents the probability that the service will be come to an end in time τ. Or otherwise, it is the probability that the channel will be relinquished in time τ.

As the channel can be in one of the two states – "engaged" or “free”, the probability that the channel will be engaged in time τ is defined as

$$1-F(\tau)=1-1+e^{-\lambda\tau}=e^{-\lambda\tau}.$$

If in the system k channels are engaged, the probability that none of them will be relinquished in timeτ is determined as $\left(e^{-\lambda\tau}\right)^k=e^{-k\lambda\tau}$.

Absence of an aftereffect is typical for the queuing system with the requests Poisson flow and an exponential servicing time distribution, that is the future state depends only on the present and does not depend on what happened in the past. Such processes are called Markoff processes [21]. They are the theoretical basis for the analysis of a big number of real queuing systems.

2.7. Generalized models on the basis of the aggregate

One of the basic problems of complex systems modeling is the problem of simultaneous modeling of different types of processes –continuous-determined, discrete-determined etc. The problem is that the mathematical schemes for modeling of different types of processes do not conjugate with each other, which does not permit to receive the generalized model of the complex system.

The solution of the problem of the generalized model construction can be development of a universal mathematical scheme, enabling to represent the peculiarities of all types of processes – continuous, discrete, determined, stochastic and their various combinations. Such universal mathematical scheme is the aggregate system or simply the aggregate. This schema was offered by a group of scientists under the guidance of N. P. Buslenko [14].

To ensure the universality the following mechanisms are realized in the aggregate [22]:

- the mechanism of a state modification of the dynamic system under the influence of the inherent causes resulting from interaction of the system elements;
- the mechanism of the input signal reception from the external environment and the state modification under the influence of this signal;

- the mechanism of the output signal forming in response to modification of the dynamic system state under the influence of inherent causes or a signal from the external environment.

The functioning of a dynamic system and of an aggregate representing it is reduced to the performance of the specified mechanisms.

For examining the mechanisms let's assume, that at the initial moment t_0 the dynamic system is in the state z^0, where z^0 is an interior point of the closed area Z in the n-dimensional space of the system states.

Within the limits of the mechanism of the state modification under the influence of inherent causes the dynamic system passes from the state z^0 into other states $z_t \in Z$, corresponding to the time moments $t > t_0$, while performing motion $z(t)$.

The character of the causes supporting the motion does not change up to the appearance of point z_t at the boundary of the closed area *Z*. Let the instant of arrival at the boundary be t^* and the state $z^* = z_{t^*}$.

The population of the ordered couples (t, z_t) for $t_0 < t < t^*$, that is $\{(t, z_t) | t_0 < t < t^*\}$, represents the fragment of motion $z(t)$ corresponding to the interval (t_0, t^*). For specifying this fragment of motion within the area *Z* it is necessary to specify correlations defining the values z_t for $t \in (t_0, t^*)$ at the known values t_0 and z^0. For example, it can be a differential equation

$$\frac{dz}{dt} = f(t, z)$$

with the entry conditions: at $t = t_0$, $z(t_0) = z^0$.

For defining $t^*$and z^* it is necessary to solve jointly the motion equation of the point z_t and the equation of the area boundary *Z*.

The moment t^* is called the reference moment. At this moment the system state z^* is discontinuous, it changes instantly, and the system turns into the state $z' = z_{t^*+0}$, where $t^* + 0$ is the time moment close to t^*.

The choice of the state $z' \in Z$ is realized randomly according to the probability distribution which depends on t^* and z^*, that formally is represented by the entry

$$z' = P\left(z'\left(t^*, z^*\right)\right).$$

The motion fragment connected with the transition from the point (t^*, z^*) to the point $(t^* + 0, z')$ of the phase space is called a state saltus at the exit onto the boundary.

Besides, at the moment t^* within the limits of the mechanism of the output signal forming, the output signal $y_{t^*} = G^*(t^*, z^*)$ depending on t^* and z^* is defined. Function G^* is the set one, having sometimes random character. Process of transition from the state z^0 into the state z^* and further into the state z' is shown in fig. 2.10.

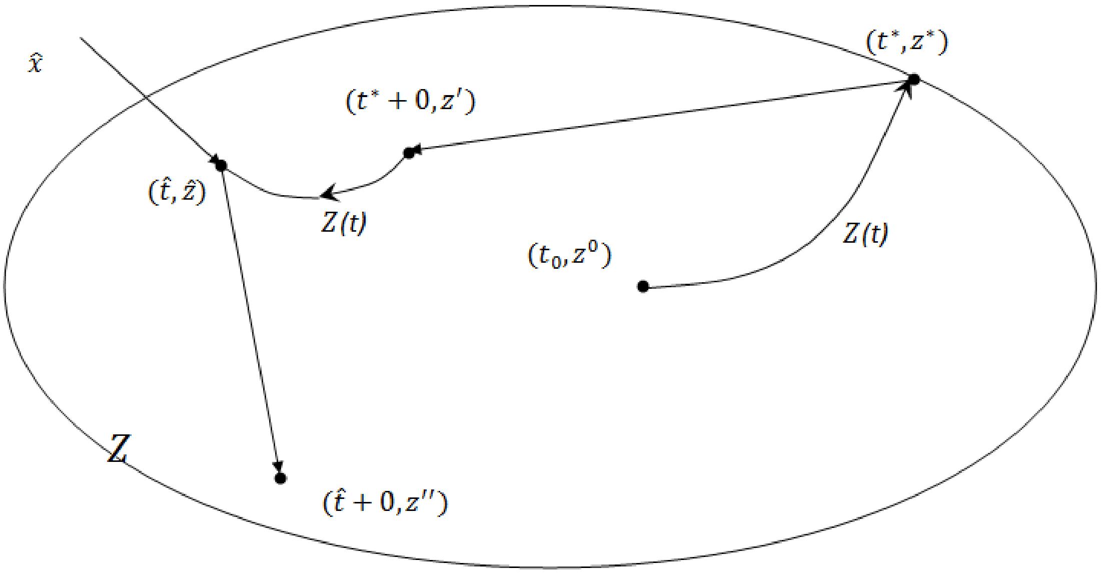


Figure 2.10 –The schema of the modification of states within the mechanisms of the states modification under the influence of inside causes and under the influence of the input signal

From the state z' under the influence of inherent causes the system turns into other states, making new transition within the area Z, a new saltus of the state when it exits onto the boundary etc.

All transitions in the area Z under the influence of inherent causes stop when an input signal $\hat{x}$ enters from the external environment. Let us assume more particularly, that at the moment $\hat{t}$ when the system was in the state $\hat{z}$, the input signal $\hat{x}$ entered.

At the moment $\hat{t}$ the transition under the influence of inherent causes stops and from the state $\hat{z}$ the system in discrete steps, instantly gets into the state $z'' = z_{\hat{t}+0}$ which is an interior point of the area Z.

The choice of the state $z'' \in Z$ is carried out randomly according to the probability distribution which depends on $\hat{t}, \hat{x}, \hat{z}$, i.e.

$$z'' = P\left(z''\left(\hat{t}, \hat{z}, \hat{x}\right)\right).$$

The fragment of motion connected with the transition from the point $\left(\hat{t}, \hat{z}\right)$ to the point $\left(\hat{t} + 0, z''\right)$ is called the state saltus at the entry of the input signal.

The moment $\hat{t}$ is the reference moment as well. At this moment, besides the specified state saltus which is realized within the limits of the reception mechanism of the input signal and the state modification under the influence of this signal, the output signal is formed and produced

$$y_{\hat{t}} = \hat{G}\left(\hat{t}, \hat{z}, \hat{x}\right),$$

where $\hat{G}$ – is known beforehand, probably random function which depends on the moment $\hat{t}$, state $\hat{z}$ and on the input signal $\hat{x}$.

Further from the state z'' the system turns into other states within the area *Z* under the influence of inherent causes, makes new saltuses while entering onto the boundary or at the entry of a new input signal from the external environment etc.

Thus, the process of the dynamic system functioning with discrete interference of events, represented by means of an aggregate, consists of transitions of the point $\left(t, z_t\right)$ within the area $\left(t, Z\right)$ which can be interrupted by state saltuses at reference moments t^* and $\hat{t}$ i.e. when entering onto the boundary or at the entry of an input signal, as is shown in fig. 2.10.

Mathematically the aggregate is represented by:

1) the equation of the area *Z* boundary;

2) the equations of the point z_t motion within the area *Z*;

3) the correlations for calculation of the probability distribution of the state saltus when entering onto the boundary and at the entry of an input signal from the external environment;

4) the correlations for calculation of output signals when entering onto the boundary and at the entry of an input signal.

All the specified equations and relations are the characteristics of the general dynamic system with discrete interference of events.

The generality of the considered dynamic system is expressed in the fact that by imposing some restrictions on the specified characteristics, it is possible to obtain the previously considered classes of dynamic systems.

In particular, if to assume for example, that saltuses are zero, i.e. the state co-ordinates before and after the saltus coincide, and the motion $z(t)$ is the differentiable function the required number of times, we obtain the class of dynamic systems, the processes in which have the continuous-determined character and which are described by the corresponding system of ordinary differential equations.

Further, if to impose restrictions onto transition under the influence of inherent causes, considering them to be zero, and to reduce all the state modifications of the system to state saltuses under the influence of input signals from the external environment we obtain the class of dynamic systems, processes in which have the discrete-stochastic character and which are described on the basis of the probabilistic automaton.

If to add to the restrictions specified above the restrictions consisting in the fact that the probability distribution of the state saltus represents a set of zeros and units we obtain the class of dynamic systems, processes in which have the discrete-determined character and which are described on the basis of the finite automata.

The closest to the general dynamic system with discrete interference of events described by means of an aggregate, are the dynamic systems which are described on the basis of the queuing systems theory. For these systems nonzero transitions within an area *Z* under the influence of inherent causes and nonzero state saltuses are common.

The characteristics of the general dynamic system with discrete interference of events, that is the corresponding equations and proportions, are selected from the specified classes of analytical expressions. For example, the equations can be chosen linear or nonlinear, with constants or variable factors etc. The proportions for calculating the probability distribution of the state saltus can consider or not consider the process pre-history.

The characteristics concretization that is their selection from the specified classes, enables to single out some most often used types of aggregates for describing the general dynamic system with discrete interference of events. The piecewise-linear aggregate (PLA) pertains to such most often used types of aggregate.

For representation of the PLA all the characteristics of the general dynamic system with discrete interference of events are necessary. The characteristics concretization for the PLA consists in the fact that the set of states *Z* is divided into parts or pieces in the form of non-overlapping subsets z_k. The boundaries of z_k are defined by simple equations specifying in n-dimensional Euclidean space certain hyperplanes which are sides of a polyhedron. Thus, the subsets z_k are polyhedrons in Euclidean space. In two-dimensional Euclidean space subsets z_k are polygons; their approximate location is shown in fig. 2.11. The basic co-ordinates of Euclidean space in fig. 2.11 are z_{01}, z_{02}. Disjoint polygons are designated as $z_1, z_2,..., z_\upsilon,..., z_k$. Each polygon $z_\upsilon \left(\upsilon \in \overline{1,k}\right)$ has a local coordinate system – $z_{\upsilon 1}, z_{\upsilon 2}$.

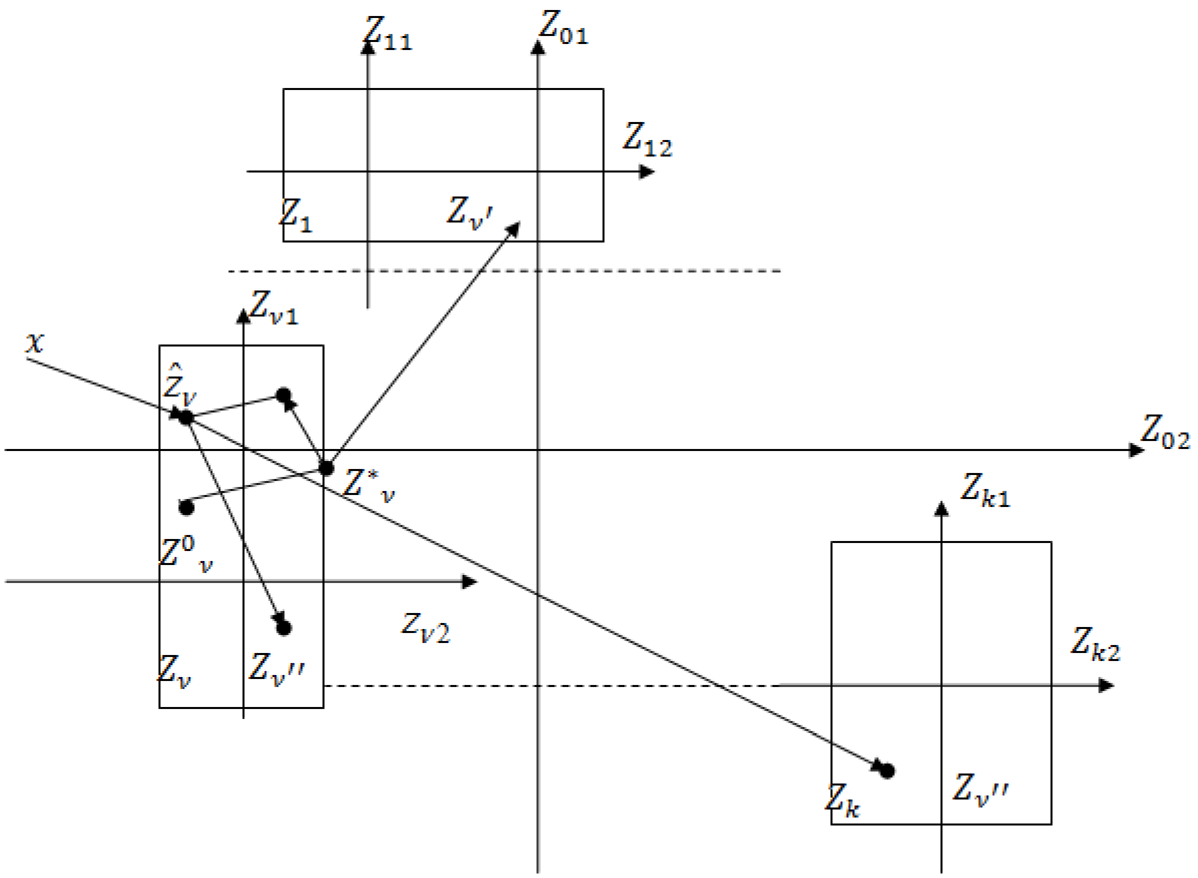


Figure 2.11 – Distribution of the PLA state subsets in 2-dimensional Euclidean space

The PLA state is represented in the form $z = (\upsilon, z_\upsilon)$, where υ is the subset (polyhedron) index, and z_υ is the point within the polyhedron with its co-ordinates in the local coordinate system $z_{\upsilon 1}, z_{\upsilon 2},..., z_{\upsilon n}$. In the two-dimensional case presented in fig. 2.11 the local system has two co-ordinates – $z_{\upsilon 1}, z_{\upsilon 2}$, that is the PLA state in this case is defined as $z = (\upsilon, z_{\upsilon 1}, z_{\upsilon 2})$.

The component υ is called a discrete component of the state or the ground state, and vector z_υ – the vector of additional co-ordinates.

Further the PLA characteristics concretization is connected with the functioning process – with the specification of concrete proportions and equations describing the transitions and saltuses within polyhedrons.

Let at the initial moment t_0 the PLA be in the state $z^0 = (\upsilon, z^0{}_\upsilon)$. The possible location of the point $z^0{}_\upsilon$ for a two-dimensional case is shown in fig. 2.11. At $t > t_0$ a certain motion $z^0{}_\upsilon(t)$ is executed within the polyhedron υ. In case of the PLA the motion is set by simple equations of the following kind

$$\frac{dz_\upsilon}{dt} = \alpha_\upsilon,$$

where $\alpha_{\upsilon} = (\alpha_{\upsilon 1}, \alpha_{\upsilon 2}..\alpha_{\upsilon n})$ is a constant vector.

The solution of the specified equation is represented as follows:

$$z_{\upsilon}(t) = z^{0}{}_{\upsilon} + \alpha_{\upsilon}(t - t^{0}),$$

that is the transition of a point within a polyhedron is realized under the linear law as shown in fig. 2.11 for a two-dimensional case.

The motion $z_{\upsilon}^{0}(t)$ within the polyhedron under the linear law is supposed to continue until the trajectory of motion reaches the polyhedron boundary in the point z_{υ}^{*}, as shown in fig. 2.11.

The point z_{υ}^{*} is reached at the moment t^{*} which is, as earlier, is defined to be the reference moment. At this moment two important operations take place – an instant state saltus as a result of entering onto the boundary and an output signal production.

The instant state saltus occurs from the point $\left(\upsilon, z_{\upsilon}^{*}\right)$ into the point $\left(\upsilon^{/}, z_{\upsilon'}\right)$ which is an internal point of the given or new polyhedron. The instant saltus is shown in fig. 2.11 in the form of arrows directed into the same or into the first polyhedron, that is $\upsilon^{/} = \upsilon$ or $\upsilon^{/} = 1$.

The new discrete component $\upsilon^{/}$ is chosen randomly according to the probability distribution $P_{\upsilon\upsilon'}$ which can depend, for example, on the polyhedron index υ , on the point of exit onto the boundary z_{υ}^{*}, on the number of the polyhedron face where the co-ordinate z_{υ}^{*} is situated, etc.

From the point $\left(\upsilon^{/}, z_{\upsilon'}\right)$ the motion $z_{\upsilon'}(t)$ is executed in the same way as earlier from the point $\left(\upsilon, z^{0}{}_{\upsilon}\ \right)$.

As it was noted, the second operation when entering onto the boundary is an output signal production.

The output signal $y = \left(\alpha, y_{\alpha}\right)$ has a discrete component α which usually defines the signal type – sound, light, etc., and the vector of additional co-ordinates $y_{\alpha} = \left(y_{\alpha 1}, y_{\alpha 2}, ..y_{\alpha n}\right)$ whose components are signal characteristics.

The motion within the polyhedron can be interrupted as a result of entering an input signal from the external environment.

Let at the moment $\hat{t}$, which is the reference one, an input signal $x = \left(\mu, x_{\mu}\right)$ enters from the external environment, where μ – is the discrete component of the signal, defining, as a rule, the type of the signal, and $x_{\mu} = \left(x_{\mu 1}, x_{\mu 2}...x_{\mu n}\right)$ – is the vector of additional co-ordinates whose components are the signal characteristics which express, for example, the signal frequency, amplitude, power, etc.

At the moment $\hat{t}$ when the aggregate was in the state $(\upsilon, \hat{z}_{\upsilon})$, where $\hat{z}_{\upsilon} = \hat{z}_{\upsilon}(\hat{t})$, the motion within the polyhedron stops, and again two operations take place – an instant state saltus as a result of the input signal entry and the output signal production.

The instant state saltus occurs from the point $(\upsilon, \hat{z}_{\upsilon})$ into the point $(\upsilon'', \hat{z}_{\upsilon''})$ which can be an internal point of the same or another polyhedron. Entering of the input signal x and the instant saltus caused by it are shown in fig. 2.11 in the form of the arrow x and the arrows starting at $\hat{z}_{\upsilon}$ and directed onto the same polyhedron $\upsilon'' = \upsilon$ or in another polyhedron, for example $\upsilon'' = k$.

The selection of the discrete component υ'' is realized randomly on the basis of probability distribution $Q_{\upsilon\upsilon''}$ which can depend on $\upsilon, \hat{z}_{\upsilon}, \mu$.

The output signal is formed in the same way as in the case of a point exit onto the polyhedron boundary, that is in the form $y = (\alpha, y_{\alpha})$.

Further the transition $z_{\upsilon''}(t)$ within the polyhedron $z_{\upsilon''}$ takes place, which turns into a new state saltus due to reaching the polyhedron boundary or due to an input signal entry from the external environment.

2.8. The model of elements composition in the complex system

As a rule, any complex system, assumes presence of a big number of elements between which various links are established.

Through these links in the form of certain signals the information used for functioning of both individual elements, and the whole system is transmitted.

These links ensure «durability» of the system and therefore are important «building material» for the system which necessarily should be considered while constructing the model of the system. Presence of links is taken into account in the interface models.

While constructing the interface models, the following preconditions and assumptions are applied [22].

To concentrate on the links between the elements, these elements are represented in the form of a «black box» model – an input signal $x = (x_1, x_2..x_n)$ is fed to the element's input where each characteristic of the input signal $x_i (i \in \overline{1, N})$ has its range of definition X_i, that is $x_i \in X_i$, and at the element's output appears an output signal $y = (y_1, y_2..y_l)$, where $y_l \in Y_l \quad (l \in \overline{1, r})$, that is each characteristic of the output signal also has its range of definition.

For reception of the input signal $x=(x_1,x_2..x_n)$ the system element $e_j(j\in\overline{1,N})$ has a set of input contacts $\left[X_i^j\right]_1^{n_j}$ where contact X_i^j is intended for receiving the i-th characteristic of the input signal, that is for receiving x_i, and n_j – is the number of input contacts which may be individual for each element.

For the production of the output signal $y=(y_1,y_2..y_r)$the system element $e_j(j\in\overline{1,N})$ has a set of output contacts $\left[Y_l^j\right]_1^{r_j}$ where contact Y_l^j is intended for output of the l-th characteristic of an output signal, that is for production of $y_l(l\in\overline{1,r})$, and r_j – is the number of output contacts which may be individual for each element.

Thus, the system element is represented by the scheme in fig. 2.12.

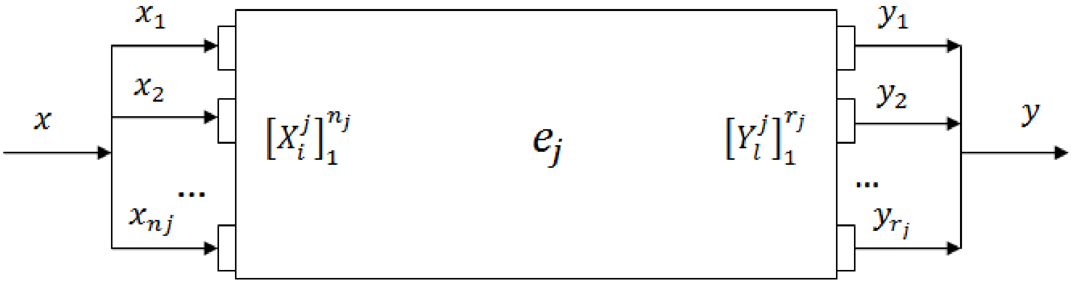


Figure 2.12 – Graphic representation of a system element in the interface models

It is further assumed, that signals are transmitted through the links in the form of elementary channels independently from each other. At the same time not more than one channel is connected to each input contact, while any number of channels come from the output contact, provided that they go to various elements of the system. The given assumption enables to avoid overlapping of the signals arriving from various sources.

External environment for the system considered is represented by two basic methods in the form of the population of elements $S=\left\{e_j \mid j\in\overline{1,N}\right\}$.

According to the first method, the external environment is represented by one separately chosen element. Usually for this element $j=0$, that is the external environment is represented by the element e_0, which as well as the system elements, is characterized by the set of input and output contacts - $[X_i^o]_1^{n_0},[Y_l^0]_1^{r_0}$.

If the external environment is represented in the form of separately chosen element, all the set of input contacts and the set of output contacts of all the system elements and the external environment can be defined as

$$[X_i^j]_N = \bigcup_{j\in\overline{0,N}} [X_i^j]_1^{n_j}; \qquad [Y_l^j]_N = \bigcup_{j\in\overline{0,N}} [Y_l^j]_1^{r_j}.$$

According to the second method, the external environment is represented as the set of elements called signal sources, and by the set of elements called consumers (receivers) of signals. Each signal source $\left\{\hat{e}_{0k} \mid k\in\overline{1,k^*}\right\}$ is characterized by the set of output contacts $[Y_l^{Ok}]_i^{r_k}\,(k\in\overline{1,k^*})$, and each signal receiver $\left\{\tilde{e}_{oj} \mid j\in\overline{1,j^*}\right\}$ is characterized by the set of input contacts $[X_i^{oj}]_i^{n_j}\,(j\in\overline{1,j^*})$.

If the external environment is represented as the sets of sources and receivers in this case the sets of input and output contacts of all the system elements and the external environment can be defined as

$$[X_i^j]_N = \{\bigcup_{j\in\overline{1,N}} [X_i^j]_1^{n_j}\} \cup \{\bigcup_{j\in\overline{1,j^*}} [X_i^{oj}]_1^{n_j}\};$$

$$[Y_l^j]_N = \{\bigcup_{j\in\overline{1,N}} [Y_l^j]_1^{r_j}\} \cup \{\bigcup_{k\in\overline{1,k^*}} [Y_l^{ok}]_1^{r_j}\}.$$

On the basis of sets $[X_i^j]_N$ and $[Y_l^j]_N$ we introduce the operator R realizing the representation $R:[X_i^j]_N \to [Y_l^j]_N, (Y_l^k = R(X_i^j))$. The operator R puts to each input contact X_i^j an output contact Y_l^k in correspondence. Physically it means, that contacts X_i^j, Y_l^k are linked through an elementary channel.

For example, let there be a system $S=\left\{e_j \mid j\in\overline{1,5}\right\}$. The external environment includes two sources $\left\{\hat{e}_{Ok} \mid K\in\overline{1,2}\right\}$ and three receivers of the information $\left\{\tilde{e}_{Oj} \mid j\in\overline{1,3}\right\}$. There are certain links between the system elements, sources and the external environment receivers which are shown in fig. 2.13.

The representation of the interface operator R in the form of a table is most often used. In this table the lines correspond to the numbers of input contacts (i), and the columns correspond to the numbers of the system elements and the external environment with input contacts.

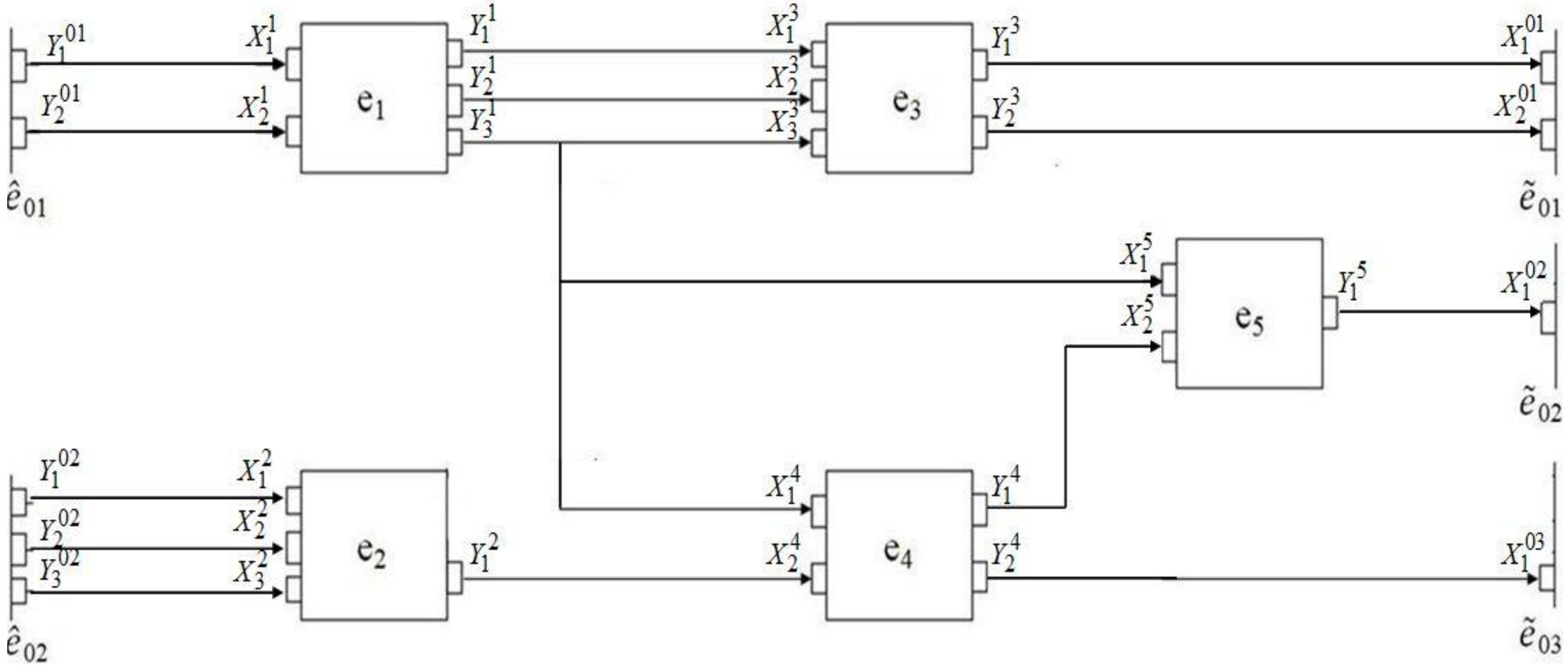


Figure 2.13 – An example of links between the system elements and the external environment elements

At the intersection of the line with *i* index and the column with *j* index, that corresponds to the input contact X_i^j, there is written the pair of indexes (*k*, *l*), where *k* specifies the number of the element of the system or of the external environment, and *l* specifies the number of the output contact Y_l^k with which the contact X_i^j is linked through the elementary channel.

The table, setting the interface operator R for the considered example of the system and the external environment, shown in fig. 2.13, has the following form (tab. 2.7).

Table 2.7 – The tabular form of definition of operator R

i	j							
	1	2	3	4	5	01	02	03
1	01,1	02,1	1,1	1,3	1,3	3,1	5,1	4,2
2	01,2	02,2	1,2	2,1	4,1	3,2		
3		02,3	1,3					

The process of filling in table 2 consists in defining the pair of indexes (*k*, *l*) at the intersection of lines with the index i and columns with the index *j*. For example, let *i* = 1, *j* = 1, that means fixing of the first input (*i* = 1) of the first element $e_1\ (j=1)-X_1^1$. According to the schema in fig. 2.13, the input X_1^1 is linked by the elementary channel with the first output contact (l=1) of the external environment element $\hat{e}_{01}$, that is *k* = 01. Thus, in the cell at the intersection of the first line (i=1) and the first column (*j* = 1) should be written down the pair (*k*, *l*) = (01,1).

Another form of the representation of the interface operator R is the incidence matrix. Such representation is more bulky, than in the tabular form, but it enables to use well developed and known methods of the graph theory for analyzing the structure of the links.

The population of sets $[X_i^j]_1^{n_j}$, $[Y_l^j]_1^{r_j}$ for all the system elements $S=\left\{\left\langle e_j \,\middle|\, j\in\overline{1,N}\right\rangle\right\}$ and the external environment $\left\{\left\langle \hat{e}_{ok} \,\middle|\, k\in\overline{1,k^*}\right\rangle\right\}\left\{\left\langle \tilde{e}_{oj} \,\middle|\, j\in\overline{1,j^*}\right\rangle\right\}$ together with the interface operator R make up the interface schema in the canonical form.

The interface schema in the canonical form is the basic interface model for the representation of links between the system and the external environment elements.

Interface models can be one- and multi-level depending on the fact if the system elements pertain directly to a system or the elements pertain to the subsystems which are distributed according to the hierarchy levels.

The interface model of the system elements in fig. 2.13 is, obviously, a single-level one, as the system elements pertain directly to the system.

For constructing a interface multilevel model it is necessary to distribute the elements by the subsystems.

For certainty, let the elements $e_j(j \in \overline{1,N})$ be distributed by the subsystems $S_\mu(\mu \in \overline{1,M})$ in such a way that each element e_j pertains only to one of the subsystems S_μ, containing not fewer than one element.

For example, in the system S in fig. 2.13 the following distribution of the elements by the subsystems is possible: $S_1 = \{e_1, e_3\}, S_2 = \{e_2, e_4\}, S_3 = \{e_5\}$, that is the system is represented as $S = \left\{\left\langle S_\mu \middle| \mu \in \overline{1,3} \right\rangle\right\}$.

The distinction between system structures when the elements pertain directly to the system and when elements form subsystems, is shown in fig. 2.14.

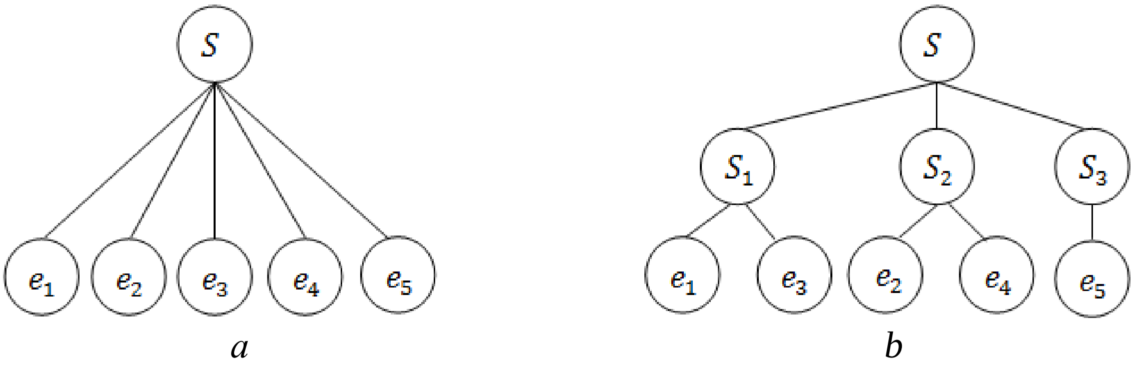


Figure 2.14 – Variants of system structures when the elements pertain directly to the system (a) and when the elements are united in the subsystems (b)

The multilevel interface model, when the system elements are united in a certain way into subsystems, is formed with unconditional taking into account the duality principle. According to this principle each subsystem is considered from two points of view. On the one hand it is considered as a system, and on the other hand – as an element.

The subsystem $S_\mu(\mu \in \overline{1,M})$ as a system has a certain structure, that is the composition elements together with the connections between the elements and connections with the external environment. At the same time the connections with the external environment are realized through the contacts $X_i^{o\mu}$ and $Y_l^{o\mu}$, characterizing the receivers and the environment sources accordingly.

The subsystem $S_\mu(\mu \in \overline{1,M})$ as an element is represented by the model of the «black box» type, that is characterized only by input $\hat{X}_i^\mu$ and output $\hat{Y}_l^\mu$ contacts for connecting with other subsystems and the external environment.

Thus, considering the subsystem $S_\mu(\mu \in \overline{1,M})$ taking into account the duality principle leads to the necessity of introducing "double" contacts at the subsystem boundary.

For example the system *S* is presented, in which three subsystems S_1, S_2, S_3 are specified, the boundaries of the subsystems with "double" contacts are shown in fig. 2.15.

At the same time the origin points of "double" contacts are cross points of the boundary contour (dashed line) with the links of the given subsystem with other subsystems and the external environment.

Let us take the subsystem $S_1 = \{e_1, e_3\}$, and firstly consider it as a system. The elements e_1, e_3 pertaining to S1, have connections between themselves and with the external environment represented, as usual, by sources and receivers of signals. The output contacts Y_1^{01}, Y_2^{01} refer to the source, and the input ones $X_1^{01}, X_2^{01}, X_3^{01}$ refer to receiver. At the same time the external environment for the subsystem S1 as a system is formed by other subsystems S2 and S3 as well as by the previously defined external environment in the form of sources $\hat{e}_{01}, \hat{e}_{02}$ and receivers $\tilde{e}_{01}, \tilde{e}_{02}, \tilde{e}_{03}$. Exactly from such external environment the subsystem S1 as a system receives signals through the contacts Y_1^{01}, Y_2^{01} and produces signals through contacts $X_1^{01}, X_2^{01}, X_3^{01}$.

Now let us consider the subsystem S1 as an element, that is as a «black box» model type. In this case we are not interested in the structure S1, that is its composition and links between its elements, we are interested only in input and output contacts. The input contacts are $\hat{X}_1^1, \hat{X}_2^1$ and the output ones are $\hat{Y}_1^1, \hat{Y}_2^1, \hat{Y}_3^1$.

Comparing the contacts of the subsystem S1 as an element and the previously specified contacts of the subsystem S1 as a system, we obtain the pairs of contacts in the cross points of the subsystem S1 boundary with the connections with other subsystems and the external environment:

$$\left(\hat{X}_1^1, Y_1^{01}\right), \left(\hat{X}_2^1, Y_2^{01}\right), \left(X_1^{01}, \hat{Y}_1^1\right), \left(X_2^{01}, \hat{Y}_2^1\right), \left(X_3^{01}, \hat{Y}_3^1\right).$$

These pairs of contacts also are "double" contacts of the subsystem S1.

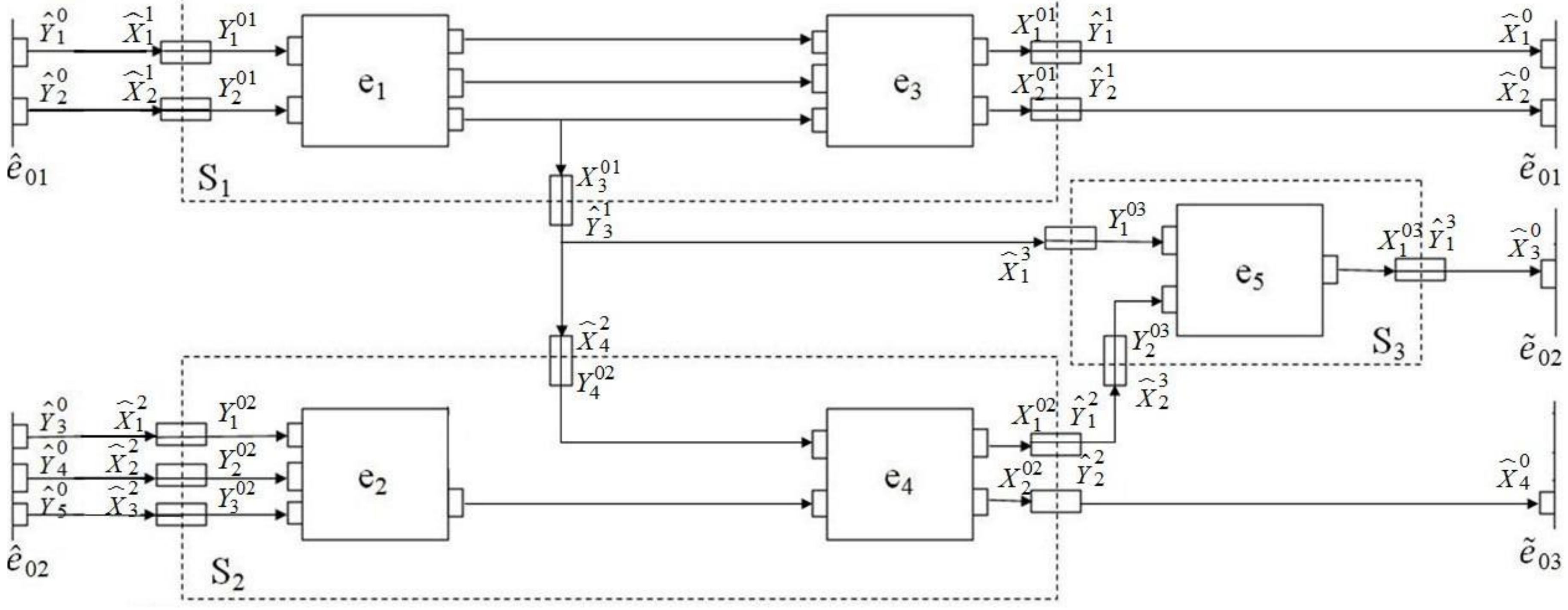


Figure 2.15 – The system structure with subsystems S1, S2, S3, where "double" contacts appear

Based on similar reasoning "double" contacts for other subsystems S2, S3 of the system *S* are defined:

$$\left(\hat{X}_1^2, Y_1^{02}\right), \left(\hat{X}_2^2, Y_2^{02}\right), \left(\hat{X}_3^2, Y_3^{02}\right), \left(\hat{X}_4^2, Y_4^{02}\right), \left(X_1^{02}, \hat{Y}_1^2\right), \left(X_2^{02}, \hat{Y}_2^2\right), \left(\hat{X}_1^3, Y_1^{03}\right), \left(\hat{X}_2^3, Y_2^{03}\right), \left(X_1^{03}, \hat{Y}_1^3\right).$$

The obtained "double" contacts are included into the corresponding sets of contacts. Thus, the contacts of $X_i^{0\mu}$ type are included into the sets $\left[X_i^{0\mu}\right]$, of $\hat{Y}_l^{\mu}$ type – into the sets $\left[\hat{Y}_l^{\mu}\right]$, of $Y_l^{0\mu}$ types – into the sets $\left[Y_l^{0\mu}\right]$, and finally, of $\hat{X}_i^{\mu}$ types – into the sets $\left[\hat{X}_i^{\mu}\right]$.

Now let us consider the subsystems S_μ as a system and as an element taking into account the introduced "double" contacts or taking into account the obtained sets $\left[X_i^{0\mu}\right], \left[Y_l^{0\mu}\right], \left[\hat{X}_i^{\mu}\right], \left[\hat{Y}_l^{\mu}\right]$. The elements of the subsystem S_μ as a system are still characterized by sets of input $\left[X_i^j\right]_i^{n_j}$ and output $\left[Y_l^j\right]_i^{r_j}$ contacts of all those elements forming S_μ, that is all $e_j \in S_\mu$. The external environment for the subsystem S_μ is represented by the sets of contacts $\left[X_i^{0\mu}\right], \left[Y_l^{0\mu}\right]$.

The operator R_μ, such that $Y_l^k = R_\mu\left(X_i^j\right)$, is introduced. The range of definition of the operator R_μ is the set $\left[X_i^{0\mu}\right] \cup \left\{ \bigcup_{e_j \in S\mu} \left[X_i^j\right]_1^{n_j} \right\}$, and the range of values is the set of contacts $\left[Y_l^{0\mu}\right] \cup \left\{ \bigcup_{e_k \in S\mu} \left[Y_l^k\right]_i^{r_k} \right\}$.

The operator R_μ is called an interior interface operator of the subsystem S_μ. It describes the links between the subsystem S_μ forming elements and the external environment represented by the set of contacts $\left[X_i^{0\mu}\right], \left[Y_l^{0\mu}\right]$.

The operator R_μ is mostly represented in a tabular form similar to the table for the interface operator R of the interface single-level model (tab. 2.7).

The tables, setting operators R_μ of subsystems S_μ of the considered example (fig. 2.15), have the following form (tab. 2.8 – 2.10).

Table 2.8 – Interface operator R_1

i	j		
	0	1	3
1	3,1	01,1	1,1
2	3,2	01,2	1,2
3	1,3		1,3

Table 2.9 – Interface operator R_2

i	j		
	0	2	4
1	4,1	02,1	02,4
2	4,2	02,2	2,1
3		02,3	

Table 2.10 – Interface operator R_3

i	j		
	0	5	
1	5,1	03,1	
2		03,2	

The subsystem S_μ as the element of the system S is characterized by a set of input $\left[\hat{X}_i^\mu\right]$ and output $\left[\hat{Y}_l^\mu\right]$ contacts. The external environment for the subsystem S_μ in this case is the elements-sources $\hat{e}_{01},\hat{e}_{02}$ and the elements-receivers $\tilde{e}_{01},\tilde{e}_{02},\tilde{e}_{03}$ which are characterized by the set of contacts $\left[\hat{X}_i^0\right],\left[\hat{Y}_l^0\right]$.

There introduced the operator R^2 such that $\hat{Y}_l^\upsilon = R^2\left(\hat{X}_i^\mu\right)$. The range of definition of the operator R^2 is the set $\bigcup\limits_{\mu\in\overline{0,M}}\left[\hat{X}_i^\mu\right]$, and the range of values is the set $\bigcup\limits_{\upsilon\in\overline{0,M}}\left[\hat{Y}_l^\upsilon\right]$.

The operator R^2 is called the subsystems interface operator in the system S and is represented in the tabular form like the previously considered operators. This table for the considered example of the system in fig. 2.15 looks like in tab. 2.11.

Table 2.11 – Interface operator R^2

i	μ			
	0	1	2	3
1	1,1	0,1	0,3	1,3
2	1,2	0,2	0,4	2,1
3	3,1		0,5	
4	2,2		1,3	

The set of contacts $\left[\hat{X}_i^\mu\right],\left[\hat{Y}_l^\mu\right]$ and the interface operator R^2 form the interface model of the second level for the system S.

The population of the internal single-level interface models of all the subsystems $S_\mu \left(\mu \in \overline{1,M}\right)$ together with the second level interface model form a two-level interface model of the system S which fully characterizes the structure of relations between the system elements.

Subsystems S_μ can merge into larger subsystems, that leads to the necessity of deriving a three-level interface model with the interface operator R^3 on the third level. Proceeding similarly, we obtain multilevel models of elements interface in complex systems.

The system complexity is sometimes expressed in its variability, controllability and stochasticity of the structure.

The specified properties of the complex system structure should be considered, for example, in the following cases:

- during a military operation when there is a transfer or resubordination of military subdivisions and the means of their support depending on the developing circumstances;
- in the course of functioning of flexible industrial systems intended for manufacturing of several types of products, while changing over from one type of product to another;
- in the course of functioning of multiprocessor computers whose operational system supports the redistribution of the computational resources workload depending on the nature of the current task.

The variability of the complex system structure is understood as the temporal variability of input and output contact sets of both individual elements and subsystems in multilevel systems, as well as the temporal variability of all types of interface operators R.

Taking into consideration temporal changes of output and input contacts consists in the fact that eventually some contacts disappear, and other may appear.

Dependence of the interface operator R on time is ensured through the set of tables, each of which is valid within a certain interval (t_i, t_{i+1}) and varies while passing to the next interval.

Controllability of the complex system structure is understood as the modification of input and output contacts sets of elements and subsystems, as well as the dependence of interface operators on some parameter $\alpha \in A$ where A – is a range of values of α.

For example, while designing the model of a flexible industrial site, the intensity of blanks arrival may be considered as the control parameter.

In that case when α is some random event, and A – is a set of events on which the probability distribution of event occurrence is set, it is possible to speak about the stochasticity of the complex system structure. The element composition and relations between the elements of such system have random character. There considered the problems connected with the estimation of distribution laws influence on the character of the systems functioning.

Interface models in the form of a interface schema in the canonical form are used for solving any problems of analyzing the pattern of interface of complex systems elements. However, for solving certain problems it is expedient to take advantage of the interface model in the form of the so-called interface schema in the natural form. The expediency is explained by simplification of solution of the analysis problems.

The basic concept of interface models in the form of interface schemas in the natural form is the concept of the plug as the population of the element e_j contacts, connected by elementary channels with the element e_k contacts.

Let there be the system $S = \left\{e_j \mid j \in \overline{1,N}\right\}$, whose external environment is K^* external sources with a set of output contacts $\left[Y_l^{0k}\right]_1^{2k} \left(k \in \overline{1,K^*}\right)$ and j^* receivers of the information with a set of input contacts $\left[X_i^{0j}\right]_1^{n_j} \left(j \in \overline{1,j^*}\right)$.

First of all let us consider the construction peculiarities of the natural form of a one-level interface schema, that is a case when the system elements are not distributed by the subsystems, and pertain directly to the system.

Let us take the element $e_j \left(j \in \overline{1,N}\right)$ or $\left(j \in \overline{01,0j^*}\right)$ which pertains to a system or to external receivers, and the element e_k which pertains to a system or to external sources. Let us define the set of input contacts $\left\lfloor X^{jk} \right\rfloor$of the element e_j connected with the elemente_k. If the set $\left\lfloor X^{jk} \right\rfloor \neq 0$ that is not empty, then the set of input contacts $X_i^j \in \left\lfloor X^{jk} \right\rfloor$ arranged in ascending index i ordering is designated as ξ_p^j and is called the input plug of the element e_j.

If the set $\left\lfloor X^{jk} \right\rfloor = 0$, then the corresponding plug does not exist.

The number of input plugs ξ_p^j is equal to the number of the external environment sources and the elements of the system S from which the element e_j receives input signals.

The set of input plugs of the element e_j is designated as $\left\lfloor \xi_p^j \right\rfloor$.

Let ξ_p^j be some input plug of the element e_j. For each contact $X_i^j \in \xi_p^j$ we will define $Y_l^k = R\left(X_i^j\right)$. All the output contacts Y_l^k received by means of the interface operator R, according to the definition of the input plug pertain to the same element e_k.

The ordered set of all the output contacts Y_l^k, connected by elementary channels with the input contacts of the plug ξ_p^j, are designated η_q^k and are called as output plugs of the element e_k.

The number of the output plugs η_q^k of the element e_k is equal to the sum of the number of elements e_j of the system *S* and the number of the external environment receivers, to which the element e_k transmits its output signals.

The set of the output plugs of the element e_k is designated as $\lfloor \eta_q^k \rfloor$.

All the information about the system S plugs and the plugs of the external environment are entered into a special table.

Let, for example, there be a system $S = \{e_j \mid j \in \overline{1,4}\}$ with the external environment of two sources $\hat{e}_{01}, \hat{e}_{02}$ and two receivers $\tilde{e}_{01}, \tilde{e}_{02}$. The links between the system elements and the environment elements are shown in fig. 2.16.

Let us make up a table with the information on the plugs of all the elements of the system and the external environment, shown in fig. 2.16. The elements of the external environment $\hat{e}_{01}, \hat{e}_{02}$, being the signal sources, have only output plugs. In particular, element $\hat{e}_{01}$ has one output plug $\eta_1^{01} = \{Y_1^{01}\}$ for connecting with the system element e_1, and element $\hat{e}_{02}$ has two output plugs:

- $\eta_1^{02} = \{Y_1^{02}\}$ for connection with the system element e_1;
- $\eta_2^{02} = \{Y_2^{02}\}$ for connection with the system element e_2.

System elements e_1, e_2, e_3, e_4 have both input and output plugs.

The element e_1 has two input plugs:

- $\xi_1^1 = \{X_1^1\}$ for connection with the external source $\hat{e}_{01}$;
- $\xi_2^1 = \{X_2^1\}$ for connection with the external source $\hat{e}_{02}$

and four output plugs:

- $\eta_1^1 = \{Y_3^1, Y_4^1\}$ for connection with element e_2;
- $\eta_2^1 = \{Y_1^1, Y_2^1\}$ for connection with element e_3;
- $\eta_3^1 = \{Y_1^1\}$ for connection with the external receiver $\tilde{e}_{01}$;
- $\eta_4^1 = \{Y_3^1\}$ for connection with the external receiver $\tilde{e}_{02}$.

The element e_2 has two input plugs:

- $\xi_1^2 = \{X_3^2\}$ for connection with the external source $\hat{e}_{02}$;
- $\xi_2^2 = \{X_1^2, X_2^2\}$ for connection with element e_1

and two output plugs:

- $\eta_1^2 = \{Y_2^2\}$ for connection with element e_3;

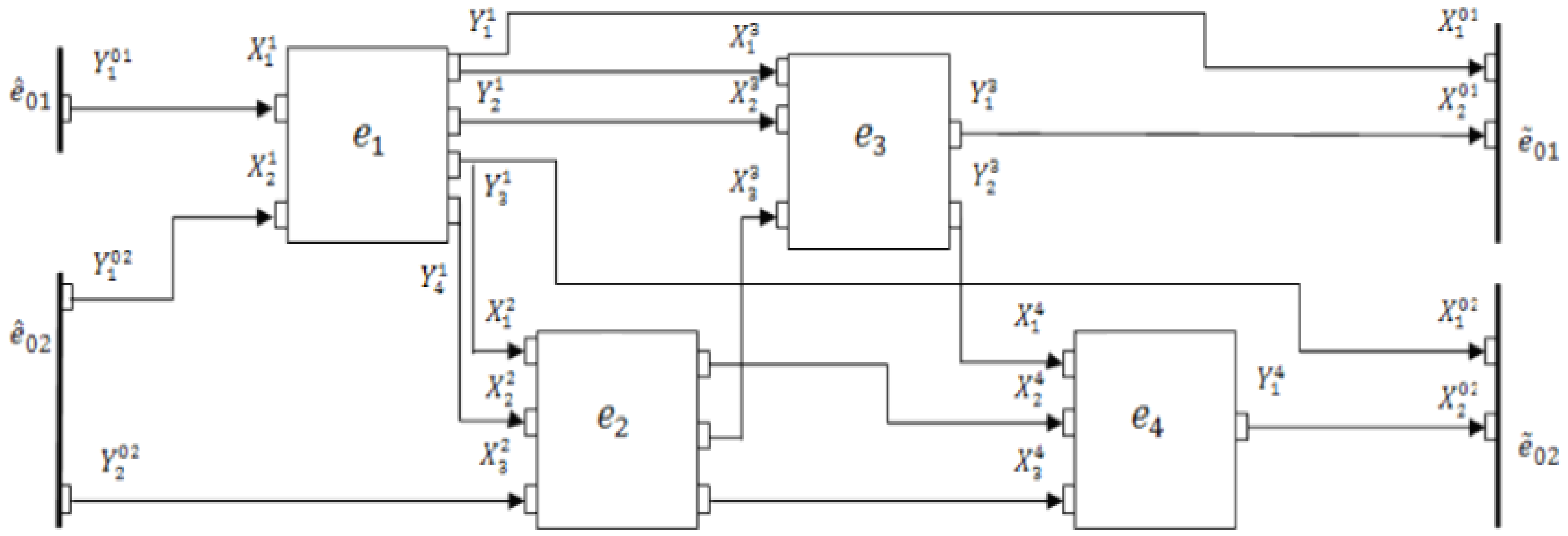


Figure 2.16 – Example of system S and the external environment, which elements form plugs

- $\eta_2^2 = \{Y_1^2, Y_3^2\}$ for connection with element e_4.

The element e_3 has two input plugs:

- $\xi_1^3 = \{X_1^3, X_2^3\}$ for connection with element e_1;
- $\xi_2^3 = \{X_3^3\}$ for connection with element e_2

and two output plugs:

- $\eta_1^3 = \{Y_2^3\}$ and two output plugs e_4;
- $\eta_2^3 = \{Y_1^3\}$ for connection with the external receiver $\tilde{e}_{01}$.

The e_4 has two input plugs:

- $\xi_1^4 = \{X_2^4, X_3^4\}$ for connection with element e_2;
- $\xi_2^4 = \{X_1^4\}$ for connection with element e_3

and one output plug $\eta_1^4 = \{Y_1^4\}$ for connecting with the external receiver $\tilde{e}_{02}$.

The external environment elements $\tilde{e}_{01}, \tilde{e}_{02}$ are the receivers of signals and therefore have only input plugs.

The element $\tilde{e}_{01}$ has two input plugs:

- $\xi_1^{01} = \{X_1^{01}\}$ for connection with element e_1;
- $\xi_1^{02} = \{X_2^{02}\}$ for connection with element e_4.

The element $\tilde{e}_{02}$ has two output plugs:

- $\xi_1^{02} = \{X_1^{02}\}$ for connection with element e_1;
- $\xi_2^{02} = \{X_2^{02}\}$ for connection with element e_4.

All the specified data on the plugs composition is recorded in a special table 2.12, whose lines are identified by numbers of plugs; and columns – by numbers of elements and the external environment.

In the cells of table 2.12 at the intersection of lines and columns are recorded the indexes of the contacts forming plugs. At the same time the contacts forming the input plugs are written down in the upper part of the cell, while the contacts forming the output plugs are written down in the lower part of the cell.

The information on the plugs composition enables to proceed to the construction of a single-level interface schema for system *S* in the natural form.

For realization of construction it is necessary to determine the sets of input and output plugs $[\xi_p^j], [\eta_q^k]$, where *j*, *k* – are indexes of the system elements and the external environment, *p*, *q* – are indexes (numbers) of the plugs included in the set for the element considered.

Table 2.12 – Data on plugs composition

Plug number	Elements							
	$\widehat{01}$	$\widehat{02}$	1	2	3	4	$\widetilde{01}$	$\widetilde{02}$
1	—	—	(1,1)	(2,3)	(3,1); (3,2)	(4,2); (4,3),	(01,1)	(02,1)
	(01,1)	(02,1)	(1,3); (1,4)	(2,2)	(3,2)	(4,1)	—	—
2	—	—	(1,2)	(2,1); (2,2)	(3,3)	(4,1)	(01,2)	(02,2)
	—	(02,2)	(1,1); (1,2)	(2,1); (2,3)	(3,1)	—	—	—
3	—	—	—	—	—	—	—	—
	—	—	(1,1)	—	—	—	—	—
4	—	—	—	—	—	—	—	—
	—	—	(1,3)	—	—	—	—	—

Thus, for the example of system *S* and the external environment in fig. 2.16 the sets of plugs look like:

$$\left[\xi_p^1\right]=\left\{\xi_1^1,\xi_2^1\right\},\ \left[\eta_q^1\right]=\left\{\eta_1^1,\eta_2^1,\eta_3^1,\eta_4^1\right\},$$

$$\left[\xi_p^2\right]=\left\{\xi_1^2,\xi_2^2\right\},\ \left[\eta_q^2\right]=\left\{\eta_1^2,\eta_2^2\right\},$$

$$\left[\xi_p^3\right]=\left\{\xi_1^3,\xi_2^3\right\},\ \left[\eta_q^3\right]=\left\{\eta_1^3,\eta_2^3\right\},$$

$$\left[\xi_p^4\right]=\left\{\xi_1^4,\xi_2^4\right\},\ \left[\eta_q^4\right]=\left\{\eta_2^4\right\},$$

$$\left[\xi_p^{01}\right]=\left\{\xi_1^{01},\xi_2^{01}\right\},\ \left[\eta_q^{01}\right]=\left\{\eta_1^{01}\right\},$$

$$\left[\xi_p^{02}\right]=\left\{\xi_1^{02},\xi_2^{02}\right\},\ \left[\eta_q^{02}\right]=\left\{\eta_1^{02},\eta_2^{02}\right\}.$$

The interface operator of the system *S* elements and the external environment in the natural form is called operator $\eta_q^k=\rho\left(\xi_p^j\right)$ which assigns to every input plug ξ_p^j a unique output plug η_q^k connected to it by a channel being generally a population of elementary channels, linking the contacts forming the specified plugs.

The interface operator ρ is assigned by means of a table, whose structure and the filling-in rules are similar to the table for the representation of the interface operator R. In particular, the table for the interface operator ρ in the considered example of a system and the external environment in fig. 2.16 looks like table 2.13. Interface scheme defined by the interface operator from table 2.13 is shown on fig. 2.17.

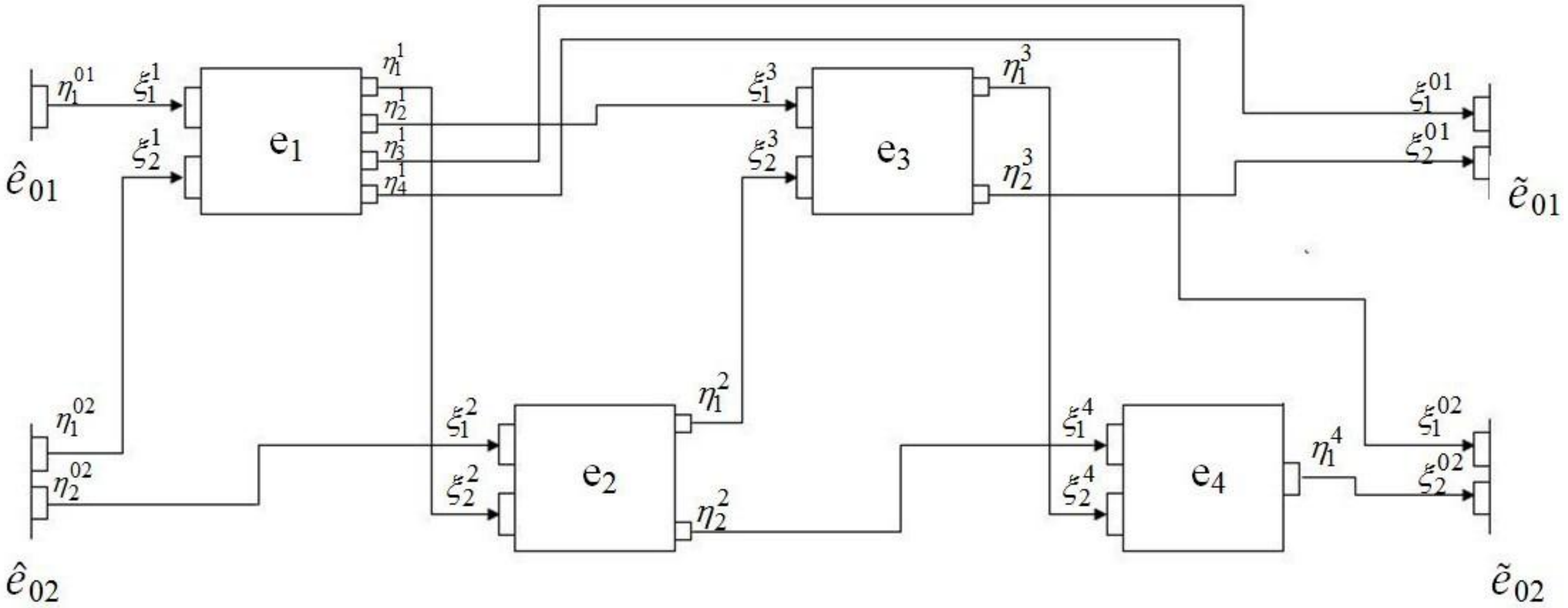


Figure 2.17 – The interface schema of the system elements and the external environment in the natural form

Table 2.13 – Example of representation of the interface operator ρ

Plug	The system elements and the external environment elements					
	1	2	3	4	01	02
1	01,1	02,2	1,2	2,2	1,3	1,4
2	02,1	1,1	2,1	3,1	3,2	4,1

The interface schemas in fig. 2.16 and fig. 2.17 are the interface schemas of the same system and the external environment. Their comparison enables to draw a conclusion, that the interface schema in the natural form in fig. 2.17 is somewhat simpler than the interface schema in the canonical form in fig. 2.16. The simplification is achieved due to the fact that the number of plugs and therefore the number of connections in fig. 2.17 of the system elements is fewer than the number of contacts in fig. 2.16.

This simplification is the principal justification of expediency of using interface schemas in the natural form for the analysis of complex systems relations.

The population of sets of input $\left[\xi_p^j\right]$ and output plugs $\left[\eta_q^k\right]$together with the table of plugs and the interface operator ρ is called the interface model in the form of single-level interface schemas of the system S elements and the external environment in the natural form.

The construction procedure of the interface model in the form of a multilevel interface schema in the natural form is similar to the construction procedure of the interface model in the form of a multilevel interface schema in the canonical form.

Summary

The given chapter considers analytical modeling of processes that take place in complex systems. The following types of processes in dynamic systems were distinguished: continuous-determined, discrete-determined, discrete-stochastic and continuous-stochastic. To describe these type of processes the special mathematical schemes are applied. They include finite automata, probabilistic automata, differential equations and queue systems. The character of the process in the system allows to define the type of model and to choose the appropriate mathematical scheme for its presentation.

Tasks for case-study

1. Let’s consider crossroads with traffic controller. Identify type of modeling and choose the mathematical scheme. Ground your decision and build the model of the investigated object.
2. Let’s consider a bank with a single teller. We suppose that customers arrive to the bank. If the teller is busy they wait in the single queue. They are served in order of their arrivals. After service all customers leave the bank. Identify type of modeling and choose the mathematical scheme. Ground your decision and build the model of the investigated object.
3. Let’s consider a warehouse. All kinds of goods have their own shelves. The content of these shelves is being changed with the goods movement. These moments are the time of goods input and output. Identify type of modeling and choose the mathematical scheme. Ground your decision and build the model of the investigated object.
4. Let’s consider a simple telephone system with two external lines. If both lines are busy the callers wait. They are served in order of their arrivals. The maximum waiting time is 4 minutes. Identify type of modeling and choose the mathematical scheme. Ground your decision and build the model of the investigated object.
5. Let’s consider a supermarket. Customer can take a cart. Each cart is locked until person put in a special hole either 25 or 50 kopecks. In this case the cart will be unlocked automatically. You can put in the same time only one kind of kopeck. Identify type of modeling and choose the mathematical scheme. Ground your decision and build the model of the investigated object.
6. Let’s consider CD-ROM that reads Audio CD. We put a CD and push the button and it starts to play the selected track. Identify type of modeling and choose the mathematical scheme. Ground your decision and build the model of the investigated object.
7. Let’s consider a microwave oven. We use a microwave oven to cook pizza. To cook pizza we need open a door, put pizza and close a door. After that we push button “start”. When pizza is cooked the microwave oven makes a signal. Identify type of modeling and choose the mathematical scheme. Ground your decision and build the model of the investigated object.
8. Let’s consider an elevator in the three-floor house. Identify type of modeling and choose the mathematical scheme. Ground your decision and build the model of the investigated object.
9. Let’s consider a gas station. In a gas station there are two gas pumps. Cars arrive at the gas station. Cars are served in order of arrival. Identify type of modeling and choose the mathematical scheme. Ground your decision and build the model of the investigated object.

Check questions

1. Give the definition of a finite automaton.
2. What parameters do you define for the description of a finite automaton?
3. What kinds of automata do you know?
4. What differs Mealy automaton from Moore one?
5. When do we use synchronous and asynchronous automata?
6. What do you know about probabilistic automata?
7. Give the definition of probabilistic automaton.
8. Why do we need to consider some random factors for description of investigated object?
9. What differs deterministic from probabilistic automaton?
10. How transitions of automaton with random transitions can be described?
11. How outputs of automaton with random outputs can be described?
12. What do you know about D-scheme?
13. In what cases do we usually use determined continuous models?
14. Provide some well-known examples of modeling with differential equations.
15. In what areas of human activity you can use D-scheme for modeling?
16. Give the definition of a notion “differential equation”.
17. What type of dynamical system is described by queue theory?
18. Which queuing disciplines do you know?
19. What are the names of the queue characteristics?
20. What characteristics do you need to describe an arrival process in a queue system?
21. What characteristics do you need to describe an service process in a queue system?
22. What characteristics do you need to describe a queue in a queue system?

3. STATISTICAL MODELING

3.1. The essence of the statistical modeling method and the main areas of its application

The essence of the statistical modeling method known as a simulation method of Monte-Carlo statistical test, lies in the development of the simulation model of the functioning process of the object under study at random input stimulus or random changes of the internal parameters. It is obvious, that the statistical modeling method is a constituent of those mathematical schemes on the basis of which the mathematical model for studying stochastic objects is developed. These mathematical schemes include, first of all, the previously considered probabilistic automatons, queuing systems and aggregates. This is the main area of this method application which can be illustrated by the following example.

Let there be some system *S* on which input some random input signals x_1 and x_2 are fed, as it is shown in fig. 3.1.

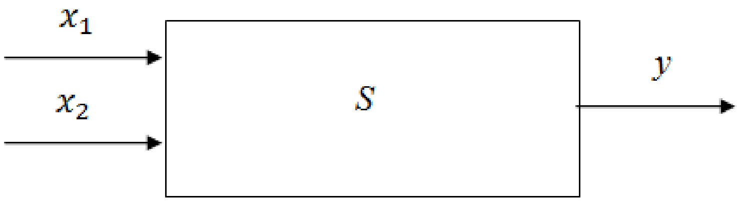


Figure 3.1 – The example of a system with random inputs

It is known, that $x_1 = 1 - e^{-\lambda}$, $x_2 = 1 - e^{-\mu}$, where λ, μ are random variables for which the distribution function (an integral distribution law) is known

It is known also, that $y = \sqrt{x_1^2 + x_2^2}$.

It is obvious, that *y* is a random variable.

It is necessary to estimate the mathematical expectation of random variables *y*, that is *M*[*y*].

The realization of the statistical modeling method for estimating the value of *M*[*y*] is reduced to the realization of the simulation model in the form of the following algorithm:

1. On the basis of the predetermined distribution laws of random variables λ and μ their values are generated: λ_i, μ_i $(i \in \overline{1,N})$, where *N* – the number of values, or the sample size of random variables.

2. For each set of values λ_i and μ_i we calculate the value of the output variable *y*:

$$\forall i \in \overline{1,N} \;\; y_i = \sqrt{(1-e^{-\lambda_i})^2 + (1-e^{-\mu_i})^2} \, .$$

3. On the basis of the set of values $\{y_i \mid i \in \overline{1,N}\}$ the average value of random variables *y* is determined:

$$\overline{y} = \frac{1}{N} \sum_{i \in \overline{1,N}} y_i \, .$$

At $N \to \infty$ the obtained value $\overline{y}$ converges to the defined estimate of expectation that is

$$\lim_{N \to \infty} \overline{y} = M[y] \, .$$

Another direction of the method's application is the solution of deterministic tasks where random factors are not considered. An example of this usage may be the solution of the task of determining the area under the curve *F*(*x*) within interval $0 \le x \le 1$. At that $0 \le F(x) \le 1$.

The geometrical interpretation of the solution of formulated task is shown in fig. 3.2.

For solving deterministic tasks using the statistical modeling method it is necessary to create a stochastic system, which input characteristics at the increase of the simulation number *N* will tend to the solution of the deterministic task.

The following may be a stochastic system variant for solving the deterministic task of determining the area under curve *F*(*x*) using the statistical modeling method.

A pair of independent random variables (x_i, x_{i+1}) at interval (0,1) is generated. This pair may be considered both as an argument and a value of function *F*(*x*) accordingly, or in other words as an abscissa and an ordinate of function *F*(*x*) on a coordinate plane (fig. 3.2).

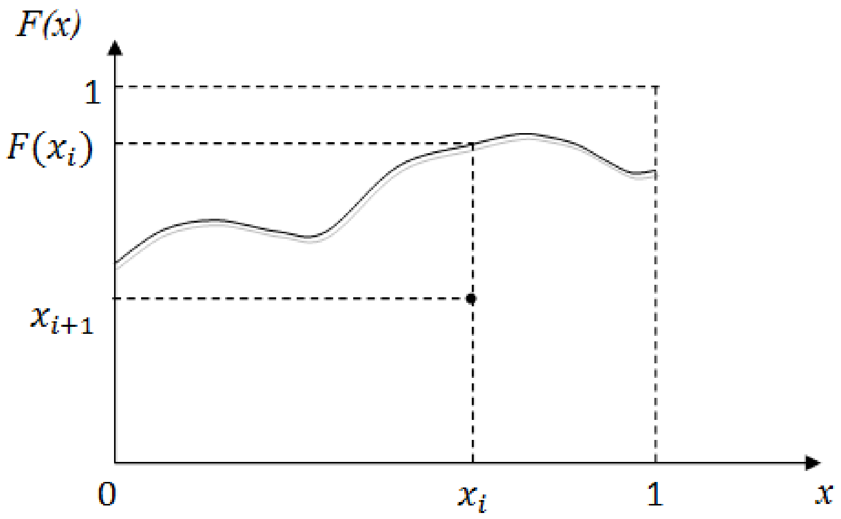


Figure 3.2 – Function $F(x)$ on the interval $0 \le x \le 1$

If point (x_i, x_{i+1}) as it is shown in fig. 3.2, belongs to the area under curve $F(x)$, signal $h_i = 1$ is produced. Otherwise, $h_i = 0$.

This is formalized by the entry:

$$h_i = \begin{cases} 1, \text{if } \ x_{i+1} \le F(x_i), \\ 0, \text{if } \ x_{i+1} > F(x_i). \end{cases}$$

The value is determined

$$\tilde{S} = \frac{1}{N} \sum_{i \in \overline{1,N}} h_i \,.$$

Value $\tilde{S}$ tends to the value of the determined area S with the increase of number N, that is $\lim\limits_{N \to \infty} \tilde{S} = S$.

3.2. Pseudorandom numbers and the procedures of their generation

In the statistical modeling method, as a rule, the generation of random numbers is used. In practice the generation is realized by three methods: instrumental or physical, tabular or file, algorithmic or software-based [6].

The instrumental method consists in generating random numbers by means of a special electronic device which is called an oscillator or a random-number generator. The physical basis of obtaining random numbers by means of the oscillator is the noises in electronic and semiconductor devices.

The process of obtaining the random numbers sequence by means of the instrumental method is realized according to scheme in fig. 3.3.

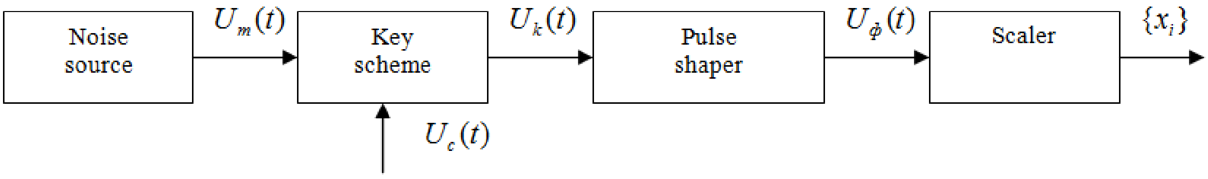


Figure 3.3 – Block diagram of the formation of the random numbers sequence by means of instrumental method

The following notations are used in fig. 3.3:

$U_m(t)$ – noise voltage at the output of an electronic or semiconductor device;

$U_c(t)$ – a pulse amplitude on interval (0, *T*);

$U_k(t)$ – a noise segment on interval (0,*T*) which amplitude value are compared with the value of the master pulse amplitude;

$U_\phi(t)$ – a pulse series at instants t_i $(i \in \overline{1,N})$ on interval [0, *T*], arising at coincidence of the noise amplitude values and the value of the master pulse amplitude;

$\{x_i\}$ – a sequence of random numbers out of interval (0,1).

The noise voltage transformation $U_m(t)$ into a pulse sequence, arising at moments t_i out of interval [0, *T*], is shown in fig. 3.4.

The moments of generation of impulses t_i are captured and on their basis are determined the values

$$x_i = \frac{t_i' - t_i}{T}, \text{ where } t_i' \geq t_i, \quad t_i, t_i' \in (0,T),$$

the set of which $\{x_i\}$ forms the set of random numbers out of interval (0,1).

The main shortcomings of the instrumental method are the impossibility of recurrence of the obtained random variables sequence, which is sometimes necessary for carrying out the repeated experiment, and the necessity to assure the operating stability of the random value generator. If the stability is not ensured, there may appear situations, when $U_c(t) > \max\limits_{t\in[0,T]} U_m(t)$ or when $U_c(t) < \min\limits_{t\in[0,T]} U_m(t)$. It is easily seen, that in the given situations the intersection of amplitude $U_m(t)$ and amplitude $U_c(t)$ on interval *(0, T)* is impossible, thus it is impossible to obtain impulses on interval *(0, T)*.

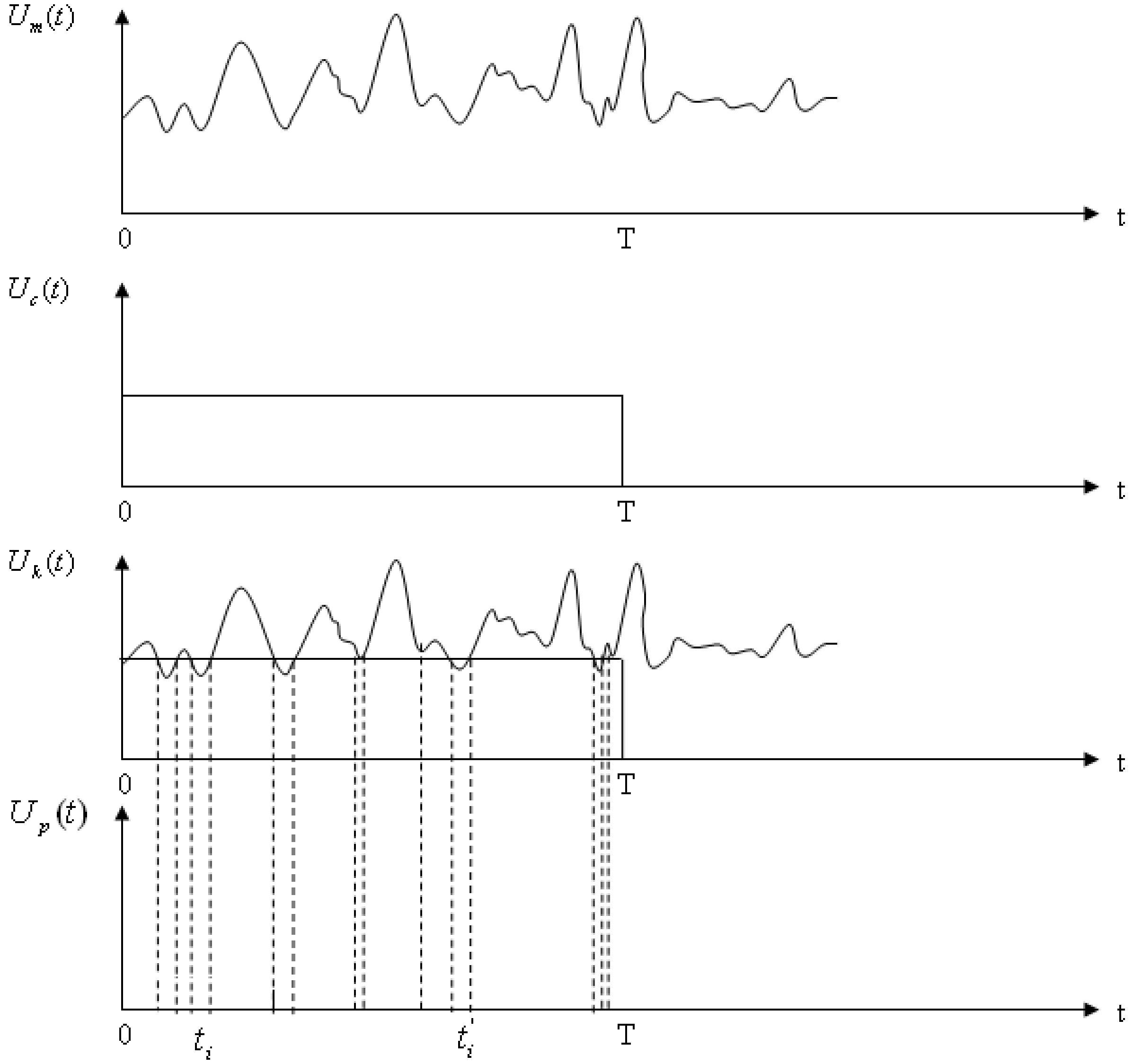


Figure 3.4 – The example of superposition of graph $U_m(t)$ onto $U_c(t)$ for the obtaining of the pulse sequence

The tabular or file method implies that previously generated random numbers are combined into a table which is put into a book or in computer memory, after having previously transformed this random numbers sequence in an appropriate file.

Shortcomings of the tabular or file method are obvious enough and consist mainly in the boundedness of the random numbers sequence and in non-rational usage of the computer memory.

The algorithmic or software-based method is realized by a computer on the basis of special algorithms and programs.

This method permits to obtain the sequence of random values, following any specified distribution laws. Such a possibility is ensured by the functional transformation of the base sequence of random variables which is viewed as the sequence of random variables uniformly distributed on interval (0,1).

Strictly speaking, the algorithmic method permits to obtain the sequence of pseudorandom variables, at least, for two reasons. Firstly, the computer operates with a finite set of numbers and, secondly, the usage of algorithms introduces certain order, regularity that is determinacy.

Widely used are the congruent generation procedures of the pseudorandom variables sequence.

Two numbers *a* and *b* are congruent (comparable) modulo *m*, where *m* – is a positive integer, when and only when there exists such a positive integer *k*, that $a-b=k\cdot m$ and if numbers *a* and *b* give identical remainder in division by *m*.

For example, numbers a=375 and b=125 are congruent modulo m=10.

Indeed, $a-b=375-125=250=25\cdot 10$, that is k=25, m=10.

Here 5 is an identical remainder in division a=375 and b=125 by m=10, that is written down as $5\equiv 375(\mathrm{mod}10)$, $5\equiv 125(\mathrm{mod}10)$.

Congruent procedures have a determinate character and are set by the recurrence formula

$$x_{i+1}\equiv \lambda x_i+\mu(\mathrm{mod}\,M) \tag{3.1}$$

where x_i,λ,μ,M – non-negative integers.

Let's expand the relation (3.1), supposing at first that i = 0.

$$x_1\equiv \lambda x_0+\mu(\mathrm{mod}\,M).$$

Further let i = 1.

$$x_2\equiv \lambda x_1+\mu(\mathrm{mod}\,M)\equiv \lambda(\lambda x_0+\mu)+\mu(\mathrm{mod}\,M)\equiv \lambda^2x_0+\mu(\lambda+1)(\mathrm{mod}\,M)\equiv$$

$$\equiv \lambda^2x_0+\mu\frac{\lambda^2-1}{\lambda-1}(\mathrm{mod}\,M)$$

Let i = 2.

$$x_3=\lambda x_2+\mu(\mathrm{mod}\,M)\equiv \lambda(\lambda^2x_0+\mu\frac{\lambda^2-1}{\lambda-1})+\mu(\mathrm{mod}\,M)\equiv$$

$$\equiv \lambda^3x_0+\mu(\lambda^2+\lambda+1)(\mathrm{mod}\,M)\equiv \lambda^3x_0+\mu\frac{\lambda^3-1}{\lambda-1}(\mathrm{mod}\,M).$$

The regularity has the following form:

$$x_i\equiv \lambda^i x_0+\mu\frac{\lambda^i-1}{\lambda-1}(\mathrm{mod}\,M) \tag{3.2}$$

Thus, if values x_0, λ, μ, M are known, then expression (3.2) determines the sequence of the non-negative integers $\{x_i\}$, composed of the remainders in division of the sequence members $\left\{\lambda^i x_0 + \mu \frac{\lambda^i - 1}{\lambda - 1}\right\}$ by value *M*.

Since $x_i < M$, then on the basis of sequence $\{x_i\}$ we may build the sequence $\{x_i\} = \{x_i / M\}$ which is the rational numbers sequence out of a unit interval (0,1).

The congruent procedure set by expression (3.2), is called a mixed generation method of pseudorandom semi-uniformly distributed numbers on interval (0,1).

If in expression (3.2) let $\mu = 0$, we obtain the expression which is the basis of the multiplicative generation method of pseudo-random numbers quasi-uniformly distributed within interval (0,1).

If to put in expression (3.2) $\mu = 0$, we obtain the expression which is the base of the multiplicative method for obtaining pseudo-random numbers quasi-uniformly distributed within interval (0,1)

Let's make use of the multiplicative method for obtaining pseudo-random semi-uniformly distributed numbers on interval (0,1) supposing that $\lambda = 3, x_0 = 4, M = 11$.

The method is realized by the series of the following steps:

1. Lets form a sequence of numbers $\{\lambda^i x_0\} = \{\lambda x_0, \lambda^2 x_0, \lambda^3 x_0, \lambda^4 x_0, ...\} =$
$= \{12, 36, 108, 324, 972, 2916, 8748, ...\}$

2. We form the sequence of numbers $\{x_i\}$ as the sequence of the remainders in division of the numbers from sequence $\{\lambda^i x_0\}$ by *M* at corresponding values of *i*. This sequence has the following form

$$\{x_i\} = \{1, 3, 9, 5, 4, 1, 3 ...\} \qquad (3.3)$$

3. We form the sequence of rational numbers out of set (0,1). This sequence for the considered example has the following form:

$$\{x_i\} = \{\frac{1}{11}, \frac{3}{11}, \frac{9}{11}, \frac{5}{11}, \frac{4}{11}, \frac{1}{11}, \frac{3}{11}, ...\} = \{0.09, 0.27, 0.81, 0.45, 0.36, 0.09, 0.27, ...\}.$$

The quality of the obtained pseudo-random semi-uniformly distributed numbers is checked by means of uniformity, stochasticity and independence tests.

The checkout for uniformity may be carried out by means of a bar graph in fig. 3.5.

For checking the interval (0,1) is divided into m equal parts ($m = 20 \div 50$). Each part forms a subinterval which is identified by index $j \in \overline{1, m}$.

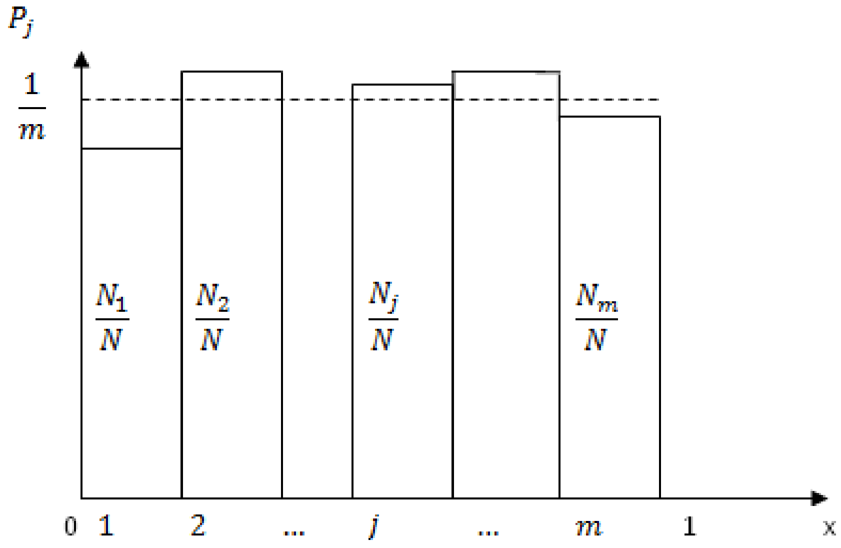


Figure 3.5 – Bar graph for uniformity sequence checkout

The hypothesis about the uniform numbers distribution of sequence $\{x_i\}$ within interval (0,1) is suggested. If the hypothesis is true, each of the numbers x_i with probability $p_j = \frac{1}{m}$ gets on the j-th subinterval ($j \in \overline{1,m}$).

For validation of the advanced hypothesis the numbers of sequence $\{x_i\}$ are distributed on subintervals $j \in \overline{1,m}$. Let N_j be the amount of numbers from sequence $\{x_i\}$ which got on subinterval $j \in \overline{1,m}$, and $N = \sum_{j \in \overline{1,m}} N_j$ be the total amount of sequence $\{x_i\}$ numbers. Relation N_j / N determines the relative frequency of getting of numbers out of sequence $\{x_i\}$ on subinterval $j \in \overline{1,m}$. This relation is a bar graph ordinate on the j-th subinterval, as is shown in fig. 3.5.

If the hypothesis about the uniformity of distribution is true, then ordinates N_j / N ($j \in \overline{1,m}$) of the experimental bar graph will be close to value $1/m$. The quantitative assessment of the degree of closeness may be obtained by means of one of the goodness-of-fit criteria – like Student, Pearson's and others.

Stochasticity test is often realized by the combinations method which is reduced to determining the distribution law of appearing of the amount of units (zeros) in n-bit binary representation of number x_i of sequence $\{x_i\}$. In practice not all n digits of number x_i are considered, but those limited only to the higher-order l digits.

According to the theory, the distribution law of appearing *j* units in *l* higher-order digits of dyadic number x_i on condition of individual digits independence is the binomial distribution law, that is

$$p(j,l) = C_l^j p^j(1)[1-p(1)]^{l-j},$$

where *p*(*j*,*l*) – is the probability of appearance of *j* units in *l* digits;

$C_l^j = \dfrac{l!}{j!(l-j)!}$ – is the number of possible combinations;

$p(1)$ – is the probability of appearance of the unit in one order.

As the probability of appearance of the unit in one digit is equal to the occurrence probability of zero, that is

$$p(1) = p(0) = [1-p(1)] = 0{,}5,$$

then the binomial distribution law for the case under consideration takes the form:

$$p(j,l) = C_l^j p^l(1).$$

The result of the experiment is obtaining the sequence of *N* numbers $\{x_i\}$. Let there be n_j numbers in this sequence with *j* units in *l* higher-order digits of numbers x_i. Then relation $\dfrac{n_j}{N}$ may be considered as the occurrence frequency of numbers x_i with *j* units in *l* higher-order digits.

The hypothesis about closeness of values $p(j,l)$ and $\dfrac{n_j}{N}$ at various values $l \le n$ is advanced. The hypothesis validity is tested by means of goodness-of-fit principles. If the hypothesis is valid, the stochasticity of sequence $\{x_i\}$ is considered valid.

The independence check of the sequence $\{x_i\}$ numbers is realized by determining the values of correlative moments Rx_kx_l, where $x_k, x_l \in \{x_i\}$, $x_k \ne x_l$. For independent random variables x_k, x_l $Rx_kx_l = 0$ [19].

An important characteristic of software generators is the aperiodicity segment length *L*. Within the aperiodicity segment the numbers of sequence $\{x_i\}$ do not recur. The segment length is determined by the amount of non-recurrent numbers.

For example, if to continue sequence $\{x_i\}$ of type (3.3) it will take the form:

$\{x_i\}$= {1,3,9,5.4,1,3,9,5,4,1,3,9,5,4, …}.

In this sequence a segment of five non-recurrent numbers 1,3,9,5,4 is selected. Therefore, the aperiodicity segment length in the given sequence $\{x_i\}$ is $L=5$.

Length L is determined by values x_0, M, that is $L=L$ (x_0, M) for the multiplicative method. For the mixed method $L=L$ $(x_0, M,)$.

Qualitative singularity of the aperiodicity segment length is that the usage of the sample out of sequence $\{x_i\}$ which length is more than L value, does not lead to new statistical results, and only demands additional time for carrying out experiments.

3.3. Modeling of random influences

The main purpose of the statistical modeling method is enabling to study systems of different nature taking into account random influences. Random influences include stochastic events, discrete and continuous random variables, random vectors, functions and processes. To consider all these types of random influences is possible by their simulation/modeling, for the realization of which we need the sequence of pseudo-random quasi-uniformly distributed numbers $\{x_i\}$ within interval (0,1), which is the base sequence of random numbers.

We will start modeling of random influences by modeling of individual stochastic events. Let it be necessary to simulate the realization of a separate stochastic event A, set by probability $P(A)$.

The modeling procedure consists in selecting out of the base random sequence $\{x_i\}$ some value x_i'.

If it appears, that $0 < x_i' \le P(A)$, then the event A will take place. Otherwise, when $x_i' > P(A)$ an event $\overline{A}$ will occur, that is the event A will not take place.

Modeling of a random event A_j entering into the complete group of events A_1 A_2 ... A_j ... A_m, set by probabilities $P(A_1)$ $P(A_2)$ ... $P(A_j)$ ... $P(A_m)$, such, that

$$\sum_{j \in \overline{1,m}} P\left(A_j\right) = 1,$$

is reduced to the realization of the following operations.

Out of the base sequence $\{x_i\}$ some value x_i' is selected.

If $0 < x_i' \le P(A_1)$ an event A_1 will occur.

If $P\left(A_1\right) < x_i' \le P(A_1) + P\left(A_2\right)$ an event A_2 will occur.

………………………………………………………………..

If $\sum_{k \in \overline{1,j-1}} P\left(A_k\right) < x_i' \le \sum_{k \in \overline{1,j}} P\left(A_k\right)$ an event A_j will occur.

………………………………………………………………..

If $\sum\limits_{k\in\overline{1,m-1}} P\left(A_m\right) < x_i' \le 1$ an event A_m will occur.

For a complex event modeling which depends on two or more simple events, set by appearance probabilities, at first it is necessary to determine the possible variants of the complex event and the probability of their appearance.

For example, let the complex event depend on two simple events *A* and *B* independent on each other with the appearance probabilities *P*(*A*), *P*(*B*). The possible variants of the complex event in this case may be:

1) occurrence of events *A* and *B* – *AB*;

2) event *A* does not occur and event *B* occurs – $\overline{A}B$;

3) event A occurs and event B does not occur – $A\overline{B}$;

1. 4) neither A nor B occur – $\overline{A}\overline{B}$.

The probabilities of the pointed variants of the complex event are determined:

P_1= P(A) P(B);

$P_2 = P(\overline{A})\, P(B) = (1–P(A))\, P(B)$;

P_3= P(A) P($\overline{B}$) = P(A) (1–P(B));

P_4= P($\overline{A}$) P($\overline{B}$) = (1–P(A)) (1–P(B)).

Now the modeling procedure of the complex event variants may be realized. Out of sequence $\{x_i\}$ value x_i' is selected:

If $0 < x_i' \le p_1$, then variant *AB* appears.

If $p_1 < x_i' \le p_1 + p_2$, then variant $\overline{A}B$ appears.

If $p_1 + p_2 < x_i' \le p_1 + p_2 + p_3$, then variant $A\overline{B}$ appears.

If $p_1 + p_2 + p_3 < x_i' \le p_1 + p_2 + p_3 + p_4$, then variant $\overline{A}\overline{B}$ appears.

Let's note, that $p_1 + p_2 + p_3 + p_4 = 1$. It guarantees the procedure finiteness and unambiguity of appearance of this or that variant of the complex event.

The modeling procedure of the complex event changes in case when events *A* and *B* are dependent. In this case it is necessary to know the values of conditional probabilities. For example, *P(A/B)*, that is the occurrence probability of event *A* is conditioned by the occurrence of event *B*.

The modeling procedure in this case may be the following. Let the values of occurrence probability of event *A* – *P(A)* and conditional probabilities *P(B/A)*, *P(B/*$\overline{A}$*)* are known. Out of the base sequence $\{x_i\}$ two values x_l' and x_k' are selected.

If it appears, that $0 < x_l' \le P(A)$, then event *A* has occurred. If, more over, it appears, that $0 < x_k' \le P(B/A)$, event *B* occurs. On the whole, there appears the variant of the complex event *AB*.

If it appears, that $0 < x_l' \le P(A)$ and $P(B/A) < x_k' \le 1$, variant $A\overline{B}$ occurs.

If it appears, that $P(A) < x_l' \le 1$ and $0 < x_k' \le P(B/\overline{A})$, variant $\overline{A}B$ occurs.

And, finally, if it appears, that $P(A) < x_l' \le 1$ and $P(B/\overline{A}) < x_k' \le 1$ variant $\overline{A}\overline{B}$ occurs.

The discrete-stochastic modeling (subsection 2.5) is realized on the basis of the concept of a probabilistic automaton, in the determination of which a random transition matrix is used. This matrix is similar to the transition matrix of the homogeneous Markovian chain:

$$P = \begin{Vmatrix} P_{11} & P_{12} & \dots & P_{1k} \\ P_{21} & P_{22} & \dots & P_{2k} \\ \dots & \dots & \dots & \dots \\ P_{k1} & P_{k2} & \dots & P_{kk} \end{Vmatrix},$$

where P_{ij} – is the transition probability from state z_i into state z_j.

At that $\forall i \in \overline{1,k} \quad \sum_{j \in \overline{1,k}} P_{ij} = 1$.

For the homogeneous Markovian chain modeling with a transition matrix P it is possible to make use of the toss procedure which is applied for the realization of the random transition automaton functioning or, in a more general case, for realization of the functioning of the probabilistic automaton [6].

At the first step of the modeling procedure the initial state z_0, set by probabilities $p_1(0), p_2(0), \dots, p_i(0), \dots, p_k(0)$ is determined, where $p_i(0)\ (i \in \overline{1,k})$ is the probability that the initial state will be the state z_i.

It is supposed, that the Markovian process is ergodic. It means, that the limiting probability distribution representing values of probabilities $p_i(n)\ (i \in \overline{1,k})$ of determining state $z_i\ (i \in \overline{1,k})$ through n transitions does not depend on the initial state, that is on value $p_i(0)\ (i \in \overline{1,k})$. Therefore, the values $p_i(0)\ (i \in \overline{1,k})$ can be considered such, that:

$$p_1(0) = p_2(0) = \dots = p_k(0) = \frac{1}{k}. \tag{3.4}$$

To determine the initial state z_0 let us make use of (3.4) and from the base sequence $\{x_i\}$ let us select some value x_i'.

If $0 < x_i' \le \frac{1}{k}$, then $z_0 = z_1$, that is the first state is selected as the initial state. If $\frac{1}{k} < x_i' \le \frac{2}{k}$, then $z_0 = z_2$. And so on. Finally, if $\frac{k-1}{k} < x_i' \le \frac{1}{k}$, then $z_0 = z_k$.

For definiteness, let it turned out, that

$$\frac{l-1}{k} < x_i' \le \frac{l}{k},$$

that is $z_0 = z_l$.

The selection of the state z_l permits for a string index $i=l$ to be fixed in the transition matrix P.

The following step of the modeling procedure is connected with the determination of the state in which the Markovian process will transform from state z_l. For this purpose let us make use of the probability distribution in the l-st row of the transition matrix P:

$$p_{l1}\ p_{l2}\ p_{l3} \dots p_{lj} \dots p_{lk}.$$

From the base sequence $\{x_i\}$ some value x_i'' is selected.

If $0 < x_i'' \le p_{l1}$, the process from state z_l will transform into state z_1.

If $p_{l1} < x_i'' \le p_{l1} + p_{l2}$, the process from state z_l will transform into state z_2.

And so on. In general, if

$$\sum_{j \in \overline{1,m-1}} p_{lj} < x_i'' \le \sum_{j \in \overline{1,m}} p_{lj},$$

the process will transform from state z_l into state z_m.

The definiteness of the selection of state z_j in which the process will transform from state z_l, is guaranteed by the fact that the amount of probabilities in each row of the transition matrix is equal to 1.

At the following step of the modeling procedure the selected state z_m determines a new index $(i = m)$ of the row of the transition matrix P in which the corresponding probability distribution of transitions is reflected. This distribution is used like at the previous step, for selecting the next process state. And so on, the procedure is repeated the necessary number of times corresponding to the number of transitions n.

Modeling of random influences in the form of random discrete and continuous quantities is realized according to the inverse function method by transforming values of random variable ξ from base sequence $\{x_i\}$ into values $\{y_i\}$ of random variable η which integral distribution law $F_\eta(y)$ is preset. Nominally this transformation is represented as

$$\eta = F_\eta^-(y), \qquad (3.5)$$

where F_η^- – is a function, inverse to F_η.

Graphically the essence of transformation (3.5) is shown in fig. 3.6.

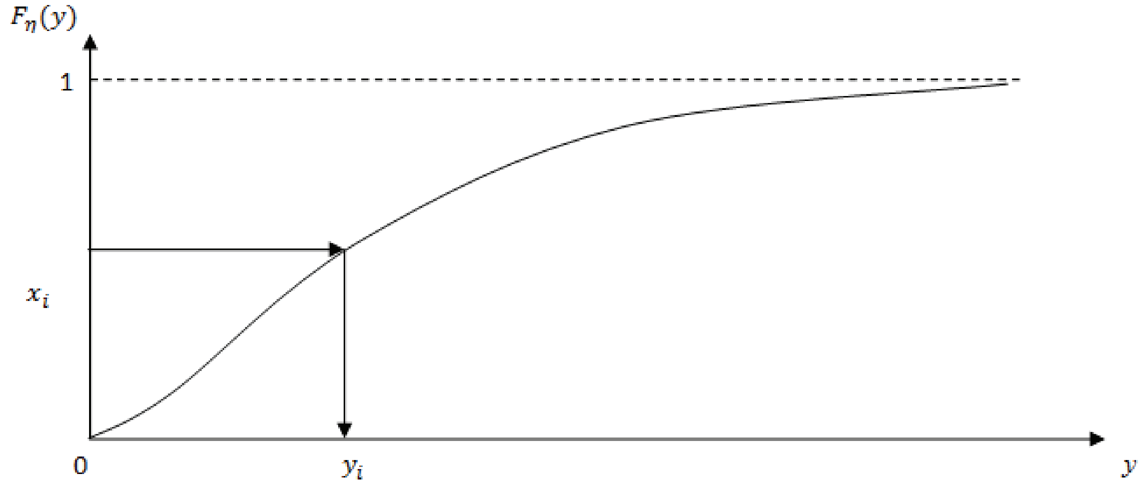


Figure 3.6 – Graphic interpretation of the inverse function method

If ξ is uniformly distributed random variable within interval (0,1), the values of random variable ξ, belong to the base sequence $\{x_i\}$. From sequence $\{x_i\}$ is selected a certain value x_i which is considered as the value of function $F_\eta(y)$ within interval (0,1).

The inverse problem, consisting in determining the value of argument by the preset function value, is solved. In fig. 3.6 it is shown, how by the value of function $F_\eta(y)$, equal to value x_i, the value of argument y_i is determined.

Let the integral distribution law of a discrete random variable η, which possesses values $0 < y_1 < y_2 < ... < y_m < ...$ with probability $p_1, p_2, ..., p_m, ...$ is determined as

$$F_\eta(y_m) = P(\eta \le y_m) = \sum_{i \in \overline{1,m}} p_i \quad (m = 1,2,...)$$
.

At that $F_\eta(y_1) = 0$.

The graph of distribution law $F_\eta(y_m)$ is shown in fig. 3.7.

The value of a discrete random variable $\eta = y_m$ as the value of the argument of the distribution law $F_\eta(y_m)$, depends on what interval of the values of the total amount of probabilities p_i $(\sum p_i)$ gets value x_i from the base sequence $\{x_i\}$, considered as the value of function $F_\eta(y_m)$.

In particular, in fig. 3.7 the situation is shown when value $x_i = x_j$ satisfies the relation

$$\sum_{i \in \overline{1,j-1}} p_i < x_j \le \sum_{i \in \overline{1,j}} p_i \, .$$

Therefore, according to the inverse function method, the appropriate argument value of function $F_\eta(y_m)$ will be $y_m = y_j$.

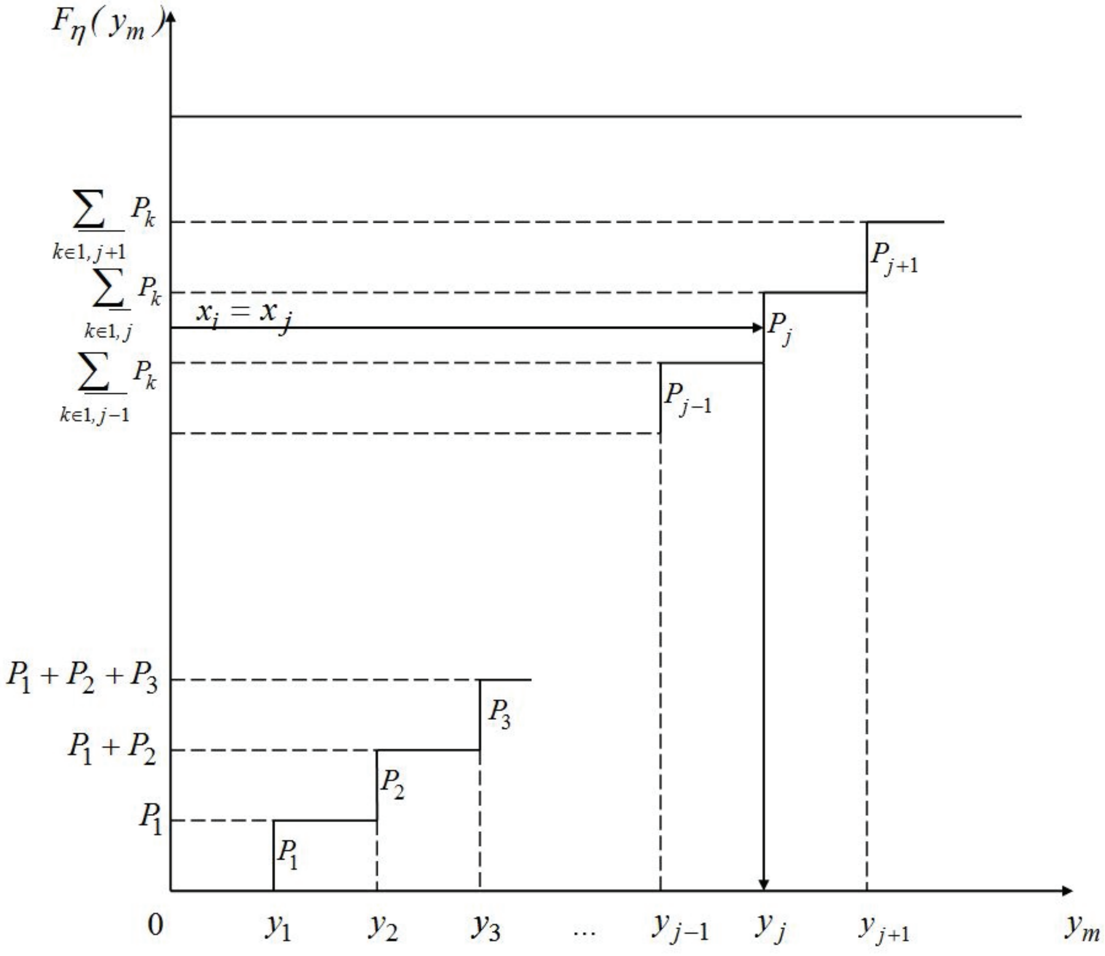


Figure 3.7 – The example of the integral distribution law of a discrete random variable

As an example of the realization of the inverse function method for obtaining sequence $\{y_i\}$ of the values of discrete random variable η let us consider the case when a random variable follows the binomial distribution law which sets the probability y of successful outcomes in N experiments:

$$P_N(y) = C_N^y p^y (1-p)^{N-y} = \frac{N!}{y!(N-y)!} p^y (1-p)^{N-y},$$

where p is the probability of successful outcome in one experiment.

For definiteness let p=0,5, N=6.

Let's determine at first, what values are possessed by the considered discrete random variable and a probability distribution of their possessing.

The possible values are:

y_1=0, y_2=1, y_3=2, y_4=3, y_5=4, y_6=5, y_7=6,

that is, in six experiments *(N*=6*)* it is possible to have zero, one and so on successful outcomes.

For determining the probability distribution of the possession of the indicated values of the discrete random variable we will make use of the expression of the distribution binomial law:

$$P_N(y_1)=\frac{N!}{y_1!(N-y_1)!}p^{y_1}(1-p)^{N-y_1}=P_6(0)=\frac{6!}{0!(6-0)!}0.5^0(1-0.5)^{6-0}=0.01562;$$

$P_N(y_2)=P_6(1)=0.09375$, $P_N(y_3)=P_6(2)=0.23438$, $P_N(y_4)=P_6(3)=0.31250$;
$P_N(y_5)=P_6(4)=0.23438$, $P_N(y_6)=P_6(5)=0.09375$, $P_N(y_7)=P_6(6)=0.01562$.

The obtained value of discrete random variable $\{y_i\}$ and probability distributions of their occurrence permits to diagram the integral binomial distribution law of the considered case which is shown in fig. 3.8.

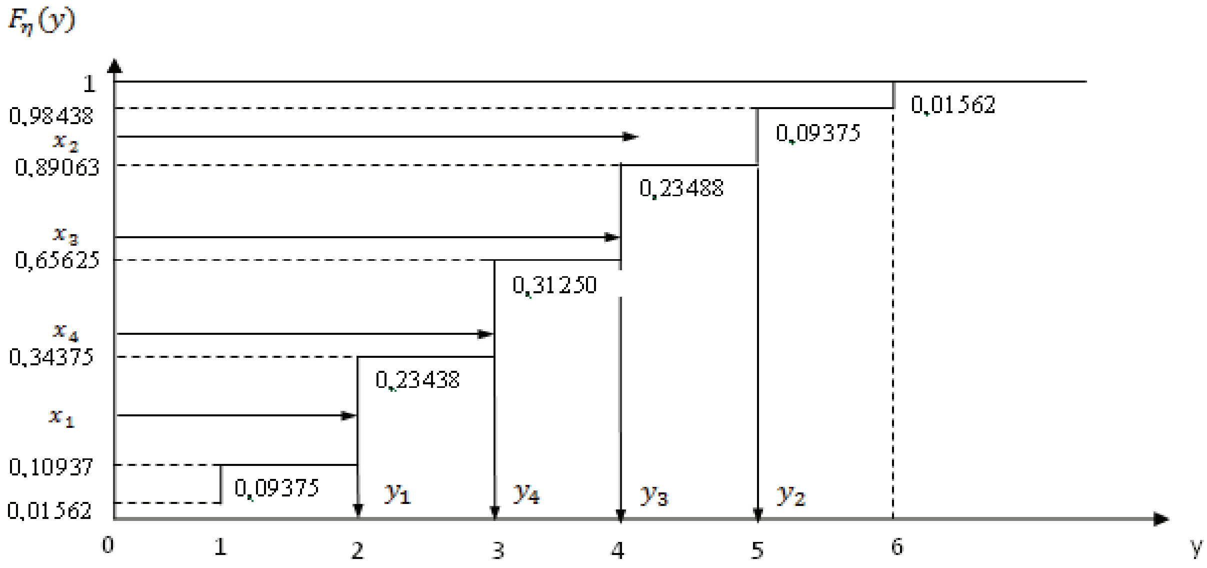


Figure 3.8 – The graph of the integral binomial distribution law and its usage for sequence $\{y_i\}$

The following essential values of $F_\eta(y_m)$ are shown on the graph:

$F_\eta(y_1) = P_N(y_1) = p_1 = 0{,}01562$,

$F_\eta(y_2) = P_N(y_1) + P_N(y_2) = p_1 + p_2 = 0{,}10937$,

$F_\eta(y_3) = P_N(y_1) + P_N(y_2) + P_N(y_3) = p_1 + p_2 + p_3 = 0{,}34375$,

$F_\eta(y_4) = P_N(y_1) + P_N(y_2) + P_N(y_3) + P_N(y_4) = p_1 + p_2 + p_3 + p_4 = \quad = 0{,}65625$,

$F_\eta(y_5) = P_N(y_4) + P_N(y_2) + P_N(y_3) + P_N(y_4) + P_N(y_5) = p_1 + p_2 + p_3 + + p_4 + p_5 = 0{,}89063$,

$F_\eta(y_6) = P_N(y_4) + P_N(y_2) + P_N(y_3) + P_N(y_4) + P_N(y_5) + P_N(y_6) = p_1 + + p_2 + p_3 + p_4 + + p_5 + p_6 = 0{,}98438$.

The indicated values $F_\eta(y_1)$, $F_\eta(y_2)$, ..., $F_\eta(y_6)$ defined as the corresponding totals of probabilities, are used for forming sequence $\{y_i\}$ on the basis of the base sequence $\{x_i\}$.

Let $\{x_i\}$= {0.19, 0.94, 0.69, 0.44, …}.

Value x_1=0.19 satisfies the relation

$$0{,}10937 < 0{,}19 \le 0{,}34375, \text{ that is}$$

$$p_1 + p_2 < x_1 \le p_1 + p_2 + p_3.$$

Therefore, as follows from graph 3.8, value y_1=2 corresponds to value x_1=0.19.

Value x_2=0.94 satisfies the relation:

$$0{,}89063 < 0{,}94 \le 0{,}98438,$$

That is $p_1 + p_2 + p_3 + p_4 + p_5 < x_2 \le p_1 + p_2 + p_3 + p_4 + p_5 + p_6.$

Therefore, as follows from graph 3.8, value y_2=5 corresponds to value x_1=0.94.

Reasoning similarly, we show, that value y_3=4 corresponds to value x_3=0.69, and value y_4=3 corresponds to value x_4=0.44.

On the whole the base sequence $\{x_i\}$ = {0.19, 0.94, 0.69, 0.44, …} corresponds to sequence of values $\{y_i\}$ = {2, 5, 4, 3, …} of the discrete random variable y, following the binomial distribution law.

In addition to the integral distribution law the differential distribution law also is used for random variables modeling.

Let random variable η follow the differential distribution law $f_\eta(y)$.

Random variable ξ, defined as

$$\xi = \int_0^\eta f_\eta(y)dy ,$$

is the uniformly distributed on the interval (0,1).

Then for determining value y_i of random variable η, following the differential distribution law $f_\eta(y)$, it is possible to make use of the equation

$$x_i = \int_0^{y_i} f_\eta(y)dy , \tag{3.6}$$

where x_i is some value out of the base sequence $\{x_i\}$.

For example, if it is necessary to obtain values y_i of the random variable η, following the demonstrative distribution law

$$f_\eta(y) = \lambda e^{-\lambda y} \quad (y > 0),$$

then in this case the equation (3.6) takes the form:

$$x_i = \lambda \int_0^{y_i} e^{-\lambda y} dy . \tag{3.7}$$

Having transformed (3.7) we obtain:

$$y_i = -\frac{1}{\lambda} \ln(1 - x_i) , \tag{3.8}$$

where value x_i is out of the base sequence $\{x_i\}$. Taking various values x_i out of this sequence and using the relation (3.8), we form sequence $\{y_i\}$ of the random variable η, following the demonstrative distribution law. The considered method of obtaining sequence $\{y_i\}$ on the basis of the preset differential distribution law $f_\eta(y)$ and the base sequence $\{x_i\}$ is limitedly applicable due to difficulties of the solution of the equations of type (3.6).

In practice are often used some approximate methods of obtaining sequences of random variables values following the preset distribution law. Among these methods the universal methods play important role. Their universal character consists in being independent on the distribution law type.

According to the first universal method, the differential distribution law $f_\eta(y)$ within the modification interval of random variable values *(a, b)* is represented in the form of a piecewise constant function.

For this purpose interval *(a, b)* is divided into *m* subintervals (a_{k-1}, a_k), where $k \in \overline{1,m}$, $a_0 = a$, $a_m = b$, on each of which $f_\eta(y)$ has a constant value, as is shown in fig. 3.9.

Then random variable η may be presented as:

$$\eta = a_{k-1} + \eta_k ,$$

where η_k – is a random variable uniformly distributed within interval (a_{k-1}, a_k).

The determination of value y_i which is included into the formed sequence $\{y_i\}$, is realized in two stages.

At the first stage subinterval (a_{k-1}, a_k) to which value y_i belongs is determined, and at the second stage value y_i itself is determined from this subinterval.

The realization of the first stage is based on the fact that the probability of getting random variable η within any of subintervals (a_{k-1}, a_k) $k \in \overline{1,m}$ was constant and did not depend on *k*. It is ensured with the fact that

$$\forall k \in \overline{1,m} \quad \int_{a_{k-1}}^{a_k} f_\eta(y) dy = \frac{1}{m} . \tag{3.9}$$

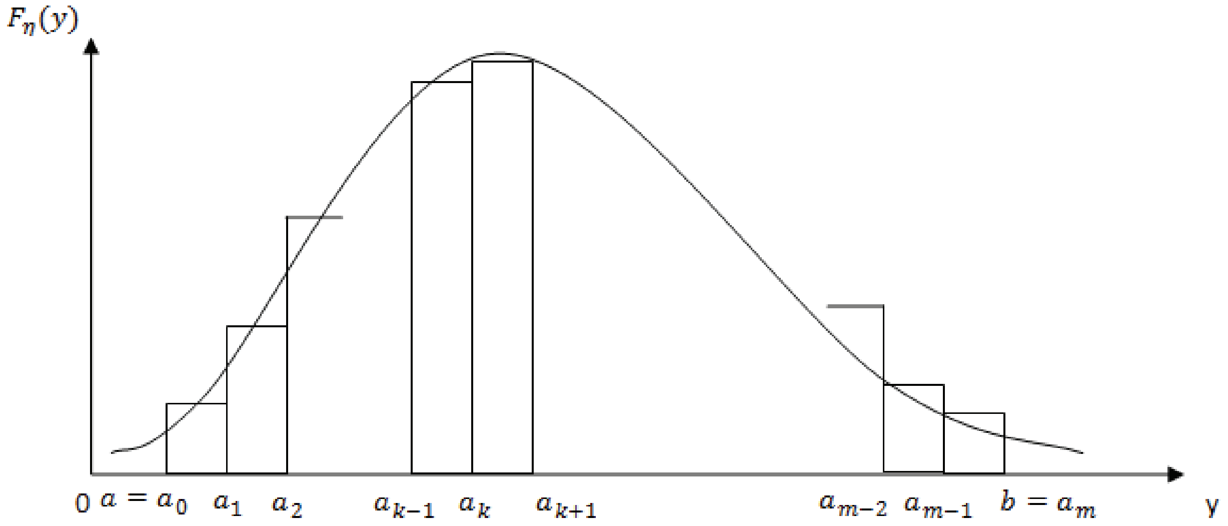


Figure 3.9 – Representation of $f_\eta(y)$ in the form of a piecewise constant function within interval (a, b)

Equations (3.9) are the equations with one unknown, the consecutive solution of which, starting with $k=1$ $(a_0 = a)$, permits to obtain the values of boundaries of all m subintervals (a_{k-1}, a_k) $k \in \overline{1,m}$.

To determine, to what subinterval (a_{k-1}, a_k) value y_i belongs, a toss procedure is used. From the base sequence $\{x_i\}$ some value x_i' is selected.

If $0 < x_i' \le \frac{1}{m}$, then $y_i \in (a_0, a_1)$.

If $\frac{1}{m} < x_i' \le \frac{2}{m}$, then $y_i \in (a_1, a_2)$.

And so on. Finally, if

$$\frac{m-1}{m} < x_i' \le 1, \text{ then } y_i \in (a_{m-1}, a_m),$$

where $a_m = b$.

In general terms, if $\frac{k-1}{m} < x_i' \le \frac{k}{m}$, then $y_i \in (a_{k-1}, a_k)$, $k \in \overline{1,m}$.

For the realization of the second stage one more concrete value x_i'' is selected from the base sequence $\{x_i\}$. If at the first stage interval (a_{k-1}, a_k) was determined, then value y_i is determined on the basis of expression

$$y_i = a_{k-1} + (a_k - a_{k-1})x_i''.$$

Obtained value y_i is included in sequence $\{y_i\}$ under formation. Stages 1, 2 are repeated as many times, as many members of sequence $\{y_i\}$ it is necessary to obtain.

The second universal approximate method is the elimination method. According to this method, all the curve $f_\eta(y)$ is enclosed by a rectangle with side lengths *(a, b)* on a *Y*-axis and *(0, f*)*, where $f^* = \max\limits_{y\in(a,b)} f_\eta(y)$, on an axis $f_\eta(y)$ as it is shown in fig. 3.10.

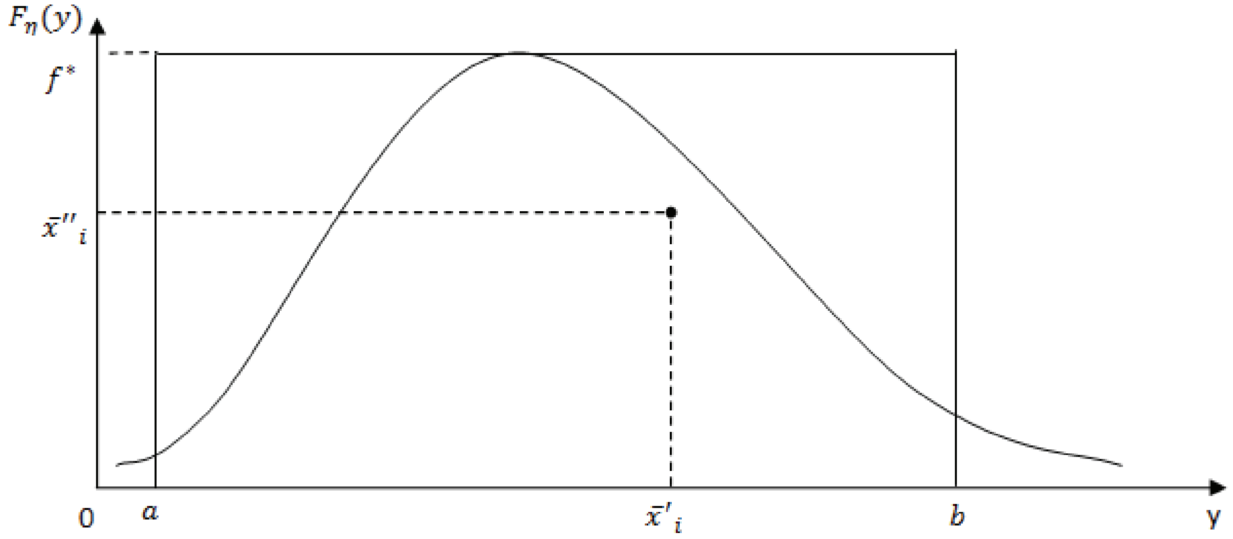


Figure 3.10 – Approximation of $f_\eta(y)$ by a rectangle with lengths *(a, b)* and $(0, f^*)$

Further, the following algorithm is realized.

Step 1. From the base sequence $\{x_i\}$ two values x_i' and x_i'' which are connected with the side lengths (*a, b*) and $(0, f^*)$, are selected

$$\overline{x}_i' = a + (b-a)x_i',$$

$$\overline{x}_i'' = 0 + (f^* - 0)x_i'' = f^* x_i''.$$

Step 2. If it appears, that $x_i'' \le f_\eta(\overline{x}_i')$, that is the point with coordinates $(\overline{x}_i', \overline{x}_i'')$ appears under curve $f_\eta(y)$, as is shown in fig. 3.10 it is supposed that $y_i = \overline{x}_i'$ and the found value is included in the constituted sequence $\{y_i\}$.

Otherwise, when $\overline{x}_i'' > f_\eta(\overline{x}_i')$, that is the point with coordinates $(\overline{x}_i', \overline{x}_i'')$ appears above curve $f_\eta(y)$, then value y_i is considered indeterminate, and the point $(\overline{x}_i', \overline{x}_i'')$ is excluded from the consideration.

Return to the beginning of the first step.

The operations of the mentioned steps are repeated until the sequence of values $\{y_i\}$ of the necessary length is created.

The number of realization of the algorithm steps essentially depends on the type of function $f_\eta(y)$. It is obvious, that for the function shown in figure 3.11, more realizations will be necessary for formation of the sequence of the preset length $\{y_i\}$, than for function $f_\eta(y)$ in fig. 3.10, as the area under the curve in fig. 3.10 is much bigger than the area under the curve in fig. 3.11 if the area of the rectangle enclosing these curves is invariable.

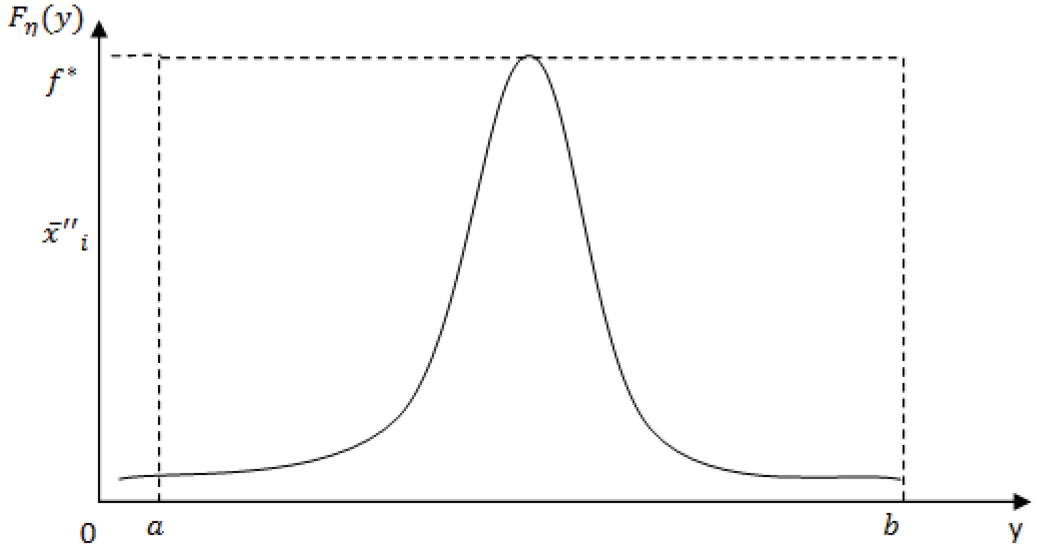


Figure 3.11 – The view of some function $f_\eta(y)$

The elimination method may be used for a random vector modeling, that is for the vector whose components represent the system of random variables with the preset joint distribution law.

For example, let be considered random vector $\eta = (\eta_1, \eta_2)$, whose components are random variables η_1, η_2 with the preset joint differential distribution law (joint probability density) $f_{\eta_1,\eta_2}(y_1, y_2)$.

The elimination method may be used for a random vector modeling, that is for the vector whose components represent the system of random variables with the preset joint distribution law.

For example, let be considered random vector $\eta = (\eta_1, \eta_2)$, whose components are random variables η_1, η_2 with the preset joint differential distribution law (joint probability density) $f_{\eta_1,\eta_2}(y_1, y_2)$.

Modeling of vector η consists in obtaining sequences $\{y_i\} = \{(y_{1i}, y_{2i})\}$ of the preset length on the basis of the base sequence $\{x_i\}$.

Modeling is realized by performing the following algorithm.

Step 1. From the base sequence $\{x_i\}$ there selected three values x_i, x_i', x_i'' which are transformed into values

$$\bar{x}_i = a_1 + (b_1 - a_1)x_i;$$

$$\bar{x}_i' = a_2 + (b_2 - a_2)x_i';$$

$$\bar{x}_i'' = f^* x_i'',$$

where $f^* = \max\limits_{y_1 y_2} f_{\eta_1,\eta_2}(y_1, y_2)$, $y_1 = (a_1, b_1)$, $y_2 = (a_2, b_2)$.

Step 2. If the requirement $\bar{x}_i'' \le f_{\eta_1,\eta_2}(\bar{x}_i, \bar{x}_i')$ is satisfied, that means getting of a random point with coordinates $(\bar{x}_i, \bar{x}_i', \bar{x}_i'')$ under the surface $f_{\eta_1,\eta_2}(y_1, y_2)$, then $y_{1i} = \bar{x}_i$, $y_{2i} = \bar{x}_i'$ and the obtained realization of vector $y_i = (y_{1i}, y_{2i})$ is included into the created sequence $\{y_i\} = \{(y_{1i}, y_{2i})\}$.

Then we return to the beginning of the first step and repeat the mentioned operations until the sequence $\{y_i\}$ of the preset length is created.

The computational procedure of obtaining realization of random high dimension vectors by the elimination method is so tedious, that it is seldom used in practice. In this case various special methods based on calculating some numerical characteristics estimates are used.

Summary

The statistical modeling is widely used for researching stochastic objects which are described by means of such mathematical schemes as probabilistic automata and queue systems. This chapter considers the main directions of usage of statistical modeling method. Different methods of pseudorandom numbers generation are presented in this chapter. These methods are often used in statistical modeling. Additional attention is paid to modeling of random influences.

Tasks for case-study

1. Construct the formula of standard modeling method for random variable with distribution density $ce^{-\sigma x}$; $0 \le x \le a$; $\sigma > 0$.
2. Construct the formula of standard modeling method for random variable with distribution density $c(x+1)^{0.75}$; $0 \le x \le 1$.

3. Construct the formula of standard modeling method for random variable with distribution density $c|\sin x|$; $-\frac{\pi}{2} \le x \le \frac{\pi}{2}$.
4. Elaborate the modeling algorithm for random variable with distribution density $\sum_{n=0}^{\infty} \frac{(n+1)(\lambda x)^n}{n!} e^{-\lambda}$; $0 \le x \le 1$; $\lambda > 0$.
5. Elaborate the modeling algorithm for random variable with distribution density $\sum_{n=0}^{\infty} (n+1) p(x(1-p))^n$; $0 \le x \le 1$; $0 < p < 1$.
6. Elaborate the modeling algorithm for random variable with distribution density $c\left(x^{-2} + x^2 e^{-x^3}\right)$; $x \ge 1$.
7. Elaborate the modeling algorithm for random variable with distribution density $ce^{-ax}\left(1 + e^{-bx}\right)$; $x \ge 0$; $a, b > 0$.
8. Construct the formula of standard modeling method for random variable with distribution density $ce^{-(x-1)^2}$; $x \ge 1$.
9. Construct the formula of standard modeling method for random variable with distribution density $cx^{10} e^{-\sigma x}$; $x \ge 0$; $\sigma > 0$.
10. Construct the formula of standard modeling method for random variable with distribution density $c(1-x^3)$; $0 \le x \le 1$.
11. Construct the formula of standard modeling method for random variable with distribution density $c(\ln x)^4$; $0 \le x \le 1$.
12. Elaborate the modeling algorithm for random variable with distribution density $cx(1-x)^2$; $0 \le x \le 1$.
13. Elaborate the modeling algorithm for random variable with distribution density $c(1-x)^3$; $0 \le x \le 1$.

Check questions

1. The part of which mathematical schemes is presented by statistical modeling?
2. What are the main directions of usage of statistical modeling method?
3. What ways of random numbers generation do you know?
4. What are the disadvantages of instrumental way of random numbers generation?
5. What is the sense of congruent procedures of pseudorandom numbers generation?

6. Describe the multiplicative method of pseudorandom numbers generation.
7. How are uniformity, stochasticity and independence tests realized for generated pseudorandom quasi-uniformly distributed numbers?
8. How is a separate random event modeled?
9. How is a complex event, which depends on two or more simple events, modeled?
10. How is a homogeneous Markovian chain modeled?
11. What is the sense of the inverse function method for modeling of random influences in the form of random discrete and continuous values?
12. What universal methods of pseudorandom numbers generation do you know?

4. SIMULATION MODELING

4.1. The essence of simulation modeling and the basic cases of its application

As a rule real complex systems are studied by means of two types of a mathematical model – analytical and simulation, as well as on the basis of their combined application (the combined modeling).

In analytical models functioning of the complex system is represented by the system of functional relations which may be supplemented with various logical conditions. Often such system is the system of differential, finite-difference and other types of equations.

The most complete research by means of analytical modeling can be conducted, when it is possible to obtain the explicit dependence between the values characterizing the studied quality of the real system and the system parameters. For obtaining such dependences it is necessary to simplify the phenomena which are observed in real systems.

When the phenomena in the system are so complicated, that their simplification is a too rough approximation to reality, it is necessary to refuse analytical modeling and to apply simulation modeling for studying such systems.

Simulation modeling does not provide for those phenomena simplifications in the real system, which are required for the implementation of analytical modeling. The complex system functioning in the simulation model is represented as a set of algorithms. These algorithms, based on the actual values of parameters and information about the system initial state, permit to reproduce the system functioning in each concrete situation.

Most often simulation modeling is applied in the following cases:

- when simulation modeling appears to be the only possibility of studying the complex system due to the difficulties of phenomena observation in real conditions;
- when it is necessary to study the processes in the complex system by means of their artificial deceleration or acceleration;

- when training specialists for new equipment while the simulation model is a means of acquiring new equipment operation skills;
- when studying new situations in complex systems of which very little or nothing is known and therefore the simulation model is a unique means of verifying strategies and rules of behavior in new situations;
- when special importance is attached to the sequence of events in the complex system and the simulation model is used for detecting possible "bottlenecks" and other difficulties in the studied system functioning when introducing new elements into the system.

The main advantages of simulation modeling include:

- the possibility of describing phenomena and processes in complex systems at the high level of details;
- the absence of limitations for the type of dependences of functional relations used for describing complex systems functioning (nonlinearity, nonstationarity, stochasticity etc.);
- the possibility of studying the interaction dynamics of the system elements and components in parameter space and in time.

However, the simulation modeling also has some disadvantages:

- the development of the complex system simulation model demands, as a rule, quite big resources and time burden;
- the simulation model like any other model is not precise, but at the same time it is difficult or impossible to estimate the extent of its inaccuracy.

The term "simulation modeling" means, that by means of such mathematical models the result is not calculated. The result, which a researcher is interested in, is obtained by means of the simulation modeling on the basis of the model-based experiment (simulation) processing with the set input data. The experiment, therefore, consists in the realization of an algorithm or a set of algorithms representing the target system functioning. Simulation is a numerical method of experimenting on a computer with mathematical models describing the complex system behavior during the preset or formed time period.

4.2. Modeling time and ways of its realization

The construction of a complex system simulation model assumes structured representation of the system in which some components are singled out, that is $S'=\{K_i \mid i \in \overline{1,n}\}$, where K_i is the *i*-th component of the system.

Functioning of component K_i represents the sequence of functional operations ΦD_{ij} $(j \in \overline{1,m_i})$. In other words, $K_i = \{\Phi D_{ij} \mid j \in \overline{1,m_i}\}$, where

$$ \Phi D_{i1} \prec \Phi D_{i2} \prec \ldots \prec \Phi D_{ij} \prec \ldots \prec \Phi D_{im_i} . \qquad (4.1) $$

As a result of each functional operation ΦD_{ij} performance, an event C_{ij} occurs in a complex system.

Any ΦD_{ij} is performed within a certain time interval τ_{ij}. For characterization of the temporal functioning of K_i local time t_i is inducted.

In a complex system all local times t_i $(i \in \overline{1,n})$ change simultaneously, however the nature of changes varies, and it is defined by the sequence of the time intervals τ_{ij} which corresponds to the sequence of functional operations ΦD_{ij}.

When constructing the simulation model of a complex system ΦD_{ij} are approximated by simplified functional operations $\Phi D'_{ij}$. The degree of simplification determines the level of details of the simulation model. The discrepancies between ΦD_{ij} and $\Phi D'_{ij}$ constitute the inaccuracy of the simulation model, the estimation of which, as it was mentioned before, is difficult to obtain.

In the simulation model the real functional operation ΦD_{ij} is represented by a pair $< \Phi D'_{ij}, \tau_{ij} >$ which is fulfilled in the following order.

At first, at a specified value of local time t_i the simplified operation $\Phi D'_{ij}$ is carried out, and then t_i changes by value τ_{ij}. After alteration of t_i by value τ_{ij} the occurrence of event C_{ij} is initiated, which confirms the fact of operation ΦD_{ij} completion in a real complex system.

Performance of all functional operations arranged according to sequence (4.1) in the specified manner in the simulation model determines the implementation order of simulation modeling of each component K_i of the complex system S'. This order is schematically shown in fig. 4.1.

According to the schema in fig. 4.1, in the beginning at $t_i = 0$ the first simplified functional operation $\Phi D'_{i1}$ is carried out. Then value $t_i = 0$ changes by value τ_{i1} and becomes equal to $t_i = \tau_{i1}$. At this instant event C_{i1} occurs, confirming the fact of the real operation ΦD_{i1} performance, which is conditionally shown in fig. 4.1 by the dotted line.

Further at the moment of $t_i = \tau_{i1}$ the simplified operation $\Phi D'_{i2}$ is carried out. After its performance value $t_i = \tau_{i1}$ changes by value τ_{i2} and becomes equal to $t_i = \tau_{i1} + \tau_{i1}$. At this moment event C_{i2} confirming the performance of real operation ΦD_{i2}, is initiated, etc. See the dotted line in figure 4.1.

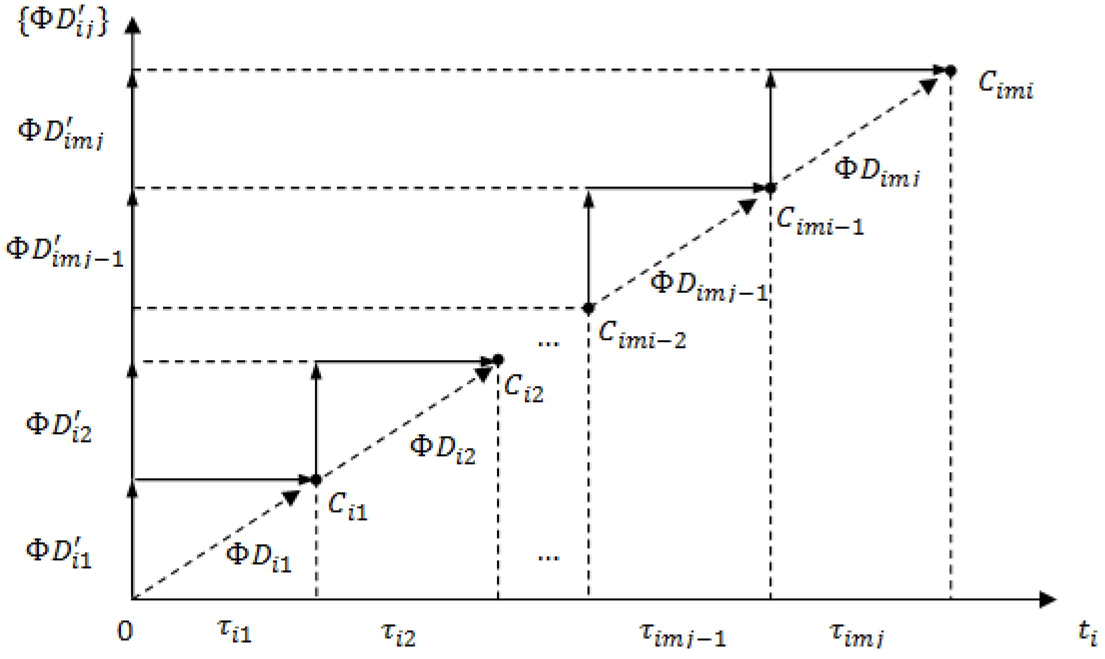


Figure 4.1 – The implementation schema of component K_i simulation model

In the simulation model each simplified operation $\Phi D'_{ij}$ is realized by a corresponding algorithm АЛ_{ij}. Thus, the real functional operation ΦD_{ij} in the simulation model is represented by algorithm АЛ_{ij} and operator Mt_{ij} changing the local temporal co-ordinate t_i by value τ_{ij}.

Pair $< A\text{Л}_{ij}, \tau_{ij} >$ is called the activity of the simulation model and is designated as AK_{ij}. The realization of activity $AK_{ij} =< A\text{Л}_{ij}, \tau_{ij} >$ leads to the appearance of event C_{ij} in the simulation model.

On the whole, the simulation model SM of a complex system can be considered as a set of activities, that is

$$\text{IM} = \left\{ \text{AK}_{ij} \mid i \in \overline{1,n},\ j \in \overline{1,m_i} \right\},$$

taking into account the activities implementation order, set by relations (4.1) for each component K_i.

The components of the complex system $K_i\,(i \in \overline{1,n})$ function simultaneously or, in other words, in parallel. However, in modern computers at each instant the algorithm or activity which refers only to one of the components is realized. Therefore the problem arises how to reflect the parallel functioning of several components of the complex system with the help of the computer.

The specified problem is solved by introducing a certain global variable t_0 which is called simulated or system time.

This variable serves for synchronization of events C_{ij} in the model or, in other words, for the performance of activities AK_{ij} of various components $K_i \, (i \in \overline{1,n})$ of the complex system.

Thus, when implementing a simulation model, three representations of time are used – real time t_r, simulated or system time t_0, machine time t_k which expresses computer timing budgets for simulation performance.

As a matter of fact, simulated time t_0 permits to realize quasi-parallel operation of a complex system components, that is simultaneously emerging events in different components of the system are registered, and then they are serviced sequentially.

The alterations of local co-ordinates of real time t_i on the basis of simulated time t_0 occur according to the following example.

Let there be two components K_1 and K_2, each of which functions in its corresponding local time t_1 and t_2. At some moment of simulated time t'_0 in components K_1 and K_2 simultaneously occur events C_{1j_1} and C_{2j_2}. If t_{1j_1} and t_{2j_2} correspond to the moments of occurrence of these events, it is obvious that $t'_0 = t_{1j_1} = t_{2j_2}$, as it is shown in fig. 4.2.

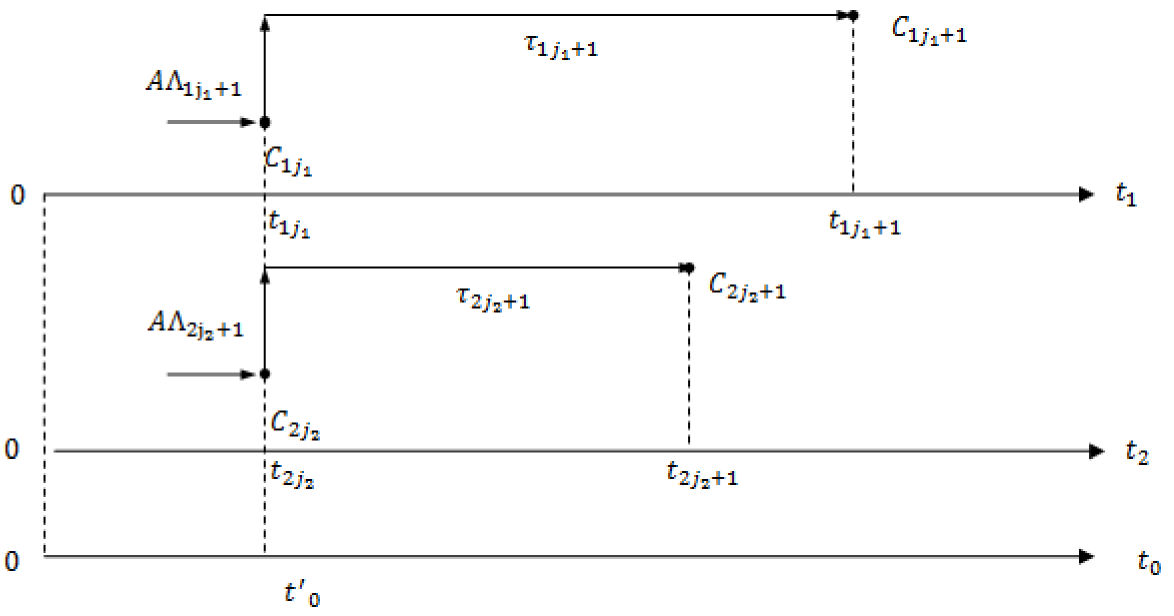


Figure 4.2 – The schema of modifications of t_i on the basis of t_0

According to the specified procedure of the simulation model implementation, at first algorithms АЛ_{1j_1+1}, АЛ_{2j_2+1} are sequentially performed, (or vice versa) at the fixed value of both local time co-ordinates t_i, and simulated time t_0, and then local co-ordinates t_i change by the value of duration of performance of the corresponding operation τ_{ij}. Thus, new values of local time co-ordinates are defined on the basis of t_0 as

$$t_{1j_1+1} = t'_0 + \tau_{1j_1+1};\ t_{2j_2+1} = t'_0 + \tau_{2j_2+1}.$$

The obtained new values t_{1j_1+1}, t_{2j_2+1} are memorized and further used for the subsequent activations of components $K_i\ (i \in \overline{1,2})$ in the simulation model. Activation is understood as the beginning of the following activity performance. In the example reviewed obviously activities AK_{1j_1+1} and AK_{2j_2+1} will be the case. The activation is reduced to performance of the corresponding algorithm and modification of the local time co-ordinate.

After the modification of all local time co-ordinates $t_i\ (i \in \overline{1,n})$ there appears the problem of defining a new value of simulation time t_0. Most often two ways of simulation time modification t_0 are used, each of which has several names. The first method is called the fixed interval method, or the fixed step, or principle Δt. The second method is called the variable interval method, or the step before the following event, or principle δz.

We will illustrate essence of these two methods by the following example.

Let there be three components in a system, that is $S' = \{K_i \mid i \in \overline{1,3}\}$ whose functioning processes are reduced to performance of functional operations. In the first component four functional operations are performed, that is

$$K_1 = \{\Phi D_{1j} \mid j \in \overline{1,4}\},$$

where $\Phi D_{11} \prec \Phi D_{12} \prec \Phi D_{13} \prec \Phi D_{14}$,

and in the second and the third components three functional operations are performed, that is

$$K_2 = \{\Phi D_{2j} \mid j \in \overline{1,3}\},$$

$$\Phi D_{21} \prec \Phi D_{22} \prec \Phi D_{23},$$

$$K_3 = \{\Phi D_{3j} \mid j \in \overline{1,3}\},$$

$$\Phi D_{31} \prec \Phi D_{32} \prec \Phi D_{33}.$$

The durations of performance of all functional operations $\{\tau_{ij}\}$ are known.

Quite a general form of the simulation model of the specified complex system is the set of activities approximating functional operations which are performed in the order similar to the order of functional operations in each component, that is

$$\mathrm{IM} = \{\mathrm{AK}_{ij_i} \mid i \in \overline{1,3},\ j_1 \in \overline{1,4},\ j_2 \in \overline{1,3},\ j_3 \in \overline{1,3}\}$$

$$\mathrm{AK}_{11} \prec \mathrm{AK}_{12} \prec \mathrm{AK}_{13} \prec \mathrm{AK}_{14};$$

$$\mathrm{AK}_{21} \prec \mathrm{AK}_{22} \prec \mathrm{AK}_{23};$$

$$\mathrm{AK}_{31} \prec \mathrm{AK}_{32} \prec \mathrm{AK}_{33}.$$

Each performance of activities AK_{ij} leads to the appearance of the corresponding event C_{ij} in the model.

Taking into account the procedure of the simulation model implementation and the values of the operations performance duration $\{\tau_{ij}\}$ the schema of the modification of the local time co-ordinates t_i and modeling time t_0 is shown in fig. 4.3.

At the same time t'_0 characterizes alteration of modeling time as via the first method, and t''_0 – as via the second one.

When simulating via the first method modeling time t'_0 varies each time by value $\varDelta t$. At the instances of modeling time $0, \varDelta t, 2\varDelta t$, etc in the simulation model events S_0, S_1, S_2, etc take place which may be considered as peculiar signals to the modeling control program (MCP) for servicing events C_{ij} getting into the next interval $\varDelta t$.

For example, when $t'_0 = 0$, event S_0 occurs and MCP services all the events C_{ij} within the interval $(0, \varDelta t)$ or (S_0, S_1). From fig. 4.3 follows, that the events C_{11} and C_{31} fall into the interval (S_0, S_1). It is presumed, that these events occur simultaneously and the moment of their occurrence is assigned to the right end of the interval which they have fallen into, that is by moment $t'_0 = \varDelta t$. MCP sequentially services the events C_{11} and C_{31}, which is reduced to the initialization of the corresponding activities AK_{11} and AK_{31}, that is to the performance of algorithms $AЛ_{11}$, $AЛ_{31}$ and to the time change by values τ_{11}, τ_{31}.

At the following instant of modeling time $t'_0 = \varDelta t$ event S_1 occurs and MCP services event C_{ij} within the interval $(\varDelta t, 2\varDelta t)$ or (S_1, S_2). These are events C_{21}, C_{12}, C_{22} which are considered to be occurring simultaneously at the moment $t'_0 = 2\varDelta t$. MCP sequentially services the events, which is reduced to the initialization of activities AK_{21}, AK_{12}, AK_{22} and so on.

Thus, on the whole the mechanism of the modeling time alternation as per the first method is reduced to the increase of t'_0 by the predetermined value Δt with the subsequent service by MCP of events C_{ij} which get into the next interval Δt. At the same time all these events refer to the right boundary of the modeling time interval.

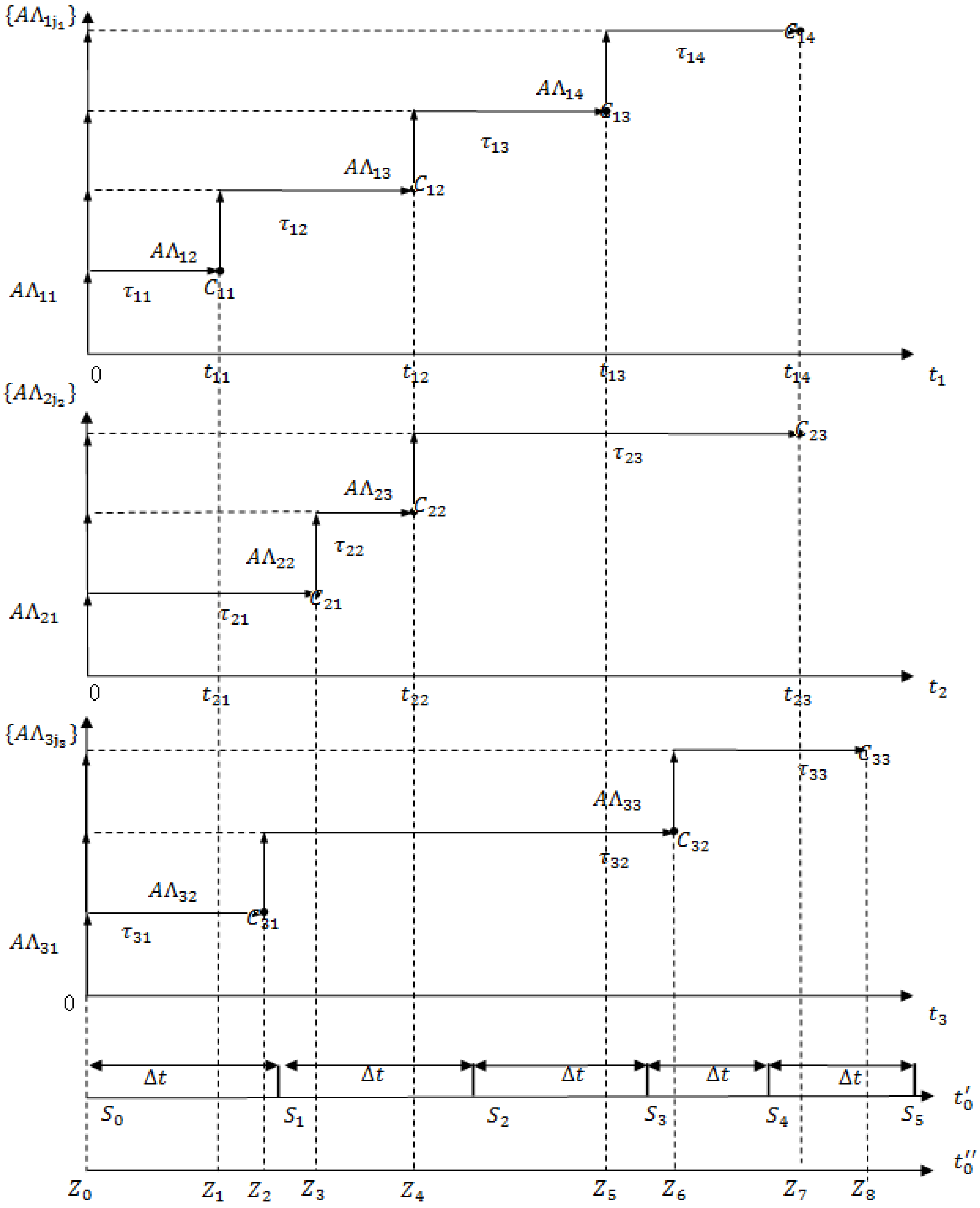


Figure 4.3 – The schema of the simulation model performance taking into account the alteration of the local time co-ordinates t_i and modeling time by two methods t'_0, t''_0

It is obvious, that the accuracy of modeling depends on Δt. It increases with the decrease of Δt, but the number of calls to MCP increases as well, that leads to the increase of consumption of computer machine time t_k required for simulation.

Thus, on the whole the mechanism of the modeling time alternation as per the first method is reduced to the increase of t'_0 by the predetermined value Δt with the subsequent service by MCP of events C_{ij} which get into the next interval Δt. At the same time all these events refer to the right boundary of the modeling time interval.

It is obvious, that the accuracy of modeling depends on Δt. It increases with the decrease of Δt, but the number of calls to MCP increases as well, that leads to the increase of consumption of computer machine time t_k required for simulation.

According to the second method of the modeling time alteration, t''_0 possesses values Z_l corresponding to instances t_{ij} at which events C_{ij}occur. If it appears, that several values t_{ij} correspond to a certain instant t''_0, that is at the moment t''_0 there occur several events C_{ij}, these events are sequentially serviced by MCP without alteration of modeling time. For example, according to fig. 4.3, instant $t''_0 = z_4 = t_{12} = t_{22}$. It means, that at instant $t''_0 = z_4$ there simultaneously occur two events – C_{12} and C_{22}. MCP sequentially services these events which is reduced to the initialization of activities AK_{12} and AK_{22}, that is to the performance of algorithms $AЛ_{12}$, $AЛ_{22}$ and alteration of times t_1, t_2 by values τ_{12}, τ_{22}. Modeling time t''_0 possesses the value of the nearest t_{ij} on the axes of local co-ordinates t_i.

Irrespective of the method of t_0 alteration, the mechanism of modeling time alteration includes the following operations:

- the events which are to be serviced at the given modeling time t_0 are identified;
- the events are serviced by the initialization of appropriate activities;
- the next value of modeling time t_0 is defined and its correction is conducted;
- the condition of the modeling completion is checked. This condition usually consists in the comparison of the current time with a preset time or in the occurrence of the preset events.

All the specified operations are performed by MCP. Servicing of events C_{ij} demands some computer time resources t_k. It is obvious, that the more often the events are serviced, the higher value t_k is. From this point of view the method of the fixed step alteration of modeling time t_0 is more efficient, as there takes place group servicing of all the events C_{ij} which fall into the next step Δt of the modeling time alteration.

In practice the fixed step method is preferred in two cases:

- when events C_{ij} are evenly arranged within entire modeling interval and it is possible to choose Δt in such a way that the minimum simulation error is ensured;
- when events C_{ij} appear in groups and it is possible to choose such value of Δt which covers all the events in a group.

In all other cases the method of the step till the next event is preferable as a more exact and economical one.

4.3. The structural scheme of a simulation model and technological stages of the simulation model construction process

There are four simulation approaches or methods used by the simulation community. They are process interaction, event scheduling, activity scanning and three-phase approaches.

The simulation structure that has the greatest intuitive appeal is the process interaction method (fig. 4.4).

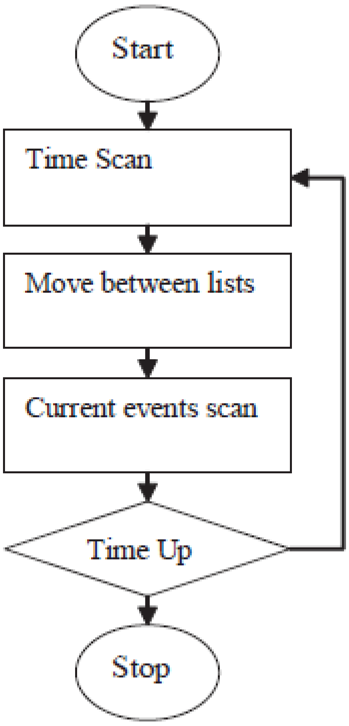


Figure 4.4 – Process interaction flow

In this method, the computer program emulates the flow of an object (for example, a load) through the system. The load moves as far as possible in the system until it is delayed, enters an activity, or exits from the system. When the load's movement is halted, the clock advances to the time of the next movement of any load. This flow describes in sequence all of the states that the object can attain in the system. In a model of a self-service laundry, for example, a customer may enter the system, wait for a washing machine to become available, wash his or her clothes in the washing machine, wait for a basket to become available, unload the washing machine, transport the clothes in the basket to a dryer, wait for a dryer to become available, unload the clothes into a dryer, dry the clothes, unload the dryer, and then leave the laundry. Each state and event is simulated.

Transaction flow approach is a simpler version of process interaction approach. It was first introduced by GPSS in 1962. In transaction flow approach models consist of entities (units of traffic), resources (elements that service entities), and control elements (elements that determine the states of the entities and resources). Discrete simulators, which are generally designed for simulating detailed processes, such as call centers, factory operations, and shipping facilities, rely on such approach.

The next simulation structure is event scheduling method. The basic concept of the event scheduling method is to advance time to the moment when something happens next (fig. 4.5).

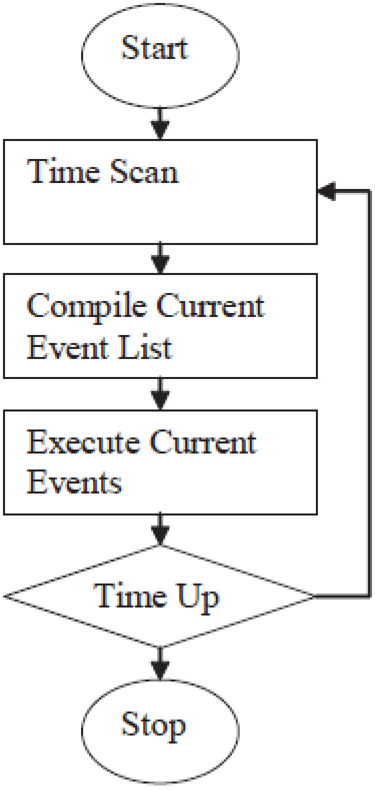


Figure 4.5 – Event scheduling approach

This means that when one event ends, time is advanced to the time of the next scheduled event. An event usually releases a resource. The event then reallocates available objects or entities by scheduling activities, in which they can now participate. Time is advanced to the next scheduled event (usually the end of an activity) and activities are examined to see whether any can now start as a consequence. Event scheduling approach has the following advantage. It does not require any specialized language or operating system support. While the disadvantage of the event scheduling approach is that describing a system as a collection of events obscured any sense of process flow.

The third simulation modeling structure is activity scanning. Activity scanning is also known as the two phase approach (fig. 4.6). Activity scanning produces a simulation program composed of independent modules waiting to be executed. In the first phase, a fixed amount of time is advanced, or scanned. In phase two, the system is updated (if an event occurs).

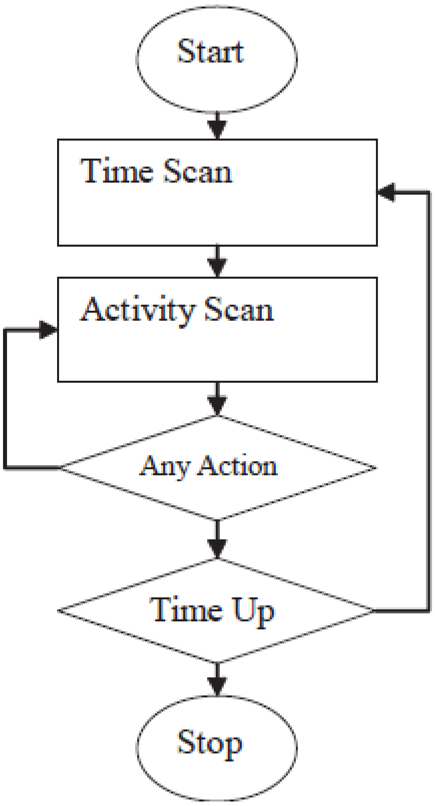


Figure 4.6 – Activity scanning approach

The three-phase approach is the fourth simulation structure (fig. 4.7). In the A phase, time is advanced until there is a state change in the system or until something happens next. The system is examined to determine all of the events that take place at this time. The B phase is the release of those resources scheduled to end their activities at this time. The C phase is to start activities, given a global picture of resource availability.

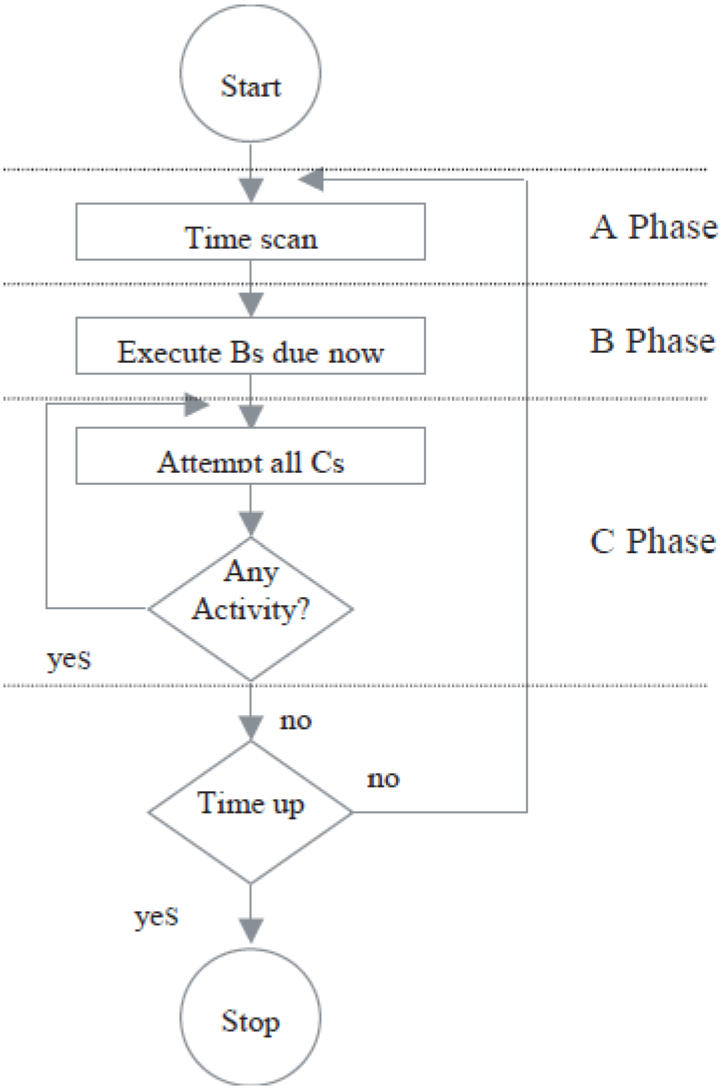


Figure 4.7 – Three-phase method execution

There are some advantages of all of the mentioned approaches. The process interaction method allows to avoid programs that are slow to run. Secondly, it avoids the need to think through all possible logical consequences of an event. Also the implementation of quick programs in this case requires the additional programming skills.

The event scheduling method is simpler than two-phase and three-phase methods and only has two phases so there is no Cs and Bs. This allows the program to run faster since there is no scanning for the conditional events.

The activity scanning approach is also simpler than the three phase method since it has no calendar, and it supports the parsimonious modeling. However this approach is much slower than the three-phase since all activities are treated as conditional.

One of the structural representations of the simulation model is the tuple which includes a number of activities $\{AK_{ij} \mid i \in \overline{1,n},\ j \in \overline{1,m_i}\}$ and MCP, ensuring the activities interaction during simulation and performance of organizational functions, that is

$$\text{IM} = \left\langle \{\text{AK}_{ij} \mid i \in \overline{1,n},\ j \in \overline{1,m_i}\}, \text{КПМ} \right\rangle.$$

Organizational functions of MCP consist in:

- alteration of modeling time t_0;
- performance of operators Mt_{ij} changing the local co-ordinates t_i by value τ_{ij};
- initialization of algorithms $АЛ_{ij}$ performance;
- verification of the conditions of the simulation completion.

In the MCP there exist three basic subprograms (SP) – SP for the simulation start, SP for the statistics collection, SP for the simulation completion.

SP for the simulation start is intended for setting input conditions and initial values of the model parameters.

In the SP for the statistics collection there are registered some modeling statistics characterizing the behavior of component $K_i\ (i \in \overline{1,n})$ of a complex system in different modes of its functioning.

In the SP for the simulation termination the condition of the simulation completion is verified and if it is met, the process of handling of the collected modeling statistics is realized, which permits to obtain some estimations of behavior performances of the system under study and to produce them for a researcher in the required form.

Two parts in the simulation model structure are singled out. One, presented by the set of activities $\{AK_{ij} \mid i \in \overline{1,n},\ j \in \overline{1,m_i}\}$, is defined by the essence of the modeled components $K_i\ (i \in \overline{1,n})$ of a complex system and therefore has variable character. The other, in the form of MCP, has rather a fixed set of elements and constant number of executable functions.

The interaction of the specified parts in the simulation model structure is shown in fig. 4.8.

Fig. 4.8 clearly shows, that each functional operation $ФD_{ij}$ of the component K_i in the simulation model is described by the corresponding activity AK_{ij} which is represented by algorithm $АЛ_{ij}$ and operator Mt_{ij} changing the local co-ordinate t_i by the duration value of operation τ_{ij} performance. From the schema it follows, that the commands for starting algorithms performance come from MCP. Following performance of operators Mt_{ij} the information about new values of events C_{ij} occurrence time comes to

MCP and is used for modification of modeling time t_0 when forming new commands for starting algorithms $АЛ_{ij}$, and verifying conditions of the simulation completion.

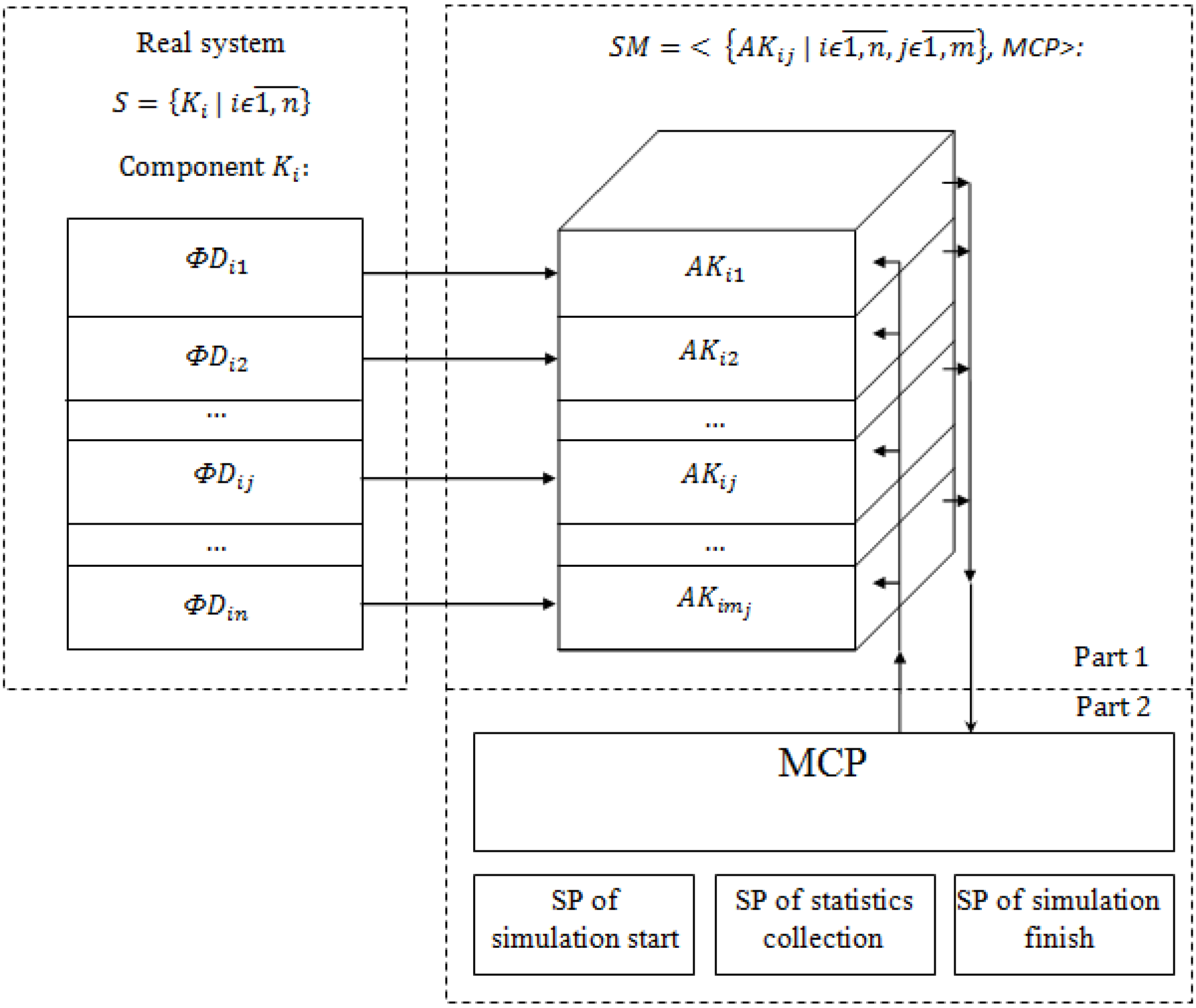


Figure 4.8 – The block diagram of a simulation model

The following interconnected stages are singled out in the course of a simulation model development:

- determining purposes of modeling;
- development of a conceptual model;
- development of the simulation model software-based realization;
- verification of the feasibility of the requirements to the model;
- planning and carrying out experiments;
- the analysis of modeling results.

Modeling goals are determined by the research objectives which, most often, consist in creating a new system or in modernizing the existing system taking the efficiency into account.

The efficiency indicator is the excess of the utility of the offered system variant over the development and operation costs of this variant. However, to estimate the effect in full, as a rule, is not possible, as it is manifested not only directly, as the result of the functioning of the offered system variant, but also indirectly, as the result of other systems functioning with which the system under study is connected.

A simpler notion is the concept of technical and economic efficiency which includes expenditures and measurable system performances or, otherwise, particular indicators of a system quality – productivity, reliability, mass, dimensions etc. These particular indicators are the components of vector Y in the model (1.1), and the indicator of technical and economic efficiency itself is the function from Y

$$E = E(Y). \qquad (4.2)$$

If the efficiency estimate (4.2) is only connected with the optimization by the quality indicator with restriction of other indicators, for example, as

$$E = y_j \rightarrow \min\,(\max);$$

$$y_i \begin{pmatrix} > \\ < \\ = \end{pmatrix} y_i^0, \ (i \in \overline{1,n}, \ i \neq j),$$

where y_j are the components of vector Y, and y_i^0 is their certain quantitative values, this efficiency estimate is called a one-criteria estimate.

A multicriteria estimate is the generalized or integral criterion which links by functional association the efficiency indicator E with the transformed components of vector $Y - y'_j \ (i \in \overline{1,n})$. The transformation of the components permits to reduce to their dimensionless form and to compare the components deviations from their optimal values.

The most widely used criterion is the generalized one in the form of a linear function from the transformed components:

$$E = \sum_{i \in \overline{1,n}} \alpha_i y'_i , \qquad (4.3)$$

where $\forall i \in \overline{1,n} \ \ 0 \leq \alpha_i \leq 1, \ \sum_{i \in \overline{1,n}} \alpha_i = 1$.

Coefficients $\alpha_i (i \in \overline{1,n})$ reflect the importance or "weight" of the corresponding components $y'_i \, (i \in \overline{1,n})$.

Thus, the modeling goals consist, firstly, in determining the runnable system variant whose output performances (parameters) satisfy the preset restrictions

$$Y\begin{pmatrix} < \\ > \\ = \end{pmatrix}Y_0 \tag{4.4}$$

and, secondly, in determining the optimum system variant in terms of the preset efficiency criterion which is reduced to the definition and the solution of the corresponding problem of the one-criteria or the multi-criteria optimization.

For achieving any of the specified goals it is necessary to define the internal parameters X whose values modification ensures the desired values of the output parameters Y.

The development of the conceptual model is connected with the creation of an abstract model with the help of which may be determined the composition and the structure of the system under study S', the elements properties and the cause-effect relationships inherent to S' and essential for achieving the modeling goal. Thus on this stage the system of the real world that has to be studied is represented in the form of conceptual model, i.e. he mathematical and logical relations that connect components into system structure. It is recommended to start with the simple model of the considered system. Then while the system is investigated the model should be extended to the necessary level of complexity.

The development cycle of the conceptual model is realized by performing such operations, as orientation, stratification, localization, structurization and control, allocation of processes.

Orientation is understood as the imaginary scrutinizing of the system under study S' within the limits of the mega-system, whose part S' is. When selecting system S' out of the mega-system a researcher has to justify, in terms of modeling goals, what properties and relations of system S' remain in the model and which ones are not taken into consideration as insignificant.

Stratification is the process of selecting the levels of details of system S' which are called strata.

As shown in fig. 4.9, at the highest n-th level there is the system which can be considered as the subsystem of the n-th level. The subsystems of the $(n-1)$-th level which make up the system directly are located lower. Still lower there are subsystems of the $(n-2)$-th level which form the subsystems of higher levels, and so on. At last, at the lowermost level there are the subsystems of zero level or otherwise the system elements. Usually detailing is done to the level for which dependences of the elements output performances from the parameters are known.

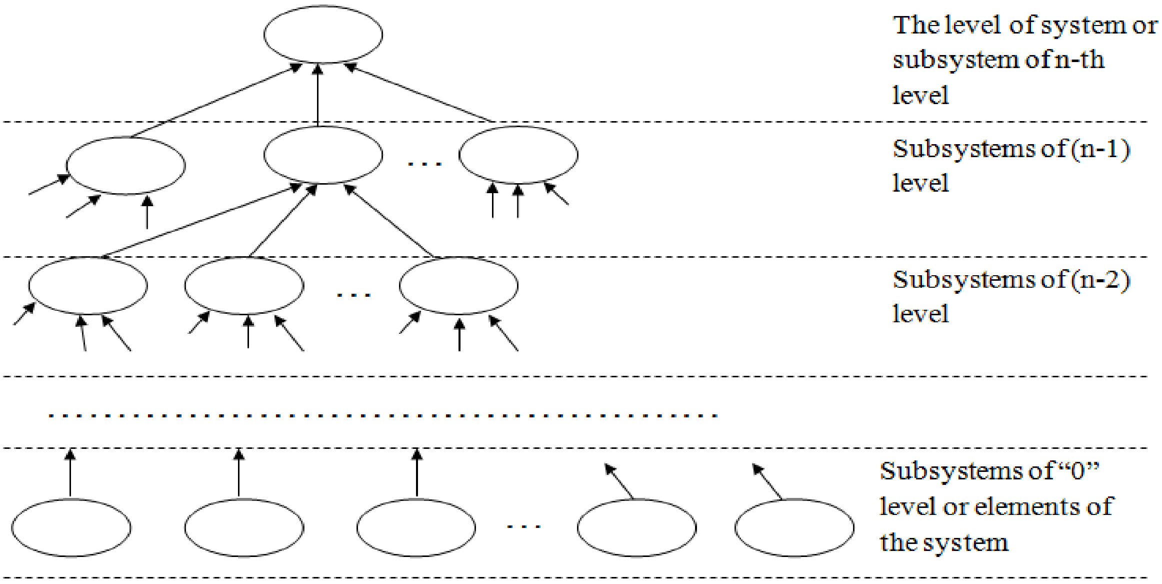

The level of system or subsystem of n-th level
Subsystems of (n-1) level
Subsystems of (n-2) level
Subsystems of "0" level or elements of the system

Figure 4.9 – Stratified representation of a complex system

The model of a complex system in the form of the stratified representation shown in fig. 4.9 is a population of models, each of which reflects the behavior of a system under study on various levels of details. At the same time corresponding principles, variables and associations may be used at each level.

Localization is realized through the representation of environment by the set of generators of external influence. As a rule, these generators are the elements of the system model.

Structurization presumes defining all model elements and their interrelations, which may be material and informational. The material ones reflect the transition of the transformation product between the elements, and the informational ones are intended for transmission of operating actions and information about the elements state.

In some simple systems there may be no informational relations. Control over the functioning of such systems is realized according to the structural control principle that is determined by the system structure.

However, most often in systems there are informational relations and decisive elements in which operating actions are produced. Controls contain instructions on what element and whence should take the initial object, what to do with it and, finally, where to transfer the transformed object. Such systems function according to the programmed or algorithmic control principle. In the conceptual model of such systems all the decision rules or control algorithms should be determined.

Selection of processes is carried out for a dynamic systems modeling. Functioning of these systems is reflected through the description of performance of technological processes of matter, energy or information transformation. Thus the technological process is represented by one of the algorithm types determining what operations, in what sequence and with the help of what resources it is necessary to perform for the system to achieve the preset target state. The set of the algorithms reflecting the realization of all the technological processes selected in the system is, as a matter of fact, the simulation model of the system under study.

The development of the software-based realization of a simulation model is realized on the basis of the chosen software which are divided into procedure-oriented, problem-oriented languages and automated modeling systems.

Following the language selection, the software model is elaborated, which covers the performance of such known operations as improvement and detailing of an algorithm, determination of the input data forms and results, software writing and debugging.

In case of the automated modeling system the software model is created automatically based on the preset mathematical schema, the initial parameters of the system set by a researcher, the environment and the particularities of the system functioning.

For many complex systems for several reasons it is difficult to realize the verification of performability of requirements to the model based on relation (4.2). Firstly, there are difficulties in deriving the output parameters vector of system Y_S when the system is under design or upgrade, that is, when in fact it does not exist. Secondly, (4.2) it is a vectorial estimation and its realization is rather difficult to achieve for all the vector components without applying the compromise schemes. Thirdly, both the vector of model Y_M output parameters and systems Y_S may be random, non-stationary functions and, therefore, for deriving their characteristics it is necessary to process rather a big volume of modeling statistics. Fourthly, it is necessary to justify the value of maximum deviations of the corresponding components of vectors Y_M and Y_S.

In practice, the verification of the feasibility of requirements to models is realized by means of several checks the results of which are analyzed by experts. Models of all the structural elements and relations between them, the environment model for the system under study, etc are subject to verification.

Most often the following types of testing are conducted:

- testing of the models of the system’s of interest elements, the result of which may be the further detailing of the elements;
- testing of the external influence models the result of which may be the modification of suppositions, hypotheses concerning the influence nature;
- testing of the conceptual model, the result of which may change the problem statement;

- testing of the adopted mathematical schema as for its conformity to the character of the processes in the system under study;
- testing of the parameters measurement and computation methods which permits to reveal the values of corresponding errors;
- testing of the model software realization.

If the results of testing show an inadmissible mismatch of a model and a system under study, there appears a necessity for model adjustment or calibration. Three principal types of modifications are singled out – global, local and parametrical.

Global modifications are conducted in case of detecting methodical errors in the conceptual model. These errors are eliminated by constructing a new model.

Local modifications are connected with the development of new models of individual elements, relations or influences.

Parametrical modifications include modifications of values of some, so-called, calibration parameters within the preset limits.

To check the fulfillment of the requirements to the model the processes of model verification and validation are used. *Verification* deals with the programming realization of the simulation model. It checks whether the model performs properly. It is advised that verification take place as a continuous process. Also it is encouraged to use run controllers and debuggers provided by the tools of automated simulation models construction.

Validation is the determination that the conceptual model is an accurate representation of the real system. Validation process must check whether the model can substitute the real system for the purposes of experimentation. If the real system exists, then the ideal way to validate the model is to compare its outputs of the real system and the model. There is not always a real system for such comparison. Therefore researchers may use other methods of validation.

Planning and carrying out of experiments is realized for deriving some modeling statistics, processing of which permits to determine the desired parameters.

The plan of experiments is a certain order of the sets of values combinations of the variable parameters (factors). The number of sets determines the number of experiments.

The main task of scheduling of experiments conducted with a model on a computer, consists in deriving all the necessary information on the system under study at the minimum or limited consumption of computer resources for realization of the process of modeling statistics deriving.

The analysis of the simulation data is carried out following the modeling statistics processing by methods of correlative, dispersing and regression analysis that permits to obtain relation (1.1) where F may be a function or an algorithm.

The analysis of the simulation data includes the problem of the analysis of model sensitivity to the variations of its parameters. In particular, the tolerance of the output parameters to the possible modifications of the system parameters and the environment is checked.

On the basis of the simulation data analysis the model parameters are defined more exactly, that leads to the correction of the conceptual model, the possibility of creating the analytical model of the system under study is sought for, or some weight factors in the efficiency criterion (4.3) are determined.

The necessary condition of successful simulation modeling is documenting and making reports. If the simulation model is reused by the same or another analyst, it may become necessary to review it again and to understand how it is functioning. The documented reports give the possibility to be sure in taken decisions. Also documentation is very useful when a model needs to be changed. The reports on the results of simulation modeling are required for decision-making.

Finally the simulation data are used for decision-making on the system workability in the case when definable parameters meet the restrictions system (4.4), or on the selection of an optimum variant from the set of admissible options, taking into account the efficiency expressing criterion (4.2).

4.4. Experimental design in the method of simulation modeling

The definition of simulation modeling expresses the important role of organization and conduction of experiments with simulator for obtaining modeling statistics which characterize the complex system’s components behavior under different functioning modes.

The main task of experimental design with simulator is getting all necessary information about modeling object under minimal or limited computer resources consumption for modeling process realization.

This problem is solved by the subprogram of MCP modeling statistics collection on the basis of experimental design theory which key notions are factor and reaction [25, 26].

If the goal of the experiment is studying the influence of variable x on variable y, then x is called a factor and y is a reaction. Each factor in the experiment takes different values which are called the levels.

The definite point in the multidimensional space called the factor space corresponds to every fixed collection of factors’ levels.

The location of the point in the factor space is defined by the collection of factors' levels $(x_1, x_2, ..., x_k)$ and the value of reaction y which is connected with the collection of levels by means of reaction function $\Psi - y = \Psi(x_1, x_2, ..., x_k)$.

The geometrical area of m points $y_l = \Psi(x_{1l}, x_{2l}, ..., x_{kl})$ $(l \in \overline{1, m})$ composes the geometrical image of reaction surface.

The view (structure) of function Ψ and its parameters often represent the model of researched system. To define them the experiments are conducted. These experiments must be organized in such a way to provide the possibility to obtain the rules of factors' levels varying under the minimal resources consumption (for example, minimal number of experiments).

The work [6] provides a quite full classification of factors and requirements to them which should be taken into account when designing the rules of factors' levels varying. This in fact makes the sense of experiments plan.

Factors may be controlled and uncontrolled, observable and unobservable, studied and unstudied, quantitative and qualitative, fixed and random.

A factor is controlled, if its levels are intentionally chosen in the experiment. Otherwise a factor is uncontrolled.

A factor is observable, if its levels are observed and registered. Otherwise a factor is unobservable.

Observable and uncontrolled factors are called prorated. There is a large number of such factors, therefore only those factors must be considered which influence the reaction in a valuable way.

A factor is studied, if it is included into the model for studying of object's properties. Otherwise a factor is unstudied.

A factor is quantitative, if its levels are presented by numeric variables. For example, in continuous stochastic models which are designed on the basis of queue theory the quantitative factors are the density of arrival flows, the time of service and the qualitative factors are the queue discipline, the discipline of service and so on.

A factor is fixed, if all its levels are considered. If we take some random sample of factors' levels from the set of possible levels, then a factor is called random.

A simulator is considered as definite, if the set of essential factors is defined and it does not change in the given experiment.

The plan of experiment usually supports the simultaneous change of several factors. The main requirements to the collection of simultaneously changing factors are their consistency and independency.

Consistency means that all combinations of factors are possible. Independency means that it is possible to define a factor on any level from the set of possible levels independently from other factors which are simultaneously changing.

In experimental design theory function $\Psi(x_1,x_2,...,x_k)$ is often represented as a polynomial of *d* degree from *k* factors:

$$y=\hat{b}_0+\sum_{1\le i\le k}\hat{b}_i x_i+\sum_{1\le i,\,j\le k}\hat{b}_{ij}x_i x_j+\ldots+\sum_{1\le i_1,i_2,\ldots,i_k\le k} b_{i_1,i_2,\ldots,i_k}x_1^{i_1}x_2^{i_2}..x_k^{i_k}\,, \qquad (4.5)$$

where $\sum_{1\le j\le k} i_j = d$.

Polynomial (4.5) has $C_{d+k}^{d}=\dfrac{(d+k)!}{d!k!}$ coefficients which must be defined on the basis of experiments results.

Let's define $x_0=1$, $x_{k+1}=x_1^2$, $x_{k+2}=x_2^2$, $x_{2k+1}=x_k^2$, $x_{2k+2}=x_1x_2$, $x_{2k+3}=x_1x_3$, …, $x_{k^1}=x_{k-(d-1)}x_{k-(d-2)}..x_{k-1}x_k$, where there are *d* multipliers in the expression $x_{k'}$. Then expression (4.5) is a homogeneous linear equation

$$y=\sum_{0\le\alpha\le k'}\hat{b}'_\alpha x_\alpha\,, \qquad (4.6)$$

where $\hat{b}'_0=\hat{b}_0$, $\hat{b}'_1=\hat{b}_1$, …, $\hat{b}'_k=\hat{b}_k$, $\hat{b}'_{k+1}=\hat{b}_{11}$, …, $\hat{b}'_{2k+1}=\hat{b}_{kk}$, $\hat{b}'_{2k+2}=\hat{b}_{12}$, …, $\hat{b}'_{k'}=\hat{b}_{(k-(d-1))\ldots(k-1)k}$.

The process of transformation of expression (4.5) into expression (4.6) is demonstrated n the example of bifactor model.

In this case expression (4.5) has the following view

$$y=\hat{b}_0+\hat{b}_1x_1+\hat{b}_2x_2+\hat{b}_{11}x_1^2+\hat{b}_{22}x_2^2+\hat{b}_{12}x_1x_2\,. \qquad (4.7)$$

If we assume that $x_0=1$ and add new factors, which denote nonlinear elements in the considered bifactor model, to the given factors x_1, x_2, i.e.

$$x_3=x_1^2,\ x_4=x_2^2,\ x_5=x_1x_2,$$

then we will obtain the following expression of (4.6) kind for the model (4.7)

$$y=\sum_{0\le\alpha\le 5'}\hat{b}'_\alpha x_\alpha\,, \qquad (4.8)$$

where $\hat{b}'_0=\hat{b}_0$, $\hat{b}'_1=\hat{b}_1$, $\hat{b}'_2=\hat{b}_2$, $\hat{b}'_3=\hat{b}_3$, $\hat{b}'_4=\hat{b}_{22}$, $\hat{b}'_5=\hat{b}_{12}$.

The advantage of expression (4.6) comparing to (4.5) is in its linearity which simplifies the calculation of coefficients $\hat{b}'_\alpha$ compared to $\hat{b}_i$. But in this case we should mention that the dimension of the problem increases as the number of defined coefficients increases ($k'>k$).

Polynomial (4.5) contains C_{d+k}^{d} coefficients which should be defined on the basis of experiments results. That's why plan of experiments *D* must have *N* rows, where the experiment is conducted, not less than the number of defined coefficient, i.e. $N\ge C_{k+d}^{d}$.

Plan D is represented as a matrix with rows that correspond to points of factor space, in which experiments are conducted. This matrix has the following form:

$$D = \begin{Vmatrix} x_{11} & x_{21} & ... & x_{k1} \\ x_{12} & x_{22} & ... & x_{k2} \\ ... & ... & ... & ... \\ x_{1N} & x_{2N} & ... & x_{kN} \end{Vmatrix}, \qquad (4.9)$$

where x_{in} $(i \in \overline{1,k}$, $n \in \overline{1,N})$ is a level of *i*-th factor in experiment *n*.

The realization of experiments in N points of factor space, which coordinates are put into the N rows of matrix D, allows to obtain vector of observations (experiments results): $y = \|y_1 \quad y_2 \quad ... \quad y_N\|^T$, where y_n $(n \in \overline{1,N})$ is a reaction that correspond to experiment *n* in the point of plan $\|x_{1n} \quad x_{2n} \quad ... \quad x_{kn}\|$.

If a model, which connects a reaction and factors, is defined by expression (4.6), then a plan of experiments X must contain N points, the number of which also must be not less than the number of defined coefficients, i.e $N \geq k^{'} + 1$.

Plan X as well as plan D is represented by the matrix with rows that correspond to points of factor spaces where experiments are conducted. Plan X is defined by matrix of the following form:

$$X = \begin{Vmatrix} x_{01} & x_{11} & x_{21} & ... & x_{k1} & x_{k+1\,1} & ... & x_{k^{'}1} \\ x_{02} & x_{12} & x_{22} & ... & x_{k2} & x_{k+1\,2} & ... & x_{k^{'}2} \\ ... & ... & ... & ... & ... & ... & ... & ... \\ x_{0N} & x_{1N} & x_{2N} & ... & x_{kN} & x_{k+1\,N} & ... & x_{k^{'}N} \end{Vmatrix}, \qquad (4.10)$$

where $x_{on} = 1$ $(n \in \overline{1,N})$, x_{kn} $(k \in \overline{1,k'}, n \in \overline{1,N})$ is a level of factor *k* in experiment *n*.

Experiment *n* conducted at a point of factor space, which coordinates are situated in the *n*-th row of matrix *X* – $\|x_{0n} \quad x_{1n} \quad x_{2n} \quad ... \quad x_{kn} \quad ... \quad x_{k'n}\|$, allows defining the corresponding value of reaction

$$y_n = \hat{b}'_0 x_{0n} + \hat{b}'_1 x_{1n} + \hat{b}'_2 x_{2n} + ... + \hat{b}'_k x_{k'n} + e_n,$$

where e_n is an experiment error which is supposed to be independent, normally distributed, random variable with expectation $M[e_n] = 0$ and constant variance $D[e_n] = \text{const}$.

The selection of factors' levels in plans of experiments *D* and *X* is realized subject to the following recommendations.

First of all we should choose the boundaries $x_{i\min}$ and $x_{i\max}$ of the domain of definition of factors. The choice is realized on the basis of physical considerations about the properties of the object under study. For example, if the factor is the current voltage of ordinary power supply network, then $x_{i_{\min}} = 200\,\text{V}$ and $x_{i_{\max}} = 250\,\text{V}$.

The selected boundaries define some local area *G* of the factor space.

Then the basic or the null level x_{i0} and varying interval Δx_i is set for each factor. For example, for linear bifactor model

$$y = b_0 + b_1 x_1 + b_2 x_2$$

area *G* with null factors' levels is shown on fig. 4.10.

Fig. 4.10 shows that the plan of experiments is designed by varying each factor x_i on several levels relatively to the initial point x_{i0} which defines the center of experiments.

The plan of experiments which realizes all possible combinations of factors' levels is called the full factor experiment (FFE).

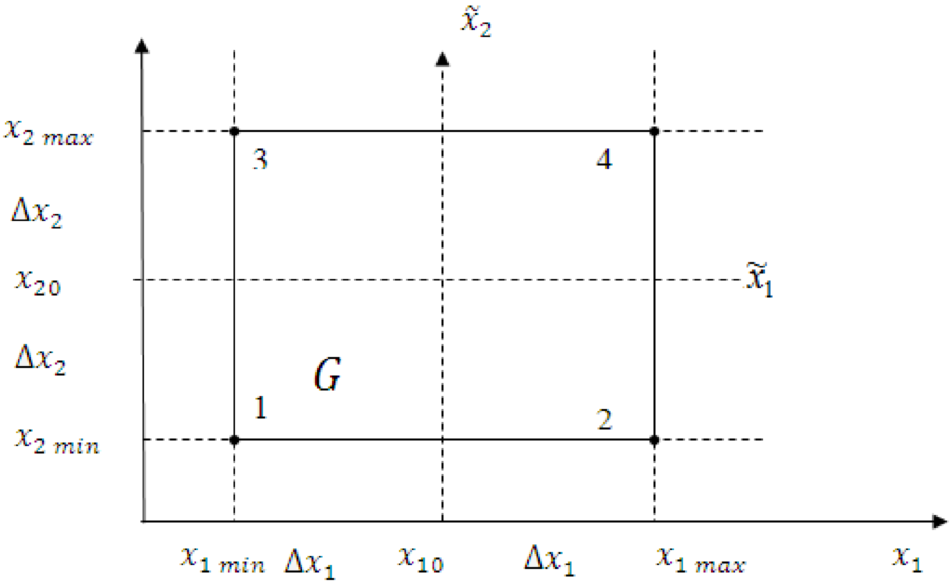


Figure 4.10 – Area *G* with different combinations of levels in the 1-4th experiments

If the structure of the model includes only linear members of polynomial and multiplications of factors, then the plan of experiments with varying of all *k* factors on both levels is used for the estimation of model's coefficients. These plans are called 2^k plans, where $N = 2^k$ is a number of all possible experiments for estimation of model's coefficients of the considered structure.

Let, for example, $k = 3$. Then a three-factor model, which structure includes only linear members and products of factors, has the following form:

$$y = b_0 + b_1 x_1 + b_2 x_2 + b_3 x_3 + b_{12} x_1 x_2 + b_{13} x_1 x_3 + b_{23} x_2 x_3 + b_{123} x_1 x_2 x_3 .$$

Eight coefficients – b_0, b_1, b_2, b_3, b_{12}, b_{13}, b_{23}, b_{123} of this model require the estimation. $N = 2^k = 2^3 = 8$ experiments are necessary for their estimation. It is easy to see that the number of experiments strictly corresponds to the number of coefficients that have to be defined. Also we should mention that this number of experiments is minimal for determining coefficients of the considered model.

In the real problems factors have different units of measurement and intervals of changes. To produce the combinations of such factors it is necessary to make the scaling operation preliminary.

The following transformation is often used for scaling operation

$$\hat{x}_i = \frac{x_i - x_{i0}}{\Delta x_i} \; (i \in \overline{1,k}), \tag{4.11}$$

where $\hat{x}_i$ is a scaled level of *i*-th factor; x_i – natural level of *i*-th factor; and the value of Δx_i is defined by expression

$$\Delta x_i = \frac{x_{i\max} - x_{i\min}}{2}.$$

As it is known in 2^k plans each factor varies on two levels. Usually these levels are $x_{i\min}$ and $x_{i\max}$ $(i \in \overline{1,k})$. On the basis of transformation (4.11) it is easy to get that the scaled level $\hat{x}_i = -1$ corresponds to the level $x_{i\min}$ and similarly $\hat{x}_i = +1$ corresponds to $x_{i\max}$.

The scaled area *G* of plan 2^2 is shown on fig. 4.11.

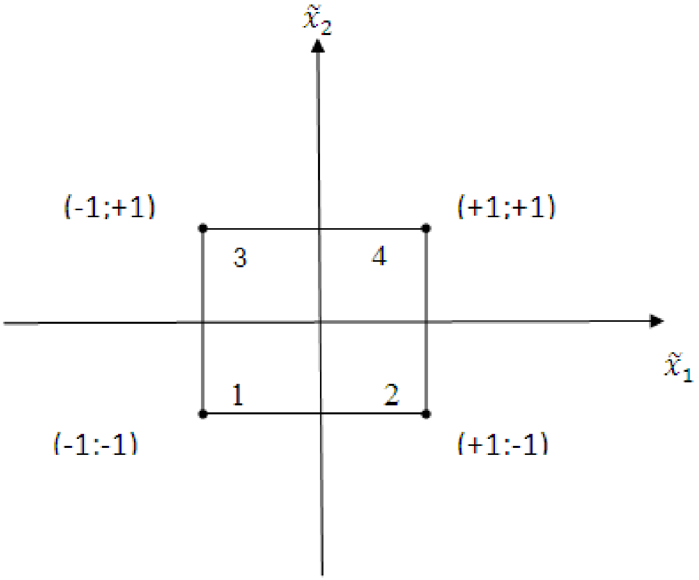


Figure 4.11 – Area *G* of plan 2^2

The plan 2^2, which is shown on fig. 4.11 in the form of vertexes of area *G*, as it was mentioned above is represented by planning matrix *D* which has the following form:

$$D = \begin{vmatrix} -1 & -1 \\ +1 & -1 \\ -1 & +1 \\ +1 & +1 \end{vmatrix}.$$

The number of row on matrix *D* corresponds to the number of plan's experiment.

The number of experiments of plan 2^2 is quite enough for estimation of four coefficients of the model of the following form

$$y = b_0 + b_1x_1 + b_2x_2 + b_{12}x_1x_2.$$

If for description of reaction's surface we take the linear model of the form

$$y = b_0 + b_1x_1 + b_2x_2,$$

in which it is necessary to estimate only three coefficients b_0, b_1, b_2, then obviously plan 2^2 is excessive, because the number of experiments of this plan is greater for a unit than the number of coefficients estimated.

It is easy to show that this abundance increases while the number of factors *k* is increased.

Really if $k = 3$, the number of experiments of plan 2^k equals to $N = 2^3 = 8$ and the number of estimated coefficients of the linear three-factor model

$$y = b_0 + b_1x_1 + b_2x_2 + b_3x_3$$

equals to 4. So in case of $k = 3$ the abundance is equal to four. We should note that when $k = 2$, the abundance was equal to unit.

Since experiments' conduction needs the resources costs, the abundance leads to inefficient resources usage.

In order to minimize the abundance with FFE of the 2^k form, when each factor varies on two levels, the fractionary factor experiments are used, the number of which corresponds to the number of estimated coefficients of the model that describes the reaction surface.

Except symmetric two-level 2^k plans experimental design theory introduces multi-level plans, in which factors vary on 3, 4, etc., *m* levels. These plans are denoted as 3^k, 4^k, …, m^k.

Except this experimental design theory also uses multi-level non-symmetric plans, in which factors vary on different levels. The number of experiments in such plans is defined as

$$N = q_1q_2..q_k, \tag{4.12}$$

where q_i ($i \in \overline{1,k}$) is a number of levels of *i*-th factor.

The decreasing of experiments' number N and consequently of resources costs for their conducting is possible by means of decreasing the number of multipliers in expression (4.12), which is equivalent to decreasing the number of factors *k*, and by means of decreasing the values of multipliers q_i $(i \in \overline{1,k})$, which is equivalent to decreasing of the number of levels of *i*-th factor.

The number of factors is defined by experiments' goal. As a rule, two goals are considered. The first one is building the dependency of reaction on factors for revealing the system's properties, which is connected to the solution of particular analysis problem. The second goal is finding of some combination of factors' levels which provides the extreme value of reaction. This goal is related to the solution of synthesis problem.

To achieve the first goal, as a rule, the realization of FFE is required. But in this case it is necessary to take into account the essential factors. The practice shows that 20 % of factors define 80 % of system's properties and the rest 80 % factors define the rest 20 % of system's properties. These 20 % of factors are essential. They are determined on the basis of experiments.

To achieve the second goal the usage of FFE is irrational, because it is equivalent to the complete search of variants, which causes huge resources costs. To achieve the second goal the more rational solution is obtaining the a priori information about the character of reaction surface and based on this information to use one of the methods of optimum searching of reaction surface. It is possible to use such well-known methods as random sampling technique, uniform network, the fastest release and others [27].

4.5. Accuracy and validity of modeling results and rules of automated experiment's finish

The problem of accuracy and validity of modeling results occurs when the simulation model has elements of statistical modeling, which leads to stochastic results of experiments with a model.

The solution of this problem is connected with obtaining estimates of accuracy and validity in case of the given number of experiments, which is caused by the limited resources, or vice versa with obtaining estimate of the experiments' number in case of the given values of accuracy and validity.

Let *E* be some characteristic of the researched system *S*, the value of which is the experiment's result. Let's denote $\hat{E}$ the estimate of the considered characteristic.

The stochasticity of experiments' results and the limitation of experiments' number *N* lead to the fact that $E \neq \hat{E}$ in general case.

The value $\varepsilon = \left|E - \hat{E}\right|$ is called the estimate's accuracy and the probability $Q = P\left\{\left|E - \hat{E}\right| < \varepsilon\right\}$ is called the validity of the estimate.

The value $\varepsilon_0 = \dfrac{\varepsilon}{E}$ is called the relative accuracy and the probability

$$Q_0 = P\left\{\frac{\left|E - \hat{E}\right|}{E} < \varepsilon\right\}$$

is called the relative validity of the estimate.

In order to define the experiments' number *N* with respect to the given ε, *Q* or in order to define the accuracy ε and reliability *Q* with respect to the given value of *N*, it is necessary to know the distribution law at least of accuracy value ε. However a priori it is unknown. This difficulty is overcome by making a hypothesis about the character of the distribution law of the random variable of the estimate $\hat{E}$.

For example, let the goal of experiments with simulation model be the finding of estimate $\hat{p}$ of the probability of some event A occurrence – $p = p(A)$.

In this case $E = p$, $\hat{E} = \hat{p}$.

The estimate $\hat{p}$ can be considered as $\hat{p} = \dfrac{m}{N}$, where *m* is the number of event A occurrence in *N* experiments.

Then the reliability *Q* is defined by expression

$$Q = P\left\{\left|P - \frac{m}{N}\right| < \varepsilon\right\};$$

$$Q = P\left\{p - \varepsilon < \frac{m}{N} < p + \varepsilon\right\}.$$

The estimate $\hat{p}$ is represented as

$$\hat{p} = \frac{1}{N}\sum_{i \in \overline{1,N}} x_i\,,$$

where x_i is the result of realization of some random variable ξ, which takes the value of 1, if the event A occurs, and the value of 0, if the event A does not occur. In this case $x_i = 1$ with probability *p* and $x_i = 0$ with probability $(1 - p)$.

Let's determine the expectancy $M[\xi]$ and variance $D[\xi]$ of the random variable ξ:

$$M[\xi] = x_i p + x_i(1 - p) = 1 \cdot p + 0 \cdot (1 - p) = p;$$

$$D[\xi] = (x_i - M[\xi])^2 p + (x_i - M[\xi])^2 (1 - p) = (1 - p)^2 p + (0 - p)^2 (1 - p) =$$
$$= p(1 - p).$$

The obtained values $M[\xi]=p$, $D[\xi]=p(1-p)$ are used for defining of the expectancy and variance of the estimate $\hat{p}$ $-M[\hat{p}],D[\hat{p}]$:

$$M[\hat{p}]=M\left[\frac{m}{N}\right]=M\left[\frac{1}{N}\sum_{i\in\overline{1,N}}x_i\right]=\frac{1}{N}M\left[\sum_{i\in\overline{1,N}}x_i\right]=\frac{1}{N}\sum_{i\in\overline{1,N}}M[x_i]=\frac{1}{N}N\cdot M[\xi]=$$
$$=M[\xi]=p;$$

$$D[\hat{p}]=D\left[\frac{m}{N}\right]=\frac{1}{N^2}D\left[\sum_{i\in\overline{1,N}}x_i\right]=\frac{1}{N^2}\sum_{i\in\overline{1,N}}D[x_i]=\frac{1}{N^2}N\cdot D[\xi]=\frac{1}{N}p(1-p).$$

The fact that $M[\hat{p}]=p$ means the nonbias of the estimate $\hat{p}$.

From the central limit theorem of probabilistic it follows that under enough big values of N $\hat{p}=\dfrac{m}{N}$ can be considered as the random variable described by the normal distribution law with expectancy $M[\hat{p}]=p$ and variance $D[\hat{p}]=\dfrac{1}{N}p(1-p)$.

Therefore to define the validity of the estimate we can use the relation

$$Q=P\left\{p-\varepsilon<\frac{m}{N}<p+\varepsilon\right\}=\Phi_0\left(\frac{p+\varepsilon-p}{\sqrt{p(1-p)}}\sqrt{N}\right)-\Phi_0\left(\frac{p-\varepsilon-p}{\sqrt{p(1-p)}}\sqrt{N}\right),$$

where Φ_0 is the standard function called the Laplace integral.

According to the property of symmetry of the normal distribution law curve, which means that

$$\Phi_0(-z)=1-\Phi_0(z),$$

for the considered case we get:

$$\Phi_0\left(\frac{p-\varepsilon-p}{\sqrt{p(1-p)}}\sqrt{N}\right)=1-\Phi_0\left(\frac{p+\varepsilon-p}{\sqrt{p(1-p)}}\sqrt{N}\right).$$

Subject to this relation the expression for estimate's validity takes the form:

$$Q=\Phi_0\left(\frac{p+\varepsilon-p}{\sqrt{p(1-p)}}\sqrt{N}\right)-1+\Phi_0\left(\frac{p+\varepsilon-p}{\sqrt{p(1-p)}}\sqrt{N}\right).$$

The elementary transformations allow obtaining the relation:

$$\Phi_0\left(\frac{\varepsilon\sqrt{N}}{\sqrt{p(1-p)}}\right)=\frac{1+Q}{2}.$$

From the obtained relation it follows that

$$\frac{\varepsilon\sqrt{N}}{\sqrt{p(1-p)}}=t_\varphi, \tag{4.13}$$

where t_φ is the quantile of the normal distribution law of the order

$$\varphi = \frac{1+Q}{2},$$

the values of which are given in the special tables.

From the expression (4.13) the accuracy is defined

$$\varepsilon = t_{\varphi}\sqrt{\frac{p(1-p)}{N}}, \qquad (4.14)$$

which is inversely proportional to $\sqrt{N}$.

From the expression (4.14) it is easy to get the experiments' number *N* which are necessary for obtaining the estimate $\hat{p}$ with the accuracy ε and validity *Q*:

$$N = t_{\varphi}^{2}\frac{p(1-p)}{\varepsilon^{2}}. \qquad (4.15)$$

We could use the expressions (4.14), (4.15) for calculation of the accuracy ε and the experiments' number *N*, if we knew the value of probability *p*.

But in practice a so-called preliminary modeling is made for definition of *p*. It requires the realization of $N = N_0$ experiments. As the result of these experiments the value $p_0 = \frac{m}{N_0}$ is defined, which is taken as the value of p, i.e. $p = p_0$.

Let's mention that from expression (4.15) for the experiments' number N it follows the strong dependency of value N from the variance of the estimated random variable $D[\hat{p}] = \frac{1}{N}p(1-p)$. Therefore, it is obvious that from the point of view of resources costs for experiments conduction it is beneficial to select such estimated characteristics of the modeling system *S* efficiency which have small values of variance, that leads to the small number of required experiments.

The rule of automatic experiment's finish is implemented in the subprogram (SP) of simulation finish. Usually this rule is included to the mentioned SP in two ways:

- on the basis of expression (4.15);
- on the basis of assumption about the distribution of probabilities of the input characteristics.

The first way is realized by the double run, when at first the run of N_0 experiments is done, which allows to determine the value $p_0 = \frac{m}{N_0}$ that is supposed to be equal to p in expression (4.15), i.e. $p = p_0$. After this the value *N* is defined with the help of expression (4.15). this value is used as a criterion of experiments' finish. In this case if $N_0 \geq N$, then experiments are finished immediately, and if $N_0 < N$, then the second run is required and $N - N_0$ experiments have to be conducted. Then the experiments are finished.

The second way means the realization of the sequential analysis for determining the minimal necessary experiments' number *N*. The practice shows that the usage of this method allows defining such experiments' number *N* which is twice less than the value of *N* obtained basing on the first method. Thos leads to the essential decreasing of used resources.

According to the second way the number *N* is considered as a random variable which depends on results of $(N-1)$ previous experiments.

In the sequential analysis the sample volume, i.e. the value *N*, is not fixed. After the latest experiment one of the possible decisions should be taken:

- to accept some hypothesis;
- to accept the alternative hypothesis;
- to continue the experiment, i.e. to conduct $(i+1)$ experiment.

Such sequential planning of the process of experiments' conduction is based on the principle of maximum likelihood and sequential testing of statistical hypotheses, the sense of which is in the following considerations.

It is supposed that the distribution of the universe population is characterized by the function of probability density $Y = f(y,\theta)$ with the unknown parameter *θ*.

Two hypotheses are made – H_0 and H_1. Hypothesis H_0 means that the value of the unknown parameter *θ* equals to θ_0, i.e. $\theta = \theta_0$. Another hypothesis H_1 supposes that $\theta = \theta_1$.

These hypotheses are tested on the basis of the sample of increasing volume *m*.

The probability of obtaining sample of volume m under the condition of valid hypothesis H_0 is equal to

$$P_{0m} = f(y_1,\theta_0)\cdot f(y_2,\theta_0)\cdot\ldots\cdot f(y_m,\theta_0).$$

The same probability but under the condition of valid hypothesis H_1 is defined as

$$P_{1m} = f(y_1,\theta_1)\cdot f(y_2,\theta_1)\cdot\ldots\cdot f(y_m,\theta_1).$$

The procedure of hypotheses validity testing is realized on the basis of the likelihood relation $\dfrac{P_{1m}}{P_{0m}}$.

After each next experiment the values P_{1m} and P_{0m} are defined with the following testing of the conditions and decision making:

- if $\dfrac{P_{1m}}{P_{0m}} \le B$, then hypothesis H_0 is accepted;

- if $\frac{P_{1m}}{P_{0m}} < A$ or $\frac{P_{1m}}{P_{0m}} > B$, then experiments are continued;
- if $\frac{P_{1m}}{P_{0m}} \geq A$, then hypothesis H_1 is accepted.

In the given conditions $0 < B < 1,\ A > 1,\ m \in \overline{1, N}$.

To provide convergence it is necessary that

$$A \leq \frac{1-\beta}{\alpha};\ B \geq \frac{\beta}{1-\alpha},$$

where α is the probability of the error of the first kind, i.e. it is the probability to reject the hypothesis H_0, if it is correct; β is the probability of the error of the second kind, i.e. it is the probability to accept the hypothesis H_1, if it is wrong.

For example, let the random variable y to be normally distributed with the known variance σ^2 and unknown expectancy μ. Two hypotheses $H_0$and H_1 are made. Hypothesis H_0 supposes that $\mu = \mu_0$ and hypothesis H_1 supposes that $\mu = \mu_1$.

Since

$$P_{1m} = \frac{e^{-\frac{1}{2\sigma_2}\sum\limits_{i\in\overline{1,m}}(y_i-\mu_1)^2}}{(\sqrt{2\pi\sigma})^m};\quad P_{0m} = \frac{e^{-\frac{1}{2\sigma_2}\sum\limits_{i\in\overline{1,m}}(y_i-\mu_0)^2}}{(\sqrt{2\pi\sigma})^m},$$

then $\frac{P_{1m}}{P_{0m}} = e^{-\frac{1}{2\sigma_2}\left(\sum\limits_{i\in\overline{1,m}}(y_i-\mu_1)^2-\sum\limits_{i\in\overline{1,m}}(y_i-\mu_0)^2\right)}$.

After obtaining the mentioned relation of likelihood for the considered case, the conditions are checked and particular decisions are taken.

4.6. Processing and analysis of experiments results with systems models

After designing experiments with system's model, realization of the composed plans it is necessary to collect the experiments' results in the form of definite modeling statistics and to organize their processing according to the modeling goal. The processed results are given to the researcher or decision-maker.

As it was mentioned in section 4.3, the collection of modeling statistics is realized by the corresponding subprogram of simulation model. The experiments' results processing and their delivery to researcher is done by the subprogram of simulation finish.

The selection of the processing methods is made with respect to the following peculiarities of experiments with systems' models [6].

Firstly, experiments with systems’ models allow obtaining samples with big enough volume for quantitative estimate of characteristics of the process of researched system functioning. Such big sample volumes allow obtaining parameters’ estimates of high accuracy and validity. However the problem of storage of intermediate results occurs during the processing of large amount of information.

As a rule, this problem is solved by elaboration of recurrent algorithms of information processing which allows obtaining estimates in the process of experiments’ conduction.

Secondly, the complexity of modeling systems does not allow a priori to make enough grounded judgment about distribution law, for example, concerning output characteristics of the system. Therefore there is a widespread practice of estimation of distribution moments – expectancy, variance, correlation moment, etc.

Thirdly, the block structure of the complex system’s model predefines the possibility of separate research of its parts. Such research is possible under the condition of simulation of input variables for one of the model’s parts according to output variables of another part.

There are some requirements to the quality of parameters’ estimation known from statistics theory [19]:

- nonbias of the estimate, when $M[\hat{x}]=x$, where $\hat{x}$ is the estimate of parameter x;
- efficiency of the estimate, when the estimate $\hat{x}$ is so that it provides the minimum of variance, i.e. $\min M\left[(\hat{x}-x)^2\right]$.
- the consistency of the estimate, which provides the convergence of probability of estimate $\hat{x}$ to the estimated parameter x under increased number of experiments, i. e.

$$\lim_{N\to\infty} P\left[|\hat{x}-x|\geq\varepsilon\right]=0,$$

where ε is error value.

In the case when the goal of experiment is exactly obtaining the distribution law, but not distribution moments, then according to the experiments’ results we can obtain sample distribution law $\hat{F}(x)$ or density function $\hat{f}(x)$. Then a hypothesis about the correspondence of the obtained empirical distribution with some theoretical distribution is made. The hypothesis testing is done with the help of statistical fitting criteria of Kolmogorov, Pearson, Smornov, Student and others [19].

The modeling statistics in the form of the set of parameters’ values and their further processing via statistical methods provide the analysis of connections between parameters' values. Methods of correlation, regression, variance analyses are used for this purpose [19].

With the help of *correlation analysis* we can define how close is the relation between random variables. The power of such relation is determined with the help of correlation coefficients

$$r_{\xi\eta}=\frac{M[\xi-M[\xi]]\cdot M[\eta-M[\eta]]}{\sqrt{D[\xi]D[\eta]}},$$

where $M[\xi]$, $M[\eta]$ – are expectancies of random variables ξ and η; $D[\xi]$, $D[\eta]$ are variances of random variables ξ and η.

The estimate of correlation coefficient is made basing on modeling statistics, or in other words basing on *N* realizations of experiments:

$$\hat{r}_{\xi\eta}=\frac{\sum\limits_{k=\overline{1,N}}(x_k-\bar{x})(y_k-\bar{y})}{\sqrt{\sum\limits_{k=\overline{1,N}}(x_k-\bar{x})^2\sum\limits_{k=\overline{1,N}}(y_k-\bar{y})^2}},$$

where $\bar{x}=\frac{1}{N}\sum\limits_{k\in\overline{1,N}}x_k$, $\bar{y}=\frac{1}{N}\sum\limits_{k\in\overline{1,N}}y_k$.

Value $\hat{r}_{\xi\eta}$ characterizes the closeness of dependency between random variables ξ and η to the linear one. In case when $|r_{\xi\eta}|=1$, there is functional, non-stochastic, linear dependency of the form

$$y=b_0+b_1x.$$

In case when $|r_{\xi\eta}|=0$, we can say about mutual uncorrelatedness.

We should mention that independent random variables are always uncorrelated. However uncorrelated random variables may be dependent.

The case when $0<r_{\xi\eta}<1$ corresponds to the presence of linear correlation with diffusion or to the presence of nonlinear correlation of experiments' results.

As it was mentioned, correlation analysis allows to find out the existence of relationship and its degree, but in this case the model of relationship between parameters or variables interesting for researcher is not defined.

To define the model of relationship the regression analysis is used. As a result of experiments we get the collections of data in the form of definite values that are interesting for the researcher. Often they are the output parameters on the one hand, and on the other hand they are internal parameters of the system.

Visual analysis of this data allows making an assumption about model's structure or, in other words, allows to define the type of the model. After this the values of model's parameters are defined. They should minimize an error between data obtained on the basis of experiments and model. This problem is often solved with the help of the least square method [19].

During processing and analysis of experiments' results the problem of comparison of mean samples occurs. For example, let n series of experiments be conducted. As a result n populations of random variable y were obtained $\{y^{(1)}\}$, $\{y^{(2)}\}$, …, $\{y^{(n)}\}$. If it happens that expectancies of these populations $M[y^{(1)}]$, $M[y^{(2)}]$, …, $M[y^{(n)}]$ differ slightly, then population $\{y^{(1)}\}$, $\{y^{(2)}\}$, …, $\{y^{(n)}\}$ makes up homogeneous statistical material which can be considered as the combined population of the form $\{y^{(1)}, y^{(2)}, ..., y^{(n)}\}$, which essentially increases the volume of information about properties of the object under study based on its model.

The hypothesis testing about equal expectancies may be realized with the help of one of the fitting criteria (Smirnov, Student or any other) and procedure of paired-comparison of populations' expectancies. In the case of large number of populations the number of paired-comparisons will be big, which makes such hypothesis testing inefficient.

The more efficient hypothesis testing about equal expectancies is realized on the basis of variance analysis which includes the hypothesis testing concerning the identity of sample and universe variances.

4.7. Languages of the simulation modeling

As it was mentioned in the previous section, at the development stage of the software realization of the simulation model an important operation is the choice of the language with the help of which the software product is created.

Simulation modeling languages most often belong to the procedure-oriented ones, that is to such languages which are intended for solving a certain class of problems and therefore contain the instructions convenient for formulating typical solution methods for the tasks of the preset class.

Over the recent years a great number of languages have been developed and upgraded

Various offered classifications serve the purpose of orienting in this maltitude. guided in this set. By one of them the simulation modeling languages are divided into classes according to the types of modeled processes. As a result the class of continuous system simulation languages is singled out, which includes MIMIC, DYNAMO, etc. Further is distinguished the class of discrete event simulation languages, of which well-known representatives are GPSS, SIMULA, SIMSCRIPT, etc. One more class is the class of combined discrete continuous simulation languages, whose representatives are GASP, FORSIM, etc.

All these modeling languages are united by the compulsory provision of such a collection of software and concepts which ensure the solution of the following six problems.

The *first* is the problem of *combination* which consists in the fact that the parallel processes in modeled systems have to be reflected by the functioning computer sequentially. This problem, as it was mentioned, is solved by introducing modeling or system time. In this case basically two methods are applied – the fixed interval (step) method and the method of the variable interval or step till the next event, explicitly described in section 4.2.

The *second* is the problem of *size.* The majority of simulated systems have a complicated structure and algorithms of behavior and therefore their models are large in volume, that is, great capacity of the computer memory is required for their allocation. In this connection the dynamic memory distribution is used when the required components of the model appear in the operative memory or leave it depending on the current state. The dynamic memory distribution is possible if the model is constructed by a block (modular) principle.

The *third* is the problem of *modifications*. Constant modification of state is characteristic of dynamic systems. Therefore modeling languages provide for the list processing reflecting the modification of condition of the modeled system functioning processes.

The *fourth* is the problem of *coherence*. Occurrence of events in the model of the system functioning is predetermined by rather complicated conditions due to the presence of a big number of reciprocal relationships between the model components. Means of algebra, logics and the set theory are necessary to describe these conditions.

The *fifth* is the problem of *stochasticity*. In the models of complex systems there appears the necessity of assigning random actions in the form of simple and complex random events, variables, functions, processes. Modeling languages provide for the possibility of their deriving based on the programs for generation of sequences of pseudorandom, quasi-uniformly distributed numbers within the interval of (0,1) with the subsequent transformation of these numbers into the sequence of random values with the preset law of distribution.

The *sixth* is the problem of *analysis* which is solved by developing certain means for the analysis of model sensitivity to variations of its parameters.

4.8. The system of modeling GPSS

The main elements of the GPSS-model are dynamic and statistical objects [25].

Dynamic objects comprise transactions which are the analogues of the dynamic elements of a modeled system. In particular, if the modeled system is a shop, the transactions are buyers; if the modeled system is an industrial site, the transactions may be represented by the blanks of the future details and so on.

Static objects include, first of all, blocks which are subprograms in macroassembler or in C language with parameters (operands) for accessing them.

The internal mechanism of transfer of control is realized by means of travel of transactions from block to block in modeling time. Let the motions be determined by a block diagram.

At the moment of a transaction entry into a block a corresponding subprogram, realizing the block function, is called. Relocation of a transaction continues until one of the following possible conditions is met:

- a transaction enters the block which performs the function of its delay for a certain time;
- a transaction enters the block which performs the function of its removal from the model;
- a transaction makes an attempt to enter the next block which does not admit it, therefore the transaction remains in the former block until the model conditions change in such a way that another attempt to enter into the block will succeed.

The types of objects in the GPSS are: "resource", "queue" and "table".

The "resource" type objects are the analogues of servicing devices of the queuing systems, which may be single-channel, multi-channel serving devices (SChD, MChD) and logic keys.

SChD, provided it is free, can be occupied only with one transaction at the given instant. MChD represents several parallel functioning SChDs and therefore can be used for simultaneous service of several transactions.

In queuing systems may appear situations when it is necessary to interrupt the work of servers for a certain time. This necessity may be connected with carrying out preventive maintenance, repair and other types of work ensuring normal server functioning. Such events may block or change the transactions relocation.

For modeling of the said situation there introduced some logic keys which can be turned in the position “switched on” or “off” by the transaction. These positions of a logic key are used by other transactions for selecting the course or for waiting for the moment of the key state change.

The motion of the transactions flow may be detained due to the absence of free SChDs or MChDs. Therefore before the busy servers there appear queues which in GPSS are reflected by the corresponding objects of the "queue" type.

Finally, the object "table" is used for gathering the statistics of the chance quantities characterizing transactions movement and the process of their service.

Objects of "resource", "queue" and "table" types possess properties and methods that change these properties. In GPSS these properties are called standard numerical attributes (SNA) for calculation of which interpreters are used.

In the object of the "resource" type the interpreter ensures automatic calculation of such SNAs, as the total holding time, number of transactions which occupied the device, the device utilization factor, the average holding time of the device by one transaction and others.

In the object of the "queue" type the interpreter calculates the queue length, the average time in the queue and others. The interpreter ensures certain discipline of service, which may be examplified by queues of the FIFO type ("First in, first out" – "the first came, the first was served ") and queues of the LIFO type ("Last in, first out" – "the last came, the first was served") and others. For the realization of this service disciplines certain lists of users are used.

In the objects of the "table" type the interpreter provides for calculation of mathematical expectation, mean-square deviation and other numerical characteristics.

On the whole the program-interpreter realizes the modeling process control. Besides the specified SNA calculation it realizes:

- start of the modeling process;
- tracing each transaction movement, calling the necessary service programs in the course of movement;
- keeping the list of time-ordered events (the list of future events) (LFE);
- modeling time alteration by advancing the CLOCK variable from one event origin to the nearest event or events origin.

All the specified operations are fulfilled by means of GPSS operators which are divided into blocks and commands. The GPSS-block format is determined by the entry

<label> <operation> <operands> <comments>

The label determines the name of the block in the form of the sequence of symbols (up to 250), beginning with the letter of the Latin alphabet. Besides the Latin letters it is admitted to use figures and the underscore character.

Blocks operations are represented by verbs expressing functions of the blocks. For example, GENERATE, TERMINATE, etc.

Blocks operands are the characteristics of the functions performed by the blocks. The number of operands does not exceed seven and they are designated by symbols A, B, C, D, E, F, G. Operands follow one another and are separated by commas or one blank space. In the absence of an operand a comma is put instead of it.

Comments serve for additional explanations and are not mandatory. Comments are separated from the operands field by symbol «;».

Commands are necessary for the model programming and for interaction with the model. Usually commands include data declarative statements, control-flow statements and GPSS World language commands.

The modeling process start is realized by means of the command START which has the following format:

START A, [B], [C], [D].

Let us note at once, that for formats description the square brackets [] mean optionality of the corresponding field.

Operand A is the positive integer which determines the value of the moment of the model execution termination. Usually this number is the current value of the completion counter, defined by the number of transactions, serviced by blocks. Block TERMINATE which removes transactions from the model is used for the counter reading decrement. It is obvious, that the modeling process comes to the end when A possesses the value zero, that is when all the transactions have been removed from the model.

The operand B is used for statistics output. On default, when no value is assigned to the operand, the standard statistics is produced. When value «NP» is assigned to the operand, the statistics is not produced. Operand C is not used in the modern versions of GPSS. Operand D is used for outputting the list of current events (LCE), including the events at the current moment of modeling time, and the previously mentioned list of future events (LFE). If some positive integral value is assigned to the operand, the lists are punched out. If the operand is assigned value "zero", the lists are not punched out.

After the execution of the START command the interpreter or the modeling control program (MCP) accesses all the blocks GENERATE, which are used for creating transactions and their input into the model.

The block format is determined by the entry

GENERATE [A], [B], [C], [D], [E],

where A – is an average value of the transaction input interval into the model; B – is the dispersion of the possible values relative to the average value; C – is an instant of time at which the first transaction appears; D – is a delimiter of the total number of transactions which is introduced into the model through the given block GENERATE; E – is a level or a class of each of the transactions, assigned by numbers from 0 to 127.

For example,

GENERATE 4, 2

Such a format of the block corresponds to setting of the law of uniform distribution of the intervals between the transactions input moments with the average value A=4 and the dispersion of possible values relative to the average value B=2. Thus, the possible values of the interval will be:

2, 3, 4, 5, 6.

The moments of transactions appearance are entered into the LFE following which the MCP accesses the LCE. Since the LCE in the beginning is empty, the MCP accesses the LFE with the purpose of choosing transactions and the minimum moment of appearance and their transfer into the LCE. The first transaction moves from the LCE along the model blocks. In case of the transaction delay for the reason not connected with the ADVANCE block functioning it remains in the LCE and the MCP moves such a transaction from this list further along the blocks. At getting of the transaction into the ADVANCE block, its output is planned, the fact being fixed in the LFE.

In the ADVANCE block the transaction delay for the period of its service by a corresponding device in the modeled system is realized. The service time interval is a random value which is described by the preset distribution law.

The block format is determined by the entry ADVANCE A [B], where A – is the average time for service delay; B – is the scatter of possible values relative to the average one under the condition of their uniform distribution.

For example, if ADVANCE 5.3 it means that the service time is a random uniformly distributed value in the range [2.8].

Service devices in a system usually have names in the form of a character set or a number. The service process provides for indication of the names of those devices in which the service begins and comes to the end. Blocks SEIZE (to occupy) and RELEASE (to free) are used for this purpose. They have the following formats:

SEIZE A,

where A – is a symbolical or numerical name of the occupied device, and

RELEASE A,

where A – is a symbolical or numerical name of the device being deallocated.

Blocks SEIZE and RELEASE, as a rule, are used jointly with block ADVANCE. For example,

```
SEIZE OBS
ADVANCE 5,2
RELEASE OBS.
```

According to the example, the transaction will occupy the serving device with the name OBS, it will be serviced during one of the possible intervals of time {3, 4, 5, 6, 7}, it will release the serving device with the name OBS. Following all abovementioned the interpreter makes an attempt to relocate the transaction into the next block of the model.

Blocks QUEUE and DEPART in the modeling system of GPSS ensure the possibility of taking account of the information on waiting for transactions in the "queue" objects. This information includes:

- the number of transactions inputs into the queue;
- the number of transactions with zero waiting time;
- the maximum queue length;
- the average number of transactions in the queue;
- the average waiting time of transactions in the queue.

Since there may be several objects of the "queue" type, for their distinction there appears the necessity of assigning names to the recorders of these queues that are introduced into the model with the help of a pair of blocks:

QUEUE A [B],
DEPART A [B],

where operand A stands for the queue name, which the transaction should enter or leave. The name can be numerical or symbolical.

Operand B determines the number of transactions by which the queue increases or decreases.

The transaction input into block QUEUE triggers the performance of the following operations:

- the inputs counter indication and the value of the queue length increase by B;
- the current queue length value is entered into the standard numerical attribute QS (queue name);
- the transaction joins the queue with storing its name and the current modeling time value.

At the transaction input into the block DEPART it leaves the queue, and the interpreter performs the following operations:

- the queue length decreases by B;
- if the transaction was in the queue for zero time it is defined as the transact with zero stay with the subsequent alteration of the zero inputs counter indication;
- "anchoring" of the transaction to the queue is eliminated.

The presence in the model of the "queue"-type objects leads to the realization by the interpreter the calculation of the following statistical data:

- the inputs counter indication;
- the maximum and the average values of the queue length;
- the current queue length value (at the end of modeling phase);
- the average time of the transaction staying in the queue.

Other data can be calculated as well.

For example, let there be a serving device USTR before which the formation of a queue is possible. If there is a need to obtain some historical data about waiting in the queue it can be realized by means of the following model fragment:

```
QUEUE QUSTR
SEIZE USTR
DEPART QUSTR
ADVANCE 5, 3
RELEASE USTR
```

It is obvious, that in the given example all the transactions will pass through the pair of blocks QUEUE - DEPART even if the device USTR is free.

To ensure passing of a transaction into the block distinct from the subsequent one, block TRANSFER is used. There are three basic modes of the TRANSFER block usage.

In the unconditional transfer mode the TRANSFER block has the following format:

TRANSFER, B.

Operand B determines the block position in which the transaction is to jump. The block position is set by the block number or the block mark.

In the statistical mode the transaction transfer into one of the two blocks is realized in a random way. In this mode the TRANSFER block format is set by the following entry:

TRANSFER A, [B], C

Operand A is the probability value of the transaction transfer into the block which position is set by operand C.

Operand B sets the block position in which the transaction transfers with probability 1-A.

The probability value is set accurate within one thousandth. For example,

TRANSFER.435, MUSTR 1, MUSTR 2

With probability 0.435 the transaction will pass into the block with the mark MUSTR 2 and with probability 0,565 – into the block with the mark MUSTR 1.

BOTH mode is determined by the entry

TRANSFER BOTH, B, C.

In this mode the transaction at first aspires to get into the block which is defined by operand B. If it fails, the transaction goes into the block defined by operand C. If the transaction fails to get either into the block defined by operand B or into the block defined by operand C, it stays in the TRANSFER block till the next revision of the current events list and will repeat its transfer attempts in the said order.

All the considered blocks are usually used for modeling the single-channel service devices. For modeling the multi-channel service devices additional blocks are used whose description is available in corresponding literature [21]. However, the description of the considered blocks is enough for understanding the performance principles of GPSS modeling system.

Summary

This chapter is devoted to issues of simulation modeling. The main cases of its application are considered in it as well as advantages and disadvantages of simulation modeling related to specific features of simulation models construction. Different ways of modeling time realization are analyzed. The simulation structures are defined. They are realized by process interaction, event scheduling, activity scanning and three-phase methods. The structural scheme of simulation model is presented. The technological stages of simulation model construction are described in details. They include definition of modeling goals, elaboration of conceptual model, software realization of simulation model, requirements fulfillment verification, design and conduction of experiments, and analysis of modeling results. Additional attention is paid to formalization of experiments designing process and to analysis of the obtained results. The modeling system GPSS is suggested to make the software implementation of the simulation model.

Tasks for case-study

1. Set simulation goals, describe the investigated process, and build the simulation model according to the given description. Ground your decision and run manual simulation.

Let’s imagine that we have a barber's shop with two barbers. We suppose that clients arrive to the barber's shop. If both barbers are busy the clients wait in the single queue. They are served in order of their arrivals. After service all clients leave the barber’s. If the client waits more than 20 minutes he/she leaves the barber’s shop. Let’s have 10 clients. Arrival time and service time are known (table 4.1).

2. Set simulation goals, describe the investigated process, and build the simulation model according to the given description. Ground your decision and run manual simulation.

Let’s imagine that we have a simple telephone system with a single external line. If the line is busy the callers wait. The phone calls are served in order of their arrivals. The maximum waiting time is 5 minutes. Let's have 10 calls. Arrival time and call duration time are known (table 4.2).

Table 4.1 – Arrival time and service time

Client's number	Time interval between clients, min	Service time, min
1	15,6	45,6
2	23,7	42,8
3	38,9	34,7
4	12,5	36,8
5	14,7	28,9
6	16,2	42,0
7	18,9	32,8
8	25,1	43,7
9	34,8	34,7
10	22,6	32,5

Table 4.2 – Arrival time and call duration time

Call's number	Time interval between calls, min	Call duration time, min
1	3,8	5,1
2	3,5	4,5
3	4,2	4,8
4	3,1	3,9
5	2,4	4,7
6	4,3	5,2
7	2,7	5,0
8	2,1	4,5
9	2,5	3,8
10	3,4	5,7

3. Set simulation goals, describe the investigated process, and build the simulation model according to the given description. Ground your decision and run manual simulation.

Let's imagine that we have a gas station. In a gas station there are two gas pumps. Cars arrive at the gas station. Cars are served in order of arrival. Let's have 10 cars. Arrival time and time needed for pumping and paying are known (table 4.3).

Table 4.3 – Arrival time and service time

Car’s number	Time interval between cars arrival, min	Service time, min
1	14,9	19,3
2	15,6	18,4
3	13,9	21,0
4	12,8	17,9
5	14,5	21,3
6	15,9	18,0
7	16,3	16,7
8	15,8	19,2
9	17,0	18,3
10	12,2	21,0

4. Set simulation goals, describe the investigated process, and build the simulation model according to the given description. Ground your decision and run manual simulation.

Let’s imagine that we have a coffee-machine. We suppose that customers arrive and buy coffee. The coffee-machine spends time (service time) to take money, to give change and to make coffee. Customers are served in order of their arrivals. After service every customer goes out. Let’s have 10 customers. Arrival time and service time are known (table 4.4).

Table 4.4 – Arrival time and service time

Customer’s number	Time interval between customers, min	Service time, min
1	3,2	3,8
2	7,8	3,5
3	4,3	4,2
4	2,6	3,1
5	1,9	3,4
6	2,8	4,3
7	5,7	3,7
8	4,1	3,1
9	2,8	3,5
10	4,2	3,4

5. Set simulation goals, describe the investigated process, and build the simulation model according to the given description. Ground your decision and run manual simulation.

Let’s imagine that we have a coffee-machine. The cup of coffee costs 3$. The coffee-machine takes coins of denomination in 1, 2 or 5$ and gives change.

We suppose that customers arrive and buy coffee. The coffee-machine spends 0,5 minute to take money , 0,3 to give change and 1 minute to make coffee. Let's have 10 customers. Arrival time and denomination of their coins are known (table 4.5).

Table 4.5 – Arrival time and denomination of coins

Customer's number	Arrival time, min	Denomination of coins, $
1	15,6	1–2
2	17,2	5
3	19,1	1–1–1
4	21,3	1–1–2
5	23,6	2–2
6	24,9	1–2
7	26,2	2–1
8	27,5	5
9	29,6	2–2
10	33,0	1–1–1

6. Set simulation goals, describe the investigated process, and build the simulation model according to the given description. Ground your decision and run manual simulation.

Let's imagine that we have a coffee-machine. We suppose that customers arrive and buy coffee. The coffee-machine spends about 2 minutes to take money, to give change and to make coffee. If the coffee-machine is busy customers wait in the single queue. Customers are served in order of their arrivals. After service every customer goes out. The maximum waiting time is 10 minutes. Let's have 10 customers. Arrival time is known (table 4.6).

Table 4.6 – Arrival time

Customer's number	Time interval between customers, min
1	14,9
2	15,6
3	13,9
4	12,8
5	14,5
6	15,9
7	16,3
8	15,8
9	17,0
10	12,2

7. Set simulation goals, describe the investigated process, and build the simulation model according to the given description. Ground your decision and run manual simulation.

Let’s imagine that we have a control system of traffic at a crossroad. Let’s consider the control system has 2 states: “opened” and “closed”. The state “opened” defines that traffic on the main street is allowed. The state “closed” defines that the main street is closed and traffic on another street is allowed. Let’s imagine that the main street will be closed when we have 2 cars on another street. Let’s have 10 time moments of car arrival on another street and every car spends about 1 minute on passing (table 4.7).

Table 4.7 – Cars arrival time

Car’s number	Arrival time, min
1	15,6
2	23,7
3	38,9
4	42,3
5	48,5
6	51,4
7	54,6
8	58,9
9	63,5
10	64,8

8. Set simulation goals, describe the investigated process, and build the simulation model according to the given description. Ground your decision and run manual simulation.

Let’s imagine that we have an elevator in the three-floor house. The elevator is called when the person presses the button. When an elevator arrives he/she comes into and selects a floor. If an elevator is busy the person waits. Let’s have 10 persons; time and floor are known (table 4.8). We suppose that customers come into and go out immediately. An elevator spends 1 minute to pass each floor.

9. Set simulation goals, describe the investigated process, and build the simulation model according to the given description. Ground your decision and run manual simulation.

Let’s imagine that we have a crossroad with traffic lights. Traffic lights have three states. They are “stop traffic”, “attention”, and “open traffic”. “Open traffic” state - nobody wants to pass a crossroad. When someone comes, he/she presses a button and traffic lights change the state. At first it moves to the “attention” state and has this state for 1 minute. Then the traffic lights changes to the “stop traffic” state for 1,5 minutes and return to the “open traffic” state. Let’s have 10 people who arrive to crossroad. Their arrival time is known (table 4.9).

Table 4.8 – Arrival time and called floor

Person’s number	Arrival time, min	Initial floor	Required floor
1	1,2	1	2
2	35,2	1	3
3	44,3	1	2
4	56,2	2	1
5	75,3	2	1
6	97,4	3	1
7	110,2	1	3
8	134,6	3	1
9	156,2	1	2
10	186,4	2	1

Table 4.9 – Arrival

Pedestrian number	Arrival time, min
1	1,5
2	3,9
3	6,0
4	8,4
5	11,2
6	14,2
7	17,2
8	20,1
9	23,4
10	28,0

10. The time between consecutive customers arrival at the store is equally allocated in the interval from 1 to 20 minutes. For 50 % of the customers the service time is 8 minutes for another 50 % the service time is 14 minutes. Suggest the appropriate for this case random number generator (chance machine) that will show the arrival time of the customers and another generator that will show the service time. Calculate the work of store during four hours. Determine the overall customers waiting period and the system’s idle time.

11. Develop an algorithm that imitates the game of dice according to the following rule. At the same time two dices with the numbers from 1 to 6 on the faces are being tossed. If the sum of digits on the upper faces equals either 2 or 12 the player receives 250 points, if the sum equals either 3 or 11 – the player

receives 100 points and if the sum is either 4 or 10 the player receives 50 points. If the sum equals from 5 till 9 the gamer losses 50 points. Calculate the expected gain or loss.

12. Everyday Mr. Ivanov is playing the game called “troika” (triplet). In this game the player tosses the coin until the difference between the number of heads and tails will be three. For each toss of the coin the player should pay 1 UAH but with the good round result he will receive 5 UAH. Every day Mr. Ivanov saves 10 UAH for this game. He is playing this game until he spends all money saved for this game or wins the round. Define with the help of simulation model the result of Mr. Ivanov for the week (7 days). Is he going to win?

Check questions

1. Define the notion “simulation modeling”.
2. In what cases is simulation modeling used?
3. In which application areas do we use simulation models?
4. Provide some examples of manufacturing systems where simulation can be applied.
5. Provide some examples of service systems where simulation can be applied.
6. What are the purposes of simulation?
7. What problems can be defined in simulation modeling?
8. Describe advantages and disadvantages of simulation.
9. Describe the order of simulation modeling of the complex system component.
10. What is an activity of simulation model?
11. Describe the mechanisms of model time changing.
12. How can we consider the time flow in the simulation model?
13. What are the modeling structures for simulation study?
14. What is the difference between process-interaction and event-scheduling methods?
15. What is the difference between two and three-phase methods?
16. What parts are distinguished in simulation model structure?
17. What stages are defined in the process of simulation model elaboration?
18. What is the sense of the stage of conceptual model construction?
19. What is the sense of verification and validation processes?
20. What is the main problem of experiments design and how is it solved?
21. Provide the classification of factors considered during experiments design with simulation model.
22. How is the number of factors defined?

23. How is the accuracy and validity of estimates obtained as a result of experiments defined?
24. How is the number of factors defined?
25. How is the number of experiments necessary for obtaining the given accuracy and validity defined?
26. How is the rule of automated experiment finish realized?
27. What defines the choice of methods of experiments' results processing?
28. What are the requirements to the quality of parameters' estimates obtained during experiments' results processing?
29. What for are different methods of correlation, regression and variance analyses are used during experiments' results analysis?
30. What problems are solved by different modeling languages?
31. What objects are referred as static and dynamic in GPSS modeling system?

5. BUSINESS-PROCESSES MODELING

5.1. Business-process definition

A successful work of a company on the market depends on many factors, for example, the variety of suggested services, market's saturation, marketing policy. To be competitive modern companies are oriented on constant development. Enterprises have to improve their activities that requires elaboration of new technologies and ways of business conduction, introduction of efficient management methods. That is why it is necessary to use business-processes modeling methodology.

Nowadays methods and instruments of business-processes modeling are the serious direction of scientific researches and the part of prosperous software market at the same time. The range of modeling methods for business-processes description is quite big. They include some simple graphical notations used for flowcharts construction, mathematical tools like Petri nets [26], object-oriented modeling languages like UML (Unified Modeling Language) [27, 28], and some special methodologies for business systems description like XPDL (XML Process Definition Language) [29] and BPEL (Business Process Execution Language) [30].

A *business-process* is a system of consequent, dedicated and regulated types of actions in which the process inputs are transformed into outputs (process results that are valuable for consumers) by control action and with the help of resources [31]. Operations are executed by different elements of enterprise's organizational structure according to definite business-rules. Business-rules are the ways of business-functions realization in the business-process scope as well as the characteristics and conditions of its execution [32].

The methods of business-processes modeling and analysis are important tools of business efficiency increasing. Their usage is intended on reorganization of business-processes and as a result on gain growth, costs reduction, products quality rise, optimal usage of floating capital, introduction of automation systems and so on.

Before making an enterprise’s work better it is necessary to analyze how it is functioning in whole, how it interacts with external organizations, customers, suppliers and how the activities at each working place are organized.

A business-process is a collection of related, structured, interconnected activities or tasks that produce a specific product or service for customers.

There are three types of business processes:

1. Management processes, the processes that govern the operation of a system. Typical management processes include quality management and strategic management.
2. Operational processes, processes that constitute the core business and create the primary value stream. Typical operational processes are purchasing, manufacturing, advertising and marketing, and sales.
3. Supporting processes, which support the core processes. Examples include accounting, recruitment, technical support.

According to another classification that comes from complex systems modeling there are the following types of business-processes [32]:

- functional processes, that describe the collection of system’s functions, their inputs and outputs;
- behavioral processes, which represent when and/or under which conditions business-functions are executed with the help of such categories as system’s state, event, transition from one state to another, condition of transition, events sequence;
- structural processes, that characterize system’s morphology – a composition of subsystems and their interconnections;
- informational processes, which reflect data structures, their composition and interconnections.

A business-process has the following characteristics:

- definability: it must have clearly defined boundaries, input and output;
- order: it must consist of activities that are ordered according to their position in time and space;
- customer: there must be a recipient of the process' outcome, a customer;
- value-adding: the transformation taking place within the process must add value to the recipient, either upstream or downstream;
- embeddedness: a process can not exist in itself, it must be embedded in an organizational structure;
- cross-functionality: a process regularly can, but not necessarily must, span several functions.

Any business-process starts from the costumer's demand and is finished by his/her satisfaction. The process-oriented organizations try to eliminate barriers on the junctions of two different organizational departments during the execution of shared business-process.

Business-processes must be organized in a way to create the value for their consumers and to eliminate any unnecessary actions. The output of business-process increases the value for a customer and profitability of producing commodity or service.

A business process can be decomposed into several sub-processes, which have their own attributes, but also contribute to achieving the goal of the super-process. The analysis of business processes typically includes the mapping of processes and sub-processes down to activity level.

An organization is considered as business-processes system that is intended on obtaining the final result (producing commodity or service). The consumers of business-process can be either internal or external clients. The aim of business-processes formalization is the estimation of their efficiency and effectiveness.

5.2. Business-processes modeling standards and technologies

Business-modeling is an activity of elaboration of enterprises' models which include business objects description (departments, positions, resources, roles, processes, operations, information systems, information carriers) and definition of connections between them. Requirements to these models and their content are defined by modeling goals.

Also business-modeling is the name of discipline and a separate subprocess in the process of software development. Enterprise's activities are described in this subprocess. This leads to system requirements, which define the processes and operations that should be automated.

A business-process model is a formalized (graphical, tabular, text, symbolic) description of a business-process that expresses a real or supposed enterprise's activity.

As a rule a model contains the following information about business-process:

- a set of steps that form a business-process – business-functions;
- an order of business-functions execution;
- mechanisms of control and management in the scope of a business-process;
- executers of each business-function;

- input documents or information used by each business-function;
- output documents or information generated by each business-function;
- resources needed for business-functions execution;
- documents or conditions that regulate business-functions execution;
- parameters that characterize business-function and process execution.

Business-modeling is carried out in one of two directions [33]. The first one is construction of AS-IS business-process model. An AS-IS model is critically analyzed and is processed by special software to be improved. The second direction of business-modeling is construction of TO-BE model, which reflects business-process efficiency from the point of selected criteria.

An AS-IS model is built after intensive investigation of an enterprise. It defines what different departments do and what their contribution to a process is. An AS-IS model helps to understand what is going on at the enterprise. It formalizes the existing processes, allows to detect weak points, redundant actions and unreasonable resources usage. In addition an AS-IS model is a basis for automation problem statement, i.e. for information system elaboration and deployment.

An information system deployment will inevitably lead to the reconstruction of existing business-processes at an enterprise. Analysis of a functional model allows understanding advantages and disadvantages of new business-processes. Also it shows how much the business structure will change. Business-processes specification allows to reveal organization's disadvantages even when the functionality is obvious.

The indicators of inefficient activities can be the following: useless, uncontrolled and duplicating works, inefficient documents circulation (the necessary document doesn't occur in some place at particular point of time), absence of a feedback in management (work's result doesn't influence the execution of works), lack of feedback to input (unreasonable usage of objects or information).

The disadvantages that have been found in an AS-IS model can be managed during TO-BE model elaboration, which gives a new model of business-processes organization. A TO-BE model is needed for estimation of effects of information system deployment and for analysis of alternative/better ways of works execution and documenting a future enterprise's functioning.

There are different methods and techniques applied for business-processes modeling. Among them we can distinguish the methods of structural analysis and design, object-oriented analysis and design, Business Process Model and Notation (BPMN), BPEL, Business Process Modeling Language (BPML), Event-driven Process Chain (EPC) and others. Let's consider them in detail.

Structural analysis is a method of research of system's static characteristics by extracting subsystems and elements of different hierarchical levels and defining relationships and connections between them. The essence of structural approach of model's elaboration is in division of a system into parts – "black boxes", and in hierarchical organization of these "black boxes". There is no need to know how "black boxes" work, it is necessary only to have information about their inputs and outputs and about performed functions. This is an advantage of using "black boxes".

Structural analysis and design of business-processes uses different models describing [34]:

- system's functional structure;
- sequence of executed actions;
- information transmission between functional processes;
- relations between data.

The most widespread models of the first three groups are the following:

- functional model SADT (Structured Analysis and Design Technique);
- IDEF3 (Integration Definition for Function Modeling) model;
- DFD (Data Flow Diagram).

SADT is a collection of rules and procedures needed for building a functional model of an object of any domain. SADT technology (renamed in IDEF0) is traditionally used for organizational systems modeling. The advantages of SADT models applying for business-processes description are the following:

- the completeness of business-process description (management, information and material flows, feedbacks);
- strict requirements of method which gives the models of a standard view;
- the correspondence of processes description approach to ISO 9000 standards.

IDEF is a widespread family of modeling methods of organizational systems. It has become a basis for many normative documents in different countries. Nowadays IDEF family includes the following methodologies [32, 35]:

- IDEF0 – methodology of functional modeling, which has a graphical language and allows representing a modeling system as interconnected functions. As a rule IDEF0 modeling is the first step of studying a system.
- IDEF1 –methodology for modeling information flows inside a system, which allows expressing and analyzing their structure and interconnections.

- IDEF1X (IDEF1 Extended) – modeling methodology, which is applied for information model construction that represent a structure of information needed for support of enterprise's functions. IDEF1X is often used for modeling relational databases.
- IDEF3 – methodology of processes description. Scenarios and sequences of operations of each process are described with the help of IDEF3. Although IDEF3 is not a standard, this technology is widespread as an addition to IDEF0: each function of IDEF0 can be represented as a separate process by IDEF3.
- IDEF4 – methodology of object-oriented design. IDEF4 implements an object-oriented analysis of complex systems. It gives a graphical language for classes' presentation, an inheritance diagram and methods' taxonomies.
- IDEF5 – methodology of ontological research of complex systems. A system's ontology can be described with the help of definite dictionary of terms and rules. On their basis system's state can be formalized at particular moment of time. Basing on system's description the conclusions about further system's development is formed and its optimization is done.

DFD is a hierarchy of functional processes that are connected with data flows. The aim of such representation is to demonstrate how each process transforms its input data in output data and to show relationships between these processes.

A system's model is defined as a hierarchy of data flow diagrams that describe an asynchronous process of information transforming from its input to a system to its output for a consumer. DFD was created as a tool for information systems design (while SADT – as a tool for system's modeling in a whole). DFD has richer set of elements that reflect a specificity of information systems (for example, data stores are prototypes of files or databases, external entities reflect system's interaction with outer space).

The above mentioned tools of structural analysis are almost similar from the point of view of graphical modeling abilities. That is why the main criterion for their selection is an extent to which an analyst can use them.

The conceptual basis of object-oriented analysis and design (OOAD) is an object model. Its main principles (abstraction, encapsulation, modularity and hierarchy) and notions (object, class, operation, interface) are formulated by G.Booch [27, 28].

Most of the modern OOAD methods are based on usage of UML (Unified Modeling Language) [36]. UML is a language for definition, representation, design and documenting of software, organizational economic, technical and other systems. UML contains a standard collection of different diagrams and notations. They can be divided into those ones that model a static system's structure (structural model) and those ones that model a dynamic system's structure (dynamic model). A static model fixes entities and structural relations between them. A dynamic model shows how entities interact to form a necessary system's behaviour.

Structure diagrams include the following:

- class diagram;
- composite structure diagram;
- component diagram;
- deployment diagram;
- object diagram;
- package diagram;
- profile diagram.

Behaviour diagrams include the following:

- activity diagram;
- use-case diagram;
- state machine diagram;
- interaction diagrams:
  - sequence diagram;
  - communication diagram;
  - interaction overview diagram;
  - timing diagram.

Except OOAD technologies, other methods and languages can be used for business-processes modeling, for example, EPC, BPMN, BPEL, BPML.

Event-driven Process Chain (EPC) is a type of flowchart used for business process modeling [37]. EPC can be used for configuring an enterprise resource planning (ERP) implementation, and for business process improvement.

An EPC consists of the following elements:

- Functions. The basic building blocks are functions. A function corresponds to an activity (task, process step) which needs to be executed.
- Events. They describe the situation before and/or after a function is executed. Functions are linked by events. An event may correspond to the postcondition of one function and act as a precondition of another function.
- Logical connectors. Connectors can be used to connect activities and events. This way the flow of control is specified. There are three types of connectors: “and”, “exclusive or” and “or”.

Organization Business Process Management Initiative (BPMI) has elaborated a special Business Process Model and Notation (BPMN) for business-processes modeling [38].

BPMN specification describes symbolic notation for expressing business-processes in the form of diagrams.

There are three types of submodels in the scope of general BPMN model [39]:

- Private – represent internal processes of organization;
- Abstract – represent interactions between internal business-process and another process or participant on the level of messages exchange, without description of details of other participants or processes;
- Collaboration – expresses interaction between two or more participants by message exchange. In fact it is supposed that there are two or more abstract processes on a single diagram.

BPMN supports only a set of concepts needed for business-processes modeling. Modeling of other aspects, like data model and organizational structure, are out of BPMN scope.

BPMN supports the properties of graphical objects which make possible generation of executable BPEL code. BPMN is mapped directly at business-processes languages like BPEL and BPML.

Business Process Execution Language (BPEL) is a language based on XML (Extensible Markup Language) for formal business-processes description and protocols of their interaction. While BPMN gives a modeling notation, BPEL is a language of processes' execution. BPEL extends a model of web-services interaction and includes transaction support into this model.

There are several original design goals associated with BPEL:

- Define business processes that interact with external entities through web service operations defined using WSDL 1.1 (Web Service Definition Language).
- Define business processes using an XML-based language.
- Define a set of web service concepts that are meant to be used by both the external (abstract) and internal (executable) views of a business process.
- Provide both hierarchical and graph-like control regimes, and allow their use to be blended as seamlessly as possible.
- Provide data manipulation functions for the simple manipulation of data needed to define process data and control flow.
- Support the implicit creation and termination of process instances as the basic lifecycle mechanism.

- Define a long-running transaction model to support failure recovery for parts of long-running business processes.
- Use Web Services as the model for process decomposition and assembly.

Business Process Modeling Language (BPML) is an XML-based language which defines a formal model for expressing abstract and executable processes that address all aspects of enterprise business processes [40]. BPML defines activities of varying complexity, transactions and their compensation, data management, concurrency, exception handling and operational semantics. BPML also provides a grammar in the form of an XML Schema for enabling the persistence and interchange of definitions across heterogeneous systems and modeling tools.

BPML is a rich language for both – simple and complex business-processes description. Since PBML and BPEL are languages with block structure, they have a similar set of expressions and syntax. Nevertheless BPML has wider possibilities than BPEL. BPML syntax support operations and their types, processes, properties, signals, timetables and nonstandard situations.

5.3. CASE-tools for business-processes modeling

Various software products of leading designers of information technologies market allow creating understandable, readable and easy-to-use visual business-processes models. We can mention the most popular CASE-tools for business-processes modeling on the basis of structural and object-oriented approaches. They include:

- Visual Paradigm;
- Sybase PowerDesigner;
- AllFusion Process Modeler;
- ER/Studio Business Architect;
- Rational Rose;
- Microsoft Office Visio.

Visual Paradigm software includes:

- Visual Paradigm for UML (VP-UML) – a tool for visual modeling of applications, documents, source code and databases generation with the help of UML.
- Smart Development Environment is a software that extends the possibilities of such development environments as Visual Studio, Eclipse, NetBeans and IntelliJ IDEA by UML and ER (Entity/Relationship) diagrams and functions of engineering of a source code.

- DB Visual ARCHITECT is a tool of databases design that supports functions of databases creation with relational diagrams and UML class diagrams.
- Business Process Visual ARCHITECT is a solution for business-processes modeling and animation, reports generation, projects publishing. This tool supports BPMN 2.0 standard. It allows building data flow diagrams and organizational diagrams.
- Agilian – a tool for construction of software development diagrams.
- Teamwork Server is a single store of software projects for collective designers' access.

Sybase PowerDesigner is a tool for business applications creation including business-processes modeling, conceptual and physical database design and UML modeling. It gives a central repository for models and objects storing. The main peculiarities of this software include:

- Business-processes modeling on the basis of control flows diagrams.
- Data modeling (conceptual and physical models) based on ER standard, including technologies of data warehouses modeling ("star" and "snowflake" schemes, multidimensional modeling, connection to a particular data source).
- Standard UML diagrams.
- Requirement model, which is a special model for documenting and analysis of information system requirements.
- Modern graphical adjustable user interface that contains:
  - improved model's management, including objects, models and databases synchronization;
  - reports generator, extended and independent from a model, which allows obtaining a document with information about several models.

AllFusion Process Modeler is a tool for modeling, analysis, documenting and optimization of business-processes. It can be used for graphical representation of business-processes. A graphical scheme of work execution, information exchange and documents circulation visualizes a business-process model. Graphical representation of this information allows translating a problem of organization management into an engineering technology.

AllFusion Process Modeler has the following peculiarities:

- Supports three standard notations: IDEF0, DFD and IDEF3.
- Increasing business efficiency, optimization of any procedures in a company.
- Supports methods of costs calculation by the volume of economical activity (value analysis, ABC analysis).
- Inexpensive, widespread and easy-to-use tool.

- Allows to simplify certification on correspondence to quality standards ISO 9000.
- Integration with ERwin Data Modeler (for databases modeling).
- Integration with simulation modeling tool Arena.
- Has report generation.
- Allows efficient manipulation with models (junction and splitting).
- Has a wide range of models and projects documenting tools.

ER/Studio Business Architect is a CASE-tool for graphical representation of business starting from basic concepts and finishing at processes, that describe company's activities. ER/Studio Business Architect allows modeling business-processes and data used by them.

The main facilities of ER/Studio Business Architect include the following:

- business-process modeling with the help of standard elements, such as sequences, tasks, swimlanes;
- support of high-level conceptual modeling with the help of domains, business-parts, interactions and other elements;
- powerful intuitively obvious interface with organizational hierarchies for easy-to-use information organization;
- usage of BPMN notation.

ER/Studio Business Architect provides documenting and analysis of business-processes, simplifies their checking, verification and analysis.

Rational Rose is a CASE-tool of Rational Software Corporation (the USA). It is intended for automation of analysis and design stages of software elaboration. Also it provides program code generation and project documentation. Rational Rose uses OOAD methodology based on UML.

The advantages of Rational Rose include the following:

- UML support.
- Program code generation.
- Inverse design of existing systems.
- Ability to be integrated with development environments (Visual Studio).
- Tools for automated control, including checking the correspondence
- Multiplatform tool.

Microsoft Office Visio is tool for modeling of data and applications. It supports creation of data models and object-oriented applications modeling. Visio has a big collection of patterns: flowcharts of business-processes, network schemes, workflow diagrams, databases models and software development diagrams. They can be used for visualization of business-processes, controlling workflows and resources usage, systems optimization, organizational structure schemes.

BPMN 2.0 Modeler for Visio is an autonomous extension of Visio, which provides business-processes models creation. BPMN 2.0 Modeler supports a set of elements of BPMN 2.0 notation.

So existing CASE-tools provide business-processes modeling with the help of different notations and standards. Selection of particular CASE-tool is defined by the modeling method, license cost of software, its ability to be integrated with other analysis and design tools, and personal preferences of an analyst.

5.4. Methodology of business-processes structural analysis and design

5.4.1. METHODOLOGY OF FUNCTIONAL MODELING IDEF0

IDEF0 is a technology of describing a system as a set of interconnected activities, or functions. IDEF0 is functional-directed, i.e. system's functions are researched independently from objects that provide their execution. Such functional point of view helps to separate aspects of system's purpose from aspects of its physical implementation.

IDEF0 is based on a simple graphical notation (it consists of only two designations: blocks and arrows) with strict recommendations how to use it. This notation provides building an understandable system's model (table 5.1).

Table 5.1 – Elements of IDEF0 diagram

Designation	Name	Description
□	Function or Activity Box	An action performed in business-process
→	Arrow	Represents input, output, control, mechanism of a functional block
[]	Arrow Tunnel	Tunnel designation around an arrow's start means that this arrow was not inherited from functional parent block and appeared at current diagram. Tunnel designation around an arrow's end near the receiver block means that this arrow will not be considered at the child diagram

Modeling business-process in IDEF0 notation is represented as a set of hierarchically ordered business-functions (or actions).

Each function processes or transforms input parameters (raw materials, information, etc.) into output ones. Since IDEF0 models represent a system as a set of hierarchical functions, the first function that has to be defined is a context function, which describes a system in whole. Functions are depicted in the form of named rectangles, or functional blocks, at the diagrams (fig. 5.1).

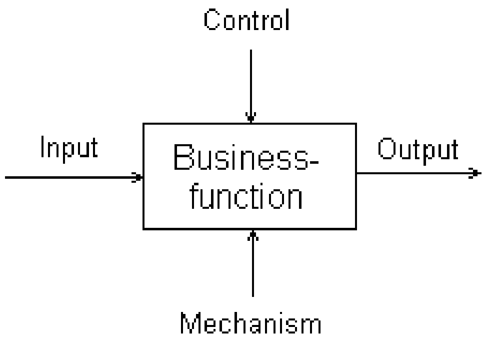


Figure 5.1 – Functional block in IDEF0

Any functional block definition must include at least description of objects that a block creates as a result of its work (output) and objects that a block transforms (input). In addition IDEF0 also models control and mechanisms (fig. 5.1). Control is made by objects that influence the way how a block transforms inputs into outputs. Mechanisms are the objects which directly perform this transformation, but are not consumed by themselves.

To represent these categories of information on IDEF0 diagrams the ICOM acronym is used:

- I – Input, what is consumed during process execution;
- C – Control, restrictions and instructions that influence process execution;
- O – Output, result of process execution;
- M – Mechanism, what is used for process execution, but is not consumed in this process.

A functional block output can be used in other blocks. The main IDEF0 advantage is that this methodology allows to reveal interconnections between system's functional blocks. IDEF0 provides splitting and junction of arrows on the diagram. As rule, arrows, that connect functional blocks, have names that differ from the source arrow name. An output and broken (or joint) arrows are called connected ones. Such a technique is usually applied to express the usage of a part of raw materials or information designated by a source arrow in a process. The similar approach is applied for joint arrows.

The notion of connected arrows is used for managing the level of diagram's refinement. If one of the arrows is absent on the parent diagram (for example, because it is not essential for parent level) and is not connected with other arrows of the same diagram, the point of this arrow entrance to the diagram or exit from the diagram is designated by the tunnel (Table 5.1).

IDEF0 implements three basic principles of business-processes modeling:

- principle of functional decomposition;
- principle of complexity restriction;
- context principle.

According to the principle of functional decomposition a complex business-function can be presented as a collection of simpler functions that compose it. These simple functions can be decomposed as well.

According to the principle of complexity restriction the number of functional blocks on one diagram must be bigger than two and less than six. This provides IDEF0 diagrams' understandability. This principle helps to create well-structured business-processes models.

The context principle means that business-process modeling starts from the context diagram construction. There is only one block on it that expresses the main business-function of a system. When you define the main function, it is necessary to take a modeling goal into account. A context diagram fixes boundaries of a business-system defining how it interacts with environment. This is done by description of arrows that are connected with main business-function block.

The result of IDEF0 applying is a system's model that is made up of hierarchical set of diagrams, documents and dictionaries interconnected with the help of cross references. The most widespread CASE-tools providing IDEF0 notation include AllFusion Process Modeler (earlier BPwin) and Microsoft Office Visio.

5.4.2. IDEF3–MODELING OF SYSTEM'S BEHAVIOUR

IDEF3–modeling completes traditional modeling with the help of IDEF0 standard. Nowadays it spreads as a way of business-processes models design.

The basis of IDEF3 model is a so-called business-process scenario, which defines the sequence of actions or subprocesses of analyzed system. Since a scenario defines the purpose and the scope of a model, the important point is the choice of appropriate names for actions' designation. To choose a name it is recommended to use verbs and verbal nouns.

An action, or Unit of Work – UOW, or Unit of Behaviour (UOB) in IDEF3 in IDEF3, is an important model's component. IDEF3 diagrams describe an action in the form of rectangle (fig. 5.2). Each action has a unique number. This number is not used again even if an action is eliminated during model's construction. The action's number is anticipated by parent's number.

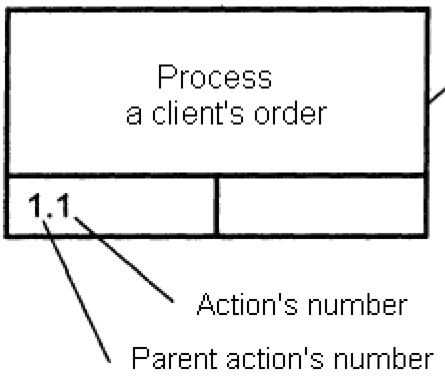


Figure 5.2 – Action in IDEF3

Links mean the essential interactions between actions (table 5.2). All links in IDEF3 are unidirectional. Usually diagrams are organized in a left-to-right way although an arrow may start or end in any side of a block that represents action. This means that arrow starts at the right side of a block and ends at the left one.

Table 5.2 – Links types in IDEF3 model

Designation	Name	Description
⟶	Temporal precedence	A source action must be finished before an end action starts
↠	Object flow	An output of a source action is an input of an end action. This also means that a source action must be finished before an end action starts
⇢	Relationship	A type of interaction between a source and an end actions is set by the analyst for each case of this kind of relationship usage

The end of one action may initiate the start of several other actions execution, or a particular action to be started may require the end of several other actions. Junctions split or join the internal flows and are used for designation of process branching (table 5.3). There are two types of junctions: expanding and reducing. Expanding junctions are used to split a flow. The end of one action causes the start of execution of several other actions. Reducing junctions join the flows. The end of one or several actions causes the start of only one another action.

There are some cases when start or end time of executed actions must be the same, i.e. action must be executed synchronously. To model such kind of system's behaviour synchronous junctions are used (table 5.4).

Table 5.3 – Junctions types in IDEF3 model

Designation	Name	Type	Description
&	AND	Expanding	Each end action is obligatory initiated
		Reducing	Each start action must be obligatory finished
X	Exclusive OR	Expanding	One and only one end action is initiated
		Reducing	One and only one start action must be finished
O	OR	Expanding	One or more end action are initiated
		Reducing	One or more start actions must be finished

Table 5.4 – Synchronous junctions in IDEF3 model

Designation	Name	Type	Description
&	AND	Expanding	All actions start simultaneously
		Reducing	All actions finish simultaneously
O	OR	Expanding	Several actions may start simultaneously
		Reducing	Several actions may finish simultaneously
X	Exclusive OR	Expanding	Simultaneous start of actions is impossible
		Reducing	Simultaneous end of actions is impossible

Pointers are used on IDEF3 diagrams to attract user's attention to some important model's aspects. Pointers are special symbols that refer to some aspects of process description (table 5.5).

Table 5.5 – Pointers types in IDEF3 model

Pointer type	Description
OBJECT	Describes an important object that takes part in action
GOTO	Is used for cyclic action's execution implementation. It can refer to junctions as well
Unit of Behavior	Is used for additional instance placement of existing action on the diagram without cycling

Continuation of table 5.5

Pointer type	Description
NOTE	Is used for documenting any important information that refers to diagram. In this case GOTO is an alternative to the placement of any text notes on the diagrams
Elaboration – ELAB	Is used for detailed description of diagram content. This kind of pointers is usually used for description of branching logics of junctions

Actions in IDEF3 may be decomposed into components for detailed analysis. An action can be decomposed several times. This provides documenting of alternative process flows in one model.

To create IDEF3 diagrams the following CASE-tools can be used: AllFusion Process Modeler, Microsoft Office Visio.

5.4.3. DATA FLOWS MODELING

As well as IDEF0 diagrams Data Flow Diagrams model a system as a set of actions connected with arrows. DFD also contain two new types of objects: objects that collect and store information – data stores, and objects that model the interaction with system's parts (or other systems) – external entities (table 5.6).

Unlike the arrows in IDEF0, which illustrate relations, arrows in DFD show how object (including data) are really transferring from one action to another. This representation together with data stores and external entities provides the expression of physical system's characteristics in DFD models. Physical characteristics include objects transferring (data flows), objects storing (data stores), sources and consumers of objects (external entities).

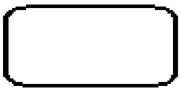
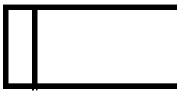
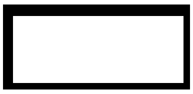
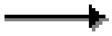
Unlike in IDEF0 the names of objects on DFD diagrams are usually nouns. Context DFD-diagram often consists of one functional block and several external entities. This functional block has a name that coincides with the name of the whole system.

Arrows may be branched on DFD-diagrams. Each segment may be renamed to show the decomposition of data that is transferred by the part of flow. Also arrows may join to form so-called complex objects.

AllFusion Process Modeler and Microsoft Office Visio are the CASE-tools that support DFD notation.

So IDEF0, IDEF3 and DFD diagrams are convenient tools for business-processes modeling. Each notation can be used either independently or together with one another. For example any functional block of IDEF0 can be represented as a sequence of processes in IDEF3.

Table 5.6 – Elements of DFD-diagram

Designation	Name	Description
	Function, Process	Functional block transforms input data into output
	Data store	Represent any mechanism that provides data storing for its intermediate processing
	External Entity	Provides necessary inputs for a system and/or is a receiver for its inputs. One external entity may give inputs (as a supplier) and receive outputs at the same time
	Data flow	Arrows describe objects flow from one part of a system to another. Also bidirectional arrows are used. They reflect interactions between blocks

5.5. Business-processes object-oriented analysis and design

UML is a successor of object-oriented analysis and design methods, which appeared in late 1980s – beginning of 1990s. UML has the following main goals to be achieved as a result of its elaboration:

- provide users with a ready-to-use, expressive visual modeling language so they can develop and exchange meaningful models;
- provide extensibility and specialization mechanisms to extend the core concepts;
- be independent of particular programming languages and development processes;
- provide a formal basis for understanding the modeling language;
- encourage the growth of the object-oriented tools market;
- integrate best practices.

UML was accepted as a standard modeling language and was encouraged by software industry.

The main advantage of object-oriented approach is that it object-oriented systems are open and easily yield to changes, because their construction is based on stable forms. This allows system to develop gradually and does not lead to its total remaking after some changes. In addition, object-oriented approach substantially increases the level of development unification and reuse.

An object model is quite natural, because it is oriented on a man’s perception of world. Also an object model provides usage of object-oriented programming languages.

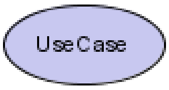
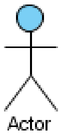
The disadvantage of object-oriented approach is in its high initial costs. Object decomposition differs from the functional one much. That is why transition to a new technology is connected with additional financial costs and overcoming psychological difficulties.

This explains the fact that structural approach is still widely used and important. UML example shows that its authors borrowed many reasonable aspects of structural approach: an element of functional decomposition in use-case diagrams, state diagrams, activity diagrams, etc. It is obvious that it is impossible to manage a complex system's project only by means of one decomposition.

Use-case diagrams and activity diagrams are often used for business-processes modeling (table 5.7, 5.8).

Use-case diagrams play the main role in modeling system's, subsystem's or class behaviour. This kind of diagrams shows a set of use-cases, actors and relations between them (table 5.7).

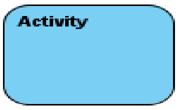
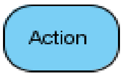
Table 5.7 – Use-case diagram's elements

Designation	Name	Description
	Use-case	A set of consequent actions that are executed to provide an actor with a definite result
	Actor	An interconnected set of roles that are played by use-cases' users during their execution
- - - - - - - ->	Dependency	A relation of usage according to which a change of one element may influence another element that uses the first one. The reverse statement does not hold true
————▷	Generalization	A relation between a parent entity and its child entity
————	Association	A structural relation which shows that objects of one type are connected with objects of another type

It is possible to apply Include and Extend types of relations between use-cases. Include relation means that behaviour of some use-case is incorporated at some point of a basic use-case. An included use-case never exists autonomously. Extent relation means that a basic use-case contains behaviour of another use-case implicitly at a point that is set by extending use-case indirectly. A basic use-case may be autonomous, but its behaviour is extended by another one under definite circumstances.

Activity diagrams can be used for modeling of business-processes and workflows. This kind of diagrams contains activities and actions, transitions and objects (table 5.8).

Table 5.8 – Activity diagram's elements

Designation	Name	Description
	Activity	A continuous nonatomic step of a process
	Action	An atomic step that changes system's state or returns some value
◇	Decision node	Describes different ways of execution depending on the value of Boolean expression
●	Initial node	Denotes a process start
◉	Activity final node	Denotes a process end
⟶	Activity	A continuous nonatomic step of a process

An action cannot be decomposed as they are atomic. This means that events may happen inside of them, but an executed in an action work cannot be interrupted.

As opposed to this, activities can be decomposed. This allows presenting an activity with the help of other activity diagrams. Activities are nonatomic, so they can be interrupted.

It is possible to use parallel flows in activity diagrams for business-processes modeling. To denote a split or join of execution flows a synchronization bar (a bold vertical or horizontal line) is used.

Also swim lanes can be used in activity diagrams. They allow to break an activity into groups that represent, for example, a department responsible for work.

So UML tools enable business-processes modeling. The main advantages of this technology reveal during business-processes automation. There is a wide range of CASE-tools that provide UML-modeling. They include Rational Rose, Visual Paradigm, Sybase Power Designer, ER/Studio Business Architect.

5.6. Business-processes graphical representation on the basis of BPMN

BPMN (Business Process Model and Notation) provides to its users a possibility to elaborate understandable representations of business-processes.

BPMN gives a set of diagrams for people who design and manage business-processes.

BPMN provides businesses with the capability of understanding their internal business procedures in a graphical notation and will give organizations the ability to communicate these procedures in a standard manner. Currently, there are scores of process modeling tools and methodologies. That is why it is likely that business analysts need to understand multiple representations of business-processes – potentially different representations of the same process as it moves through its lifecycle of development, implementation, execution, monitoring, and analysis. Therefore, a standard graphical notation will facilitate the understanding of the performance collaborations and business transactions within and between the organizations. This will ensure that businesses will understand themselves and participants in their business and will enable organizations to adjust to new internal and business circumstances quickly. BPMN follows the tradition of flowcharting notations for readability and flexibility.

Since a BPMN diagram may depict the processes of different participants, each participant could view the diagram differently. That is, the participants have different points of view regarding how the processes will apply to them. Some of the activities will be internal to the participant (meaning performed by or under control of the participant) and other activities will be external to the participant. Each participant will have a different perspective as to which are internal and external. However, the diagram itself remains the same. It is open to the modeler or modeling tool vendor to provide any visual cues to emphasize this characteristic of a diagram.

There are five categories of elements in BPMN:

1. Flow objects:

- Events;
- Activities;
- Gateways.

2. Data:

- Data Objects;
- Data Inputs;
- Data Outputs;
- Data Stores.

3. Connecting Objects:

- Sequence Flows;
- Message Flows;
- Associations;
- Data Associations.

4. Swimlanes:

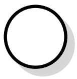
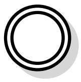
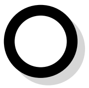
- Pools;
- Lanes.

5. Artifacts:

- Groups;
- Text Annotations.

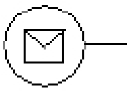
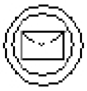
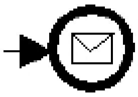
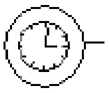
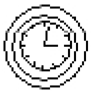
There are three types of events in BPMN: start, intermediate and end event (table 5.9). Events influence a process flow, they may be a reason or a result of a process.

Table 5.9 – Event types in BPMN model

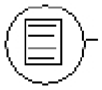
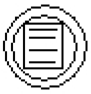
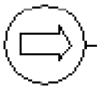
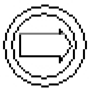
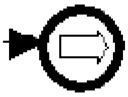
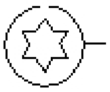
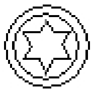
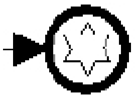
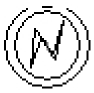
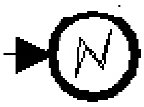
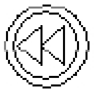
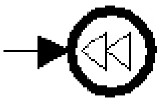
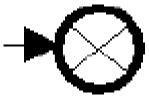
Designation	Name	Description
	Start Event	Starts a process flow
	Intermediate Event	Happens during a process execution
	End Event	Finishes a process flow

Triggers are used for modeling more complex business-events, like messages, timers, business-rules, errors, etc (table 5.10).

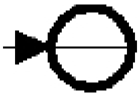
Table 5.10 – Event trigger types in BPMN model

Trigger	Start event	Intermediate event	End event	Description
Message				A start message arrives from a participant and triggers the start of the process, or continues the process in the case of an intermediate event. An end message denotes a message generated at the end of a process
Timer			A Timer cannot be an end event	A specific time or cycle can be set to trigger the start of the process, or continue the process in the case of an intermediate event

Continuation of table 5.10

Trigger	Start event	Interme-diate event	End event	Description
Rule			A Rule cannot be an end event	Triggers when the conditions for a rule become true
Link				A link is a mechanism for connecting the end event of one process flow to the start event of another process flow
Multiple				For a start multiple event, there are multiple ways of triggering the process, or continuing the process in the case of the intermediate event. Only one of them is required. The attributes of the event define which of the other types of triggers apply. For end multiple, there are multiple consequences of ending the process, all of which will occur
Exception	An Exception cannot be a start event			An end exception event informs the process engine that a named error should be generated. This error will be caught by an intermediate exception event
Compen-sation	A Compen-sation event cannot be a start event			An end compensation event informs the process engine that a compensation is necessary. This compensation identifier is used by an intermediate event when the process is rolling back
End	An End event cannot be a start event	An End event cannot be an intermediate event		An end event means that the user has decided to cancel the process. The process is ended with normal event handling

Continuation of table 5.10

Trigger	Start event	Interme-diate event	End event	Description
Kill	An End Kill event cannot be a start event	An End Kill event cannot be a intermediate event		An end kill event means that there is a fatal error and that all activities in the process should be immediately ended. The process is ended without compensation or event handling

There are three types of activities in BPMN: a process, a subprocess, a task (fig. 5.3). Their names reflect the hierarchical relationships between them. Each one is designated by a rectangle. Tasks cannot be decomposed.

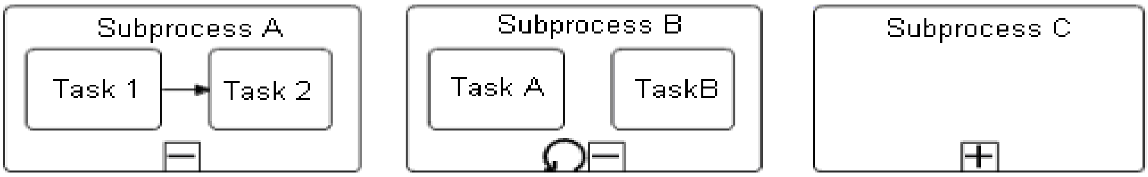
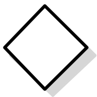
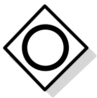


Figure 5.3 – Subprocesses examples (A – ordinary detailed, B – repeated detailed, C – ordinary hidden)

Gateways are used to denote decision nodes. Gateways define a direction of flow basing on data (data-based) or basing on event's occurrence (event-based). Table 5.11 contains the main gateway types.

Data Objects provide information about what activities require to be performed and/or what they produce. Data Objects can represent a singular object or a collection of objects. Data Input and Data Output provide the same information for processes. Data Objects are presented in table 5.12.

Table 5.11 – Types of gateways in BPMN model

Designation	Name	Description
	Exclusive Decision (XOR)	Used to create alternative paths within a process flow
	Inclusive OR Decision	Used to create alternative but also parallel paths within a process flow

Continuation of table 5.11

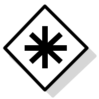
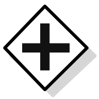
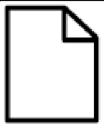
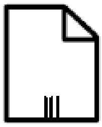
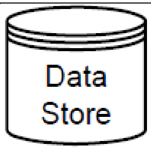
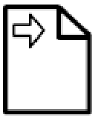
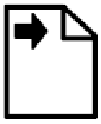
Designation	Name	Description
	Complex Decision	Used to model complex synchronization behavior
	Parallel Forking (Joining)	Used to synchronize (combine) parallel flows and to create parallel flows

Table 5.12 – Data Object types in BPMN model

Designation	Name	Description
	Singular Data Object	Data used in a process that is provided by a single object
	Data Object that is a collection	Data used in a process that is provided by a collection of objects
	Data Store	Provides a mechanism for activities to retrieve or update stored information that will persist beyond the scope of the process
	Data Input	Represent data at entrance of a process
	Data Output	Represent data at exit of a process

Connecting Objects are used to connect diagram's elements. Connectors types are shown in table 5.13.

Table 5.13 – Connecting Objects types in BPMN model

Designation	Name	Description
⟶	Sequence Flow	Defines a process direction
····›	Message Flow	Expresses a flow of messages between elements
······	Association	Denotes association between elements

Swimlanes group a process according to executors categories and process participants. Artifacts include Text Annotations and Groups that group activities not affecting a process.

Such SACE-tools as Visual Paradigm, Sybase Power Designer, ER/Studio Business Architect, and BPMN 2.0 Modeler for Microsoft Office Visio provide business-processes modeling in BPMN notation.

5.7. The review of Visual Paradigm for UML

Visual Paradigm for UML (VP-UML) is a powerful, cross-platform and yet the most easy-to-use visual UML modeling and CASE tool. VP-UML provides software developers the cutting edge development platform to build quality applications faster, better and cheaper. It facilitates excellent interoperability with other CASE tools and most of the leading IDEs which excels your entire Model-Code-Deploy development process in this one-stop-shopping solution.

Figure 5.4 shows how the environment of Visual Paradigm for UML Enterprise Edition looks like. The elements of the VP-UML environment and their description are given in table 5.14.

Table 5.14 – VP-UML environment elements

№	Element	Description
1	Menu bar	The menu bar at the top of the window allows to select and perform various operations in Visual Paradigm for UML
2	Toolbar	Toolbar is the extension of menu. All buttons are presented as groups of icons that handily placed for users
3	Diagram Navigator	A place where diagrams are listed, and where you can create and access diagrams base on their types
4	Properties Pane	The properties of chosen model/ shapes will be shown on properties pane upon selection
5	Message Pane	Possible information or warnings will be shown here
6	Diagram Pane	The diagram will be displayed in diagram pane

Visual Paradigm for UML is designed for a wide range of users, including software engineers, system analysts, business analysts, system architects alike, who are interested in building software systems reliably through the use of the object-oriented approach.

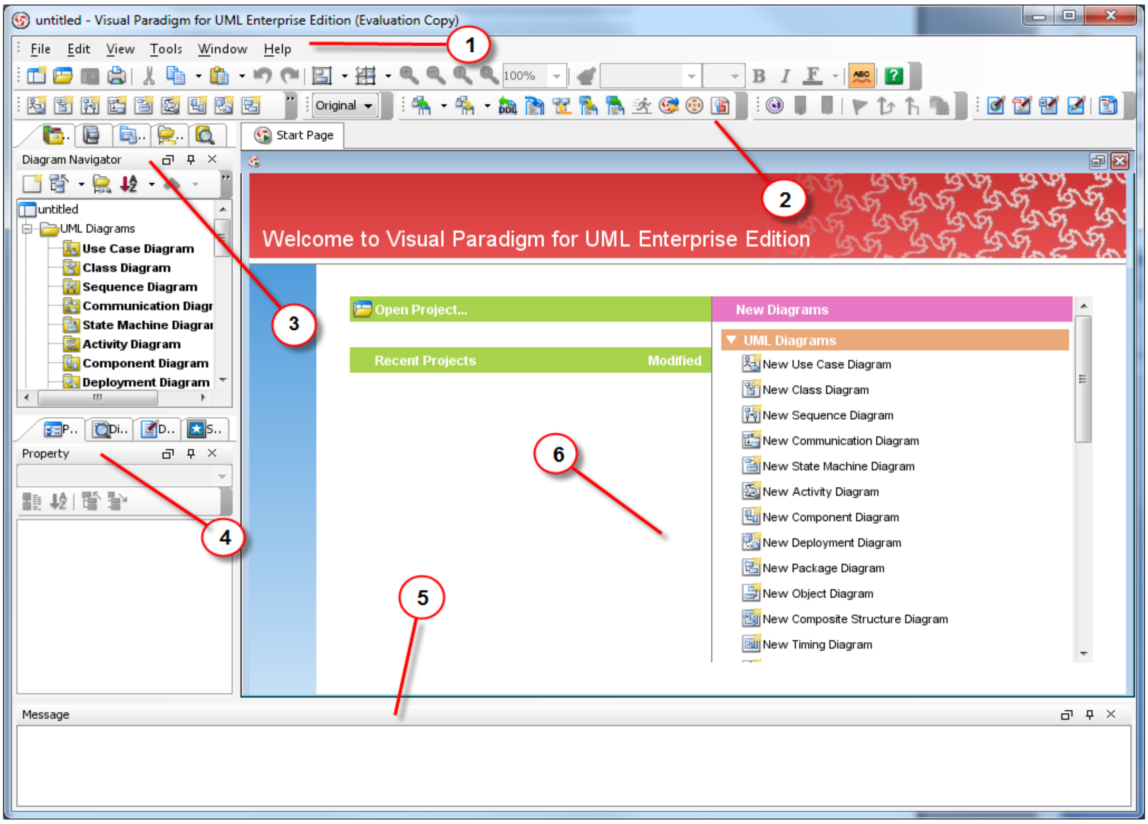


Figure 5.4 – Visual Paradigm for UML Enterprise Edition

To create a diagram say, a use case diagram it is necessary:

1) Right click Use Case Diagram in Diagram Navigator and select New Use Case Diagram (fig. 5.5).

2) Enter the name for the new diagram after creating it.

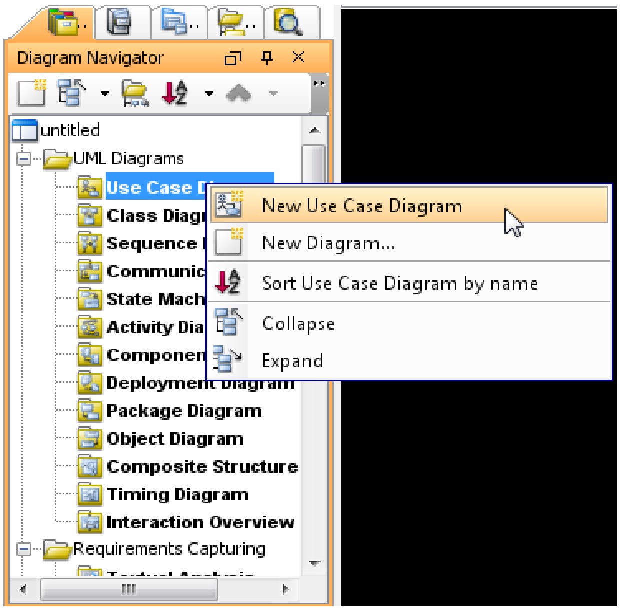


Figure 5.5 – Creating a new diagram

To create a shape, select the type of shape from the diagram toolbar, click on diagram and start dragging it to create with a preferred size. Taking creating use case as an example:

1) Select Use Case from the diagram toolbar, click on the diagram and then start dragging. Release the mouse to confirm creation (fig. 5.6).

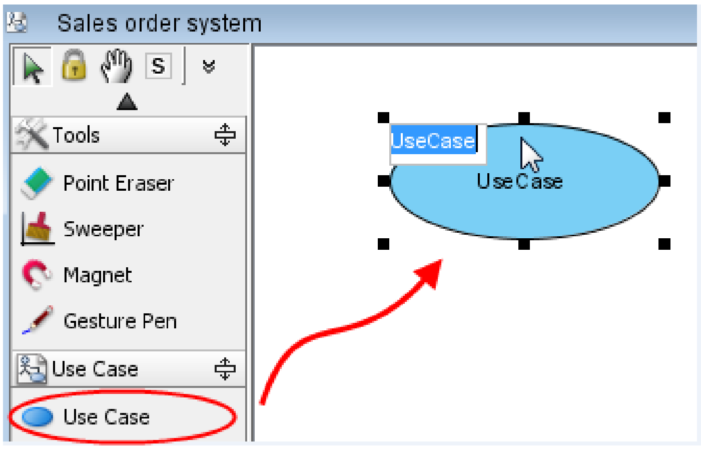


Figure 5.6 – Creating a use-case

2) Enter the name for the use case and press enter to confirm the name (fig. 5.7).

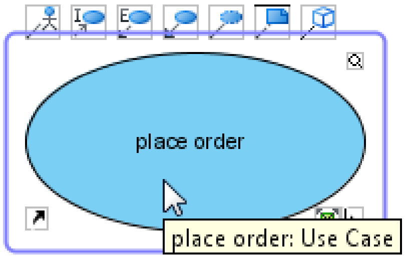


Figure 5.7 – Entering use-case name

Two shapes can be connected by making use of the resource icons surrounding a shape. Let say if we want to associate an actor with a use case, move the mouse pointer towards the actor, press on its Association resource icon and drag it to the use case, finally release the mouse button (fig. 5.8).

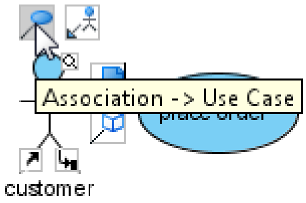
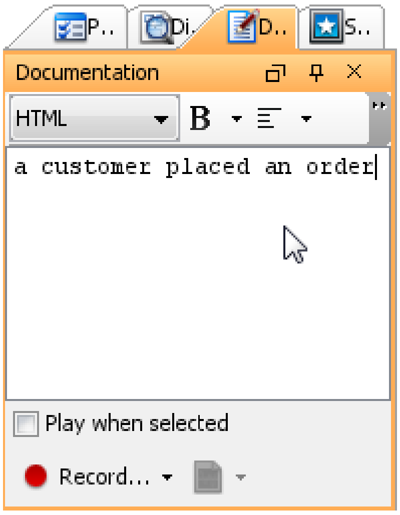


Figure 5.8 – Association between an actor and a use-case

You can type in the textual description to your shape by opening the Documentation Pane at the bottom left of screen and typing in the space provided (fig. 5.9).



a customer placed an order

Figure 5.9 – Textual description

You can format a shape with your preference by right clicking on it, selecting Styles and Formatting and then Formats… from the popup menu. Taking changing the use case's background color as an example:

1) Right clicking use case and select Styles and Formatting and then Formats… from the popup menu (fig. 5.10).

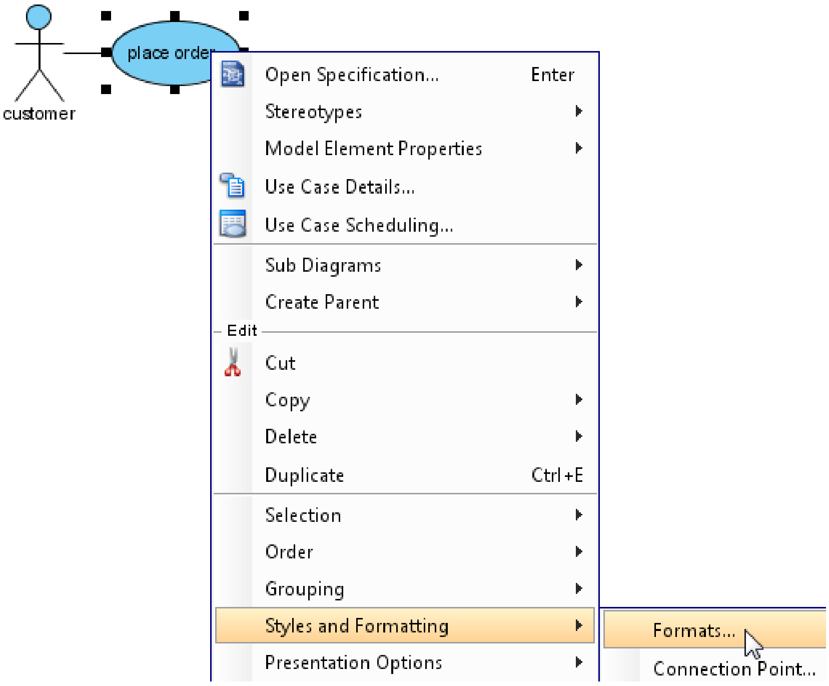
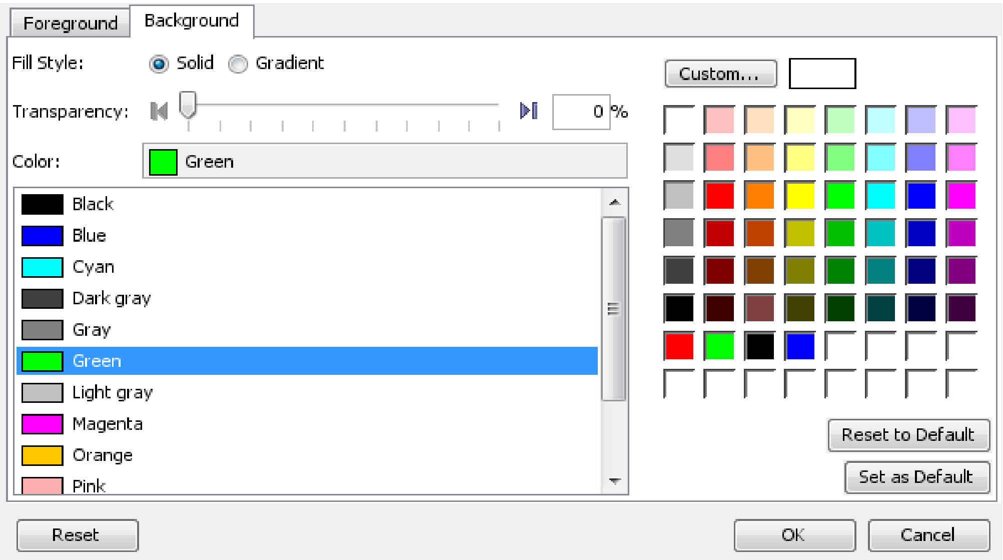


Рисунок 5.10 – Форматування прецеденту

2) In the Formats dialog box, select a color such as green in the Background tab and click OK to confirm the change, the color of use case will be changed into green (fig. 5.11).



Fill Style: ● Solid ○ Gradient

Transparency: 0 %

Color: Green

Figure 5.11 – Changing use-case colour

Layer is a feature that lets you divide shapes on a diagram into logical groups, and perform various actions including changing their visibility, and disabling the editing on them.

You can create a new layer by selecting View > Layers… from the main menu. In the Diagram Layers dialog box, click on the Create new layer button. Name the new layer as Annotation Layer and then click Close to confirm the change (fig. 5.12).

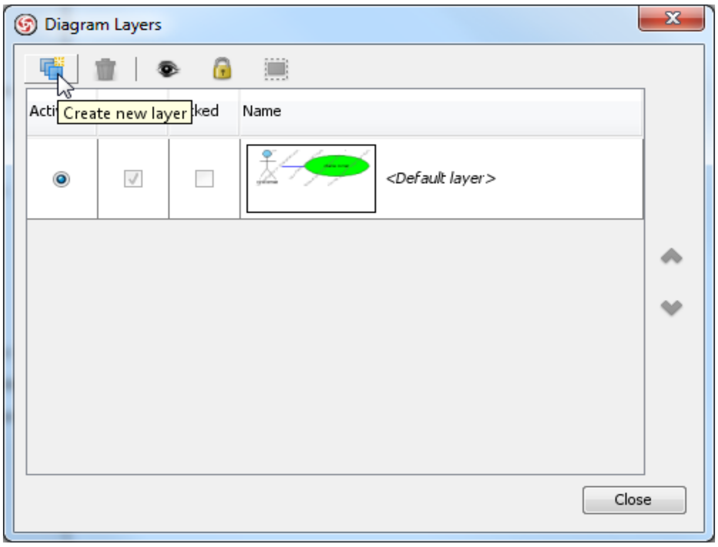


Figure 5.12 – Creating a new level

A shape can be moved to another layer by right clicking on it and selecting the layer to be moved to. For example, a callout can be moved from the current layer to a new layer specialized for annotation by right clicking on the callout shape.

To create a use-case diagram it is necessary to define actors and use-cases. So first of all you should choose actor grid (fig. 5.13).

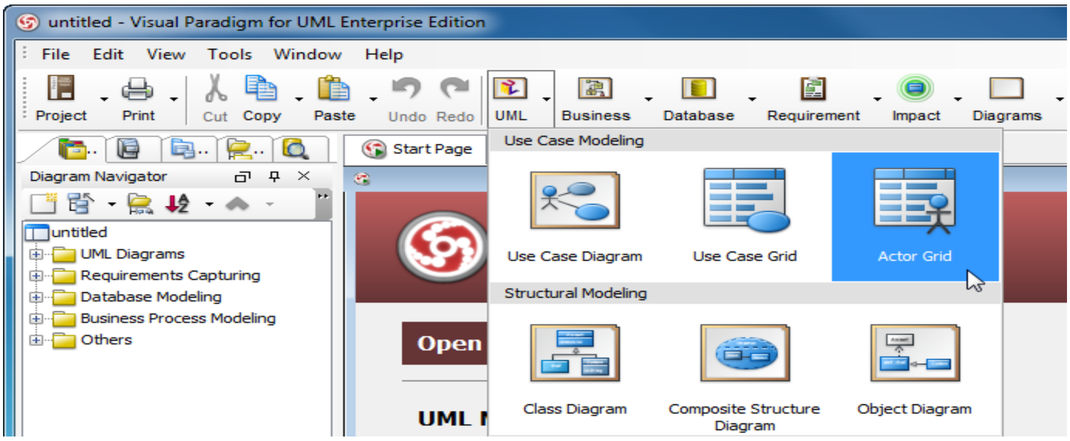
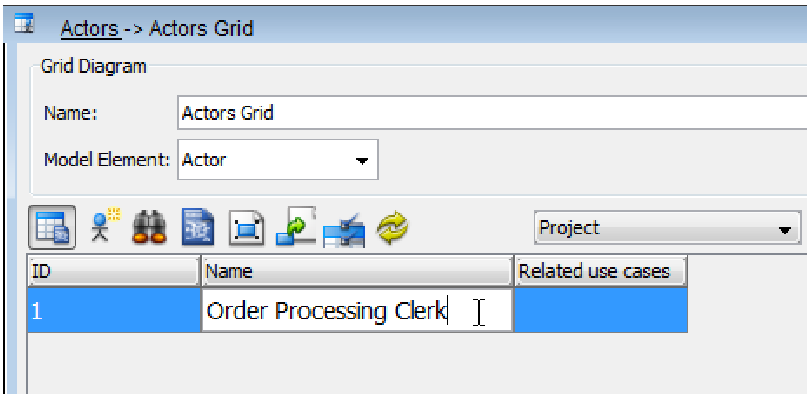

untitled - Visual Paradigm for UML Enterprise Edition
File Edit View Tools Window Help
Project Print Cut Copy Paste Undo Redo UML Business Database Requirement Impact Diagrams
Diagram Navigator
untitled
UML Diagrams
Requirements Capturing
Database Modeling
Business Process Modeling
Others
Start Page
Open
UML
Use Case Modeling
Use Case Diagram
Use Case Grid
Actor Grid
Structural Modeling
Class Diagram
Composite Structure Diagram
Object Diagram

Figure 5.13 – Choosing Actor Grid

Then the following actions must be performed:

1) Click the Create Actor button at the top left hand corner to create a new actor.

2) Name the actor (fig. 5.14).


Actors -> Actors Grid

Grid Diagram

Name: Actors Grid

Model Element: Actor

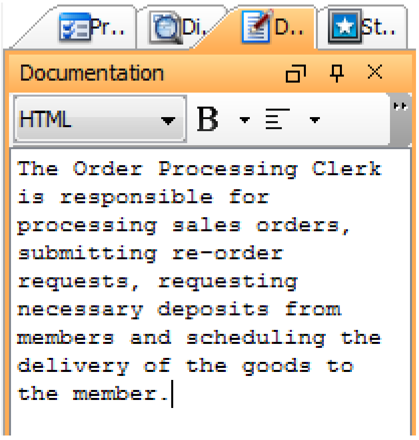
Project

ID	Name	Related use cases
1	Order Processing Clerk	

Figure 5.14 – Entering actor's name

3) Specify the documentation in the Documentation pane (fig. 5.15).

After creating actors, we now switch to create use cases. Select UML > Use Case Grid from toolbar (fig. 5.16).



The Order Processing Clerk is responsible for processing sales orders, submitting re-order requests, requesting necessary deposits from members and scheduling the delivery of the goods to the member.

Figure 5.15 – Documenting actor

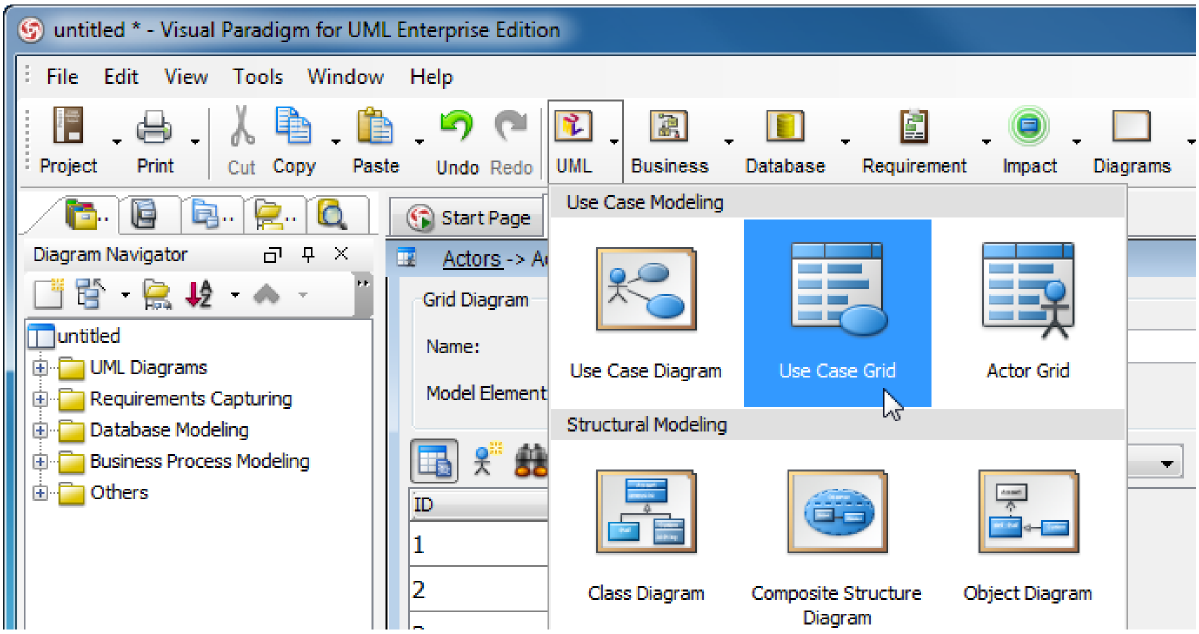


Figure 5.16 – Choosing Use-Case Grid

Then the following actions should be performed:

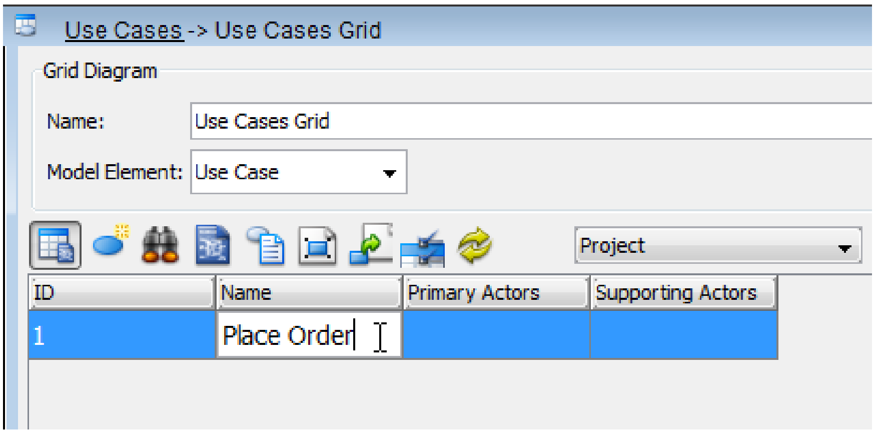
1) Similar to the Actor Grid, you can create use case with the Create Use Case button at the top left hand corner of the grid.

2) Name the use case (fig. 5.17).

To draw use case diagram:

1) After we have defined the details of the use case, we can now put it into diagram. Select the Place Order use case in the Use Case Grid, and press the Visualize button (fig. 5.18).

2) Let's visualize the Place Order use case in a new use case diagram. Select Create new diagram and leave the name as default in the Visualize Model Element dialog. Click the Create button to proceed (fig. 5.19).


Use Cases -> Use Cases Grid

Grid Diagram

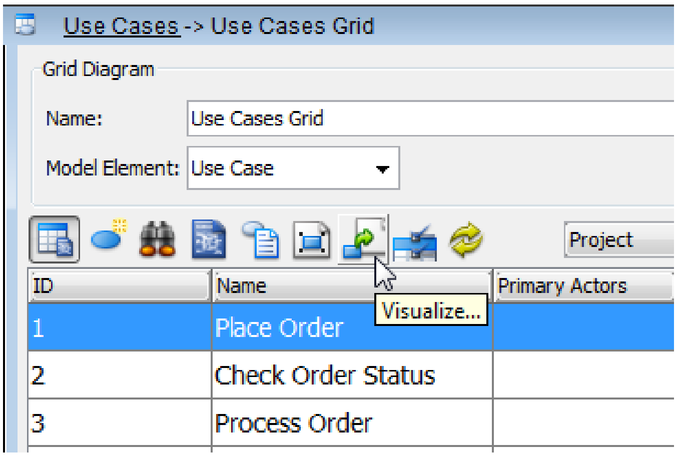
Name: Use Cases Grid

Model Element: Use Case

Project

ID	Name	Primary Actors	Supporting Actors
1	Place Order		

Figure 5.17 – Entering use-case name


Use Cases -> Use Cases Grid

Grid Diagram

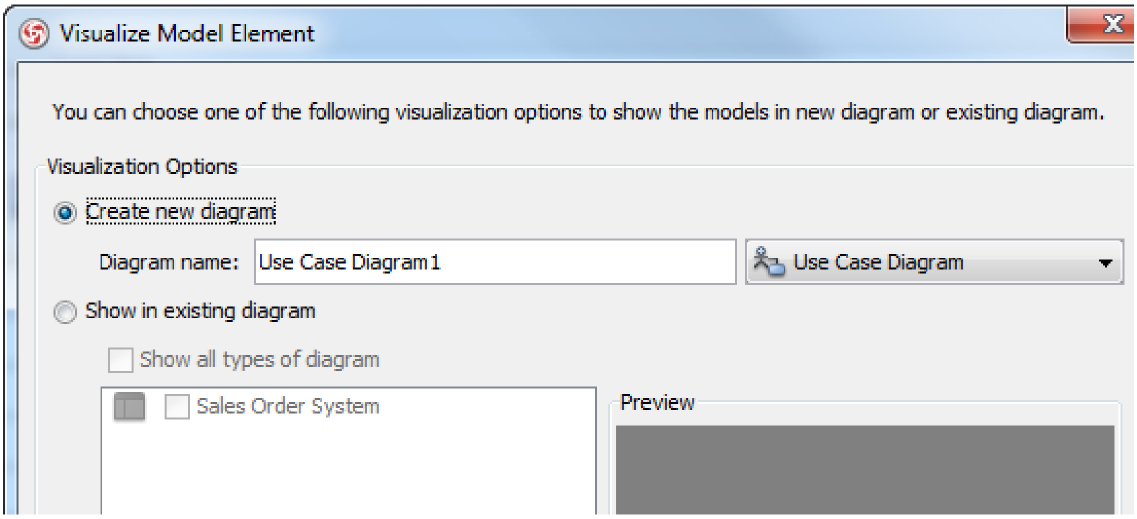
Name: Use Cases Grid

Model Element: Use Case

Project

ID	Name	Primary Actors
1	Place Order	
2	Check Order Status	
3	Process Order	

Figure 5.18 – Use-case visualization


Visualize Model Element

You can choose one of the following visualization options to show the models in new diagram or existing diagram.

Visualization Options

● Create new diagram

Diagram name: Use Case Diagram1 | Use Case Diagram

○ Show in existing diagram

□ Show all types of diagram

□ Sales Order System

Preview

Figure 5.19 – Creating a new use-case diagram

3) Open Model Explorer, and drag the Customer Service Assistant actor to the newly created use case diagram (fig. 5.20).

4) Since the actor is not appearing on any diagram yet, you will be prompted to specify it as Master View in the project (fig. 5.21).

5) Connect the actor and use case by selecting the Association resource icon from actor, and drop it on the use case (fig. 5.22).

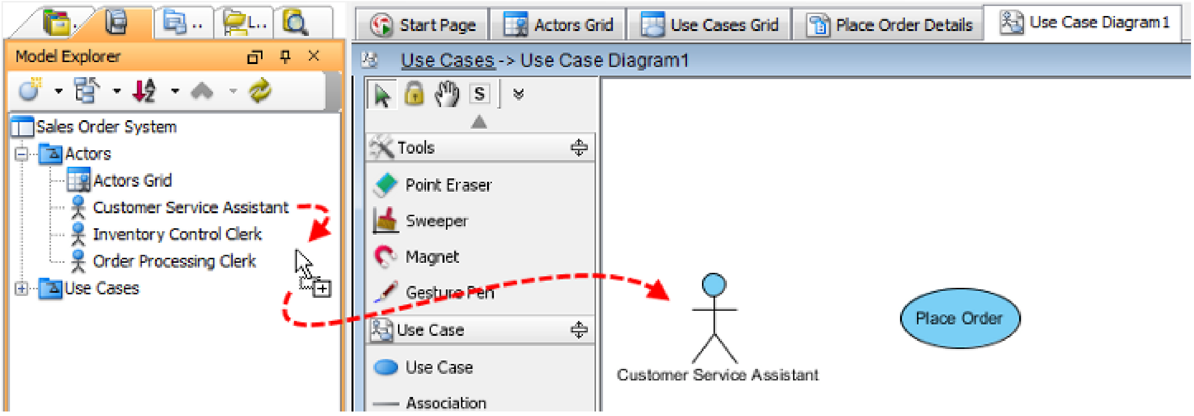
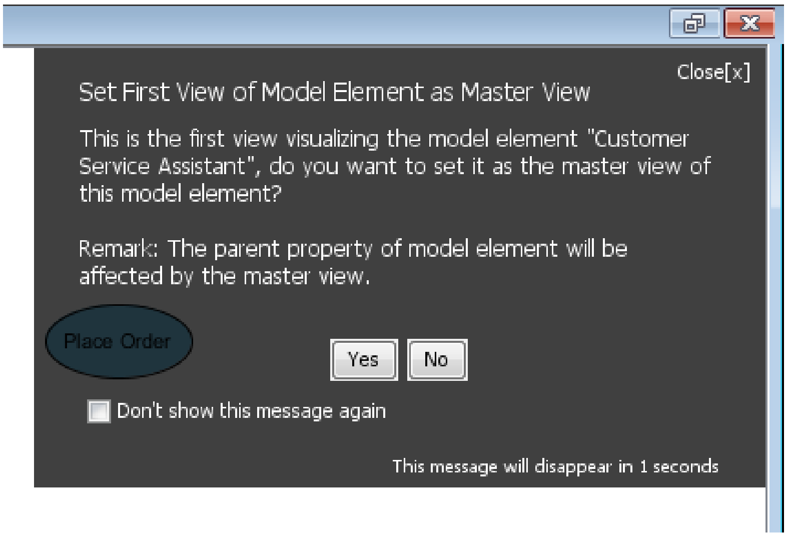


Figure 5.20 – Actor's visualization



Close[x]

Set First View of Model Element as Master View

This is the first view visualizing the model element "Customer Service Assistant", do you want to set it as the master view of this model element?

Remark: The parent property of model element will be affected by the master view.

Yes No

□ Don't show this message again

This message will disappear in 1 seconds

Figure 5.21 – Actor's specification

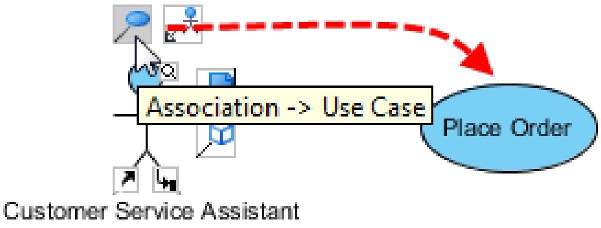
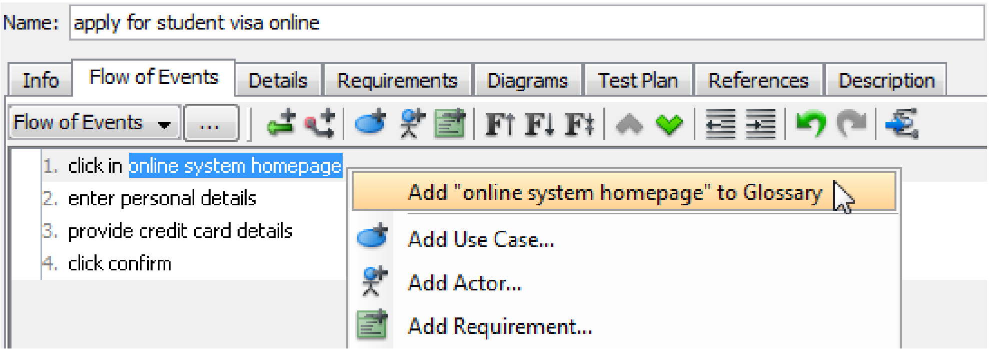


Figure 5.22 – Relation between actor and use-case

Glossary is a place where domain-specific vocabularies are stored and managed. Instead of adding terms (vocabularies) from your imagination, you can develop a glossary by identifying terms from flow of events.

1) Suppose online system homepage in a step of flow of events is a key phrase. Select it in flow of events, right click and select Add ‘online system homepage’ to Glossary from the popup menu to extract this as a term (fig. 5.23).

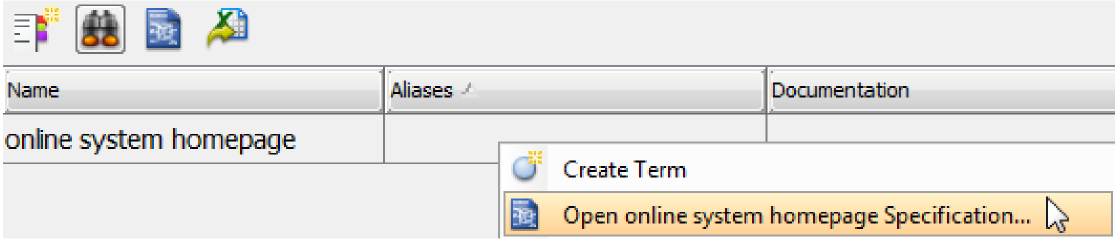


Name: apply for student visa online

1. click in online system homepage
2. enter personal details
3. provide credit card details
4. click confirm

Figure 5.23 – Adding a term to glossary

2) This opens the glossary with term online system homepage added. Right click on the term and select Open online system homepage Specification… (fig. 5.24).



Name	Aliases	Documentation
online system homepage		

Figure 5.24 – Open term’s specification

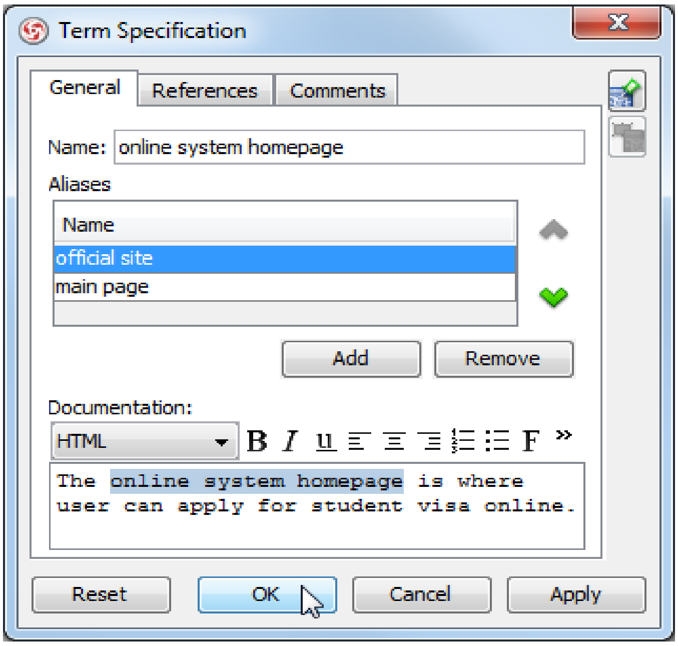
3) Try to specify its alias. In the Term Specification dialog box, click Add and enter official site as the first alias. Repeat to add alias main page. Enter description of the term in Documentation. Finally, click OK to finish editing (fig. 5.25).

A sequence diagram is used primarily to show the interactions between objects that are represented as lifelines in a sequential order. More importantly, lifelines show all their points of interaction with other objects in events. A sequence diagram can be created by right clicking Sequence Diagram on Diagram Navigator and then selecting New Sequence Diagram from popup menu.

Taking the use case of apply for student visa online as an example:

1) Select Actor from the diagram toolbar to create an actor. Name it as Student.

2) Press on the resource icon Message -> LifeLine and drag it into the location you prefer to create a lifeline with interaction to actor Student. Name the lineline online system, and the message in between enter personal details (fig. 5.26).



Name: online system homepage

Name
official site
main page

The online system homepage is where user can apply for student visa online.

Figure 5.25 – Editing a term

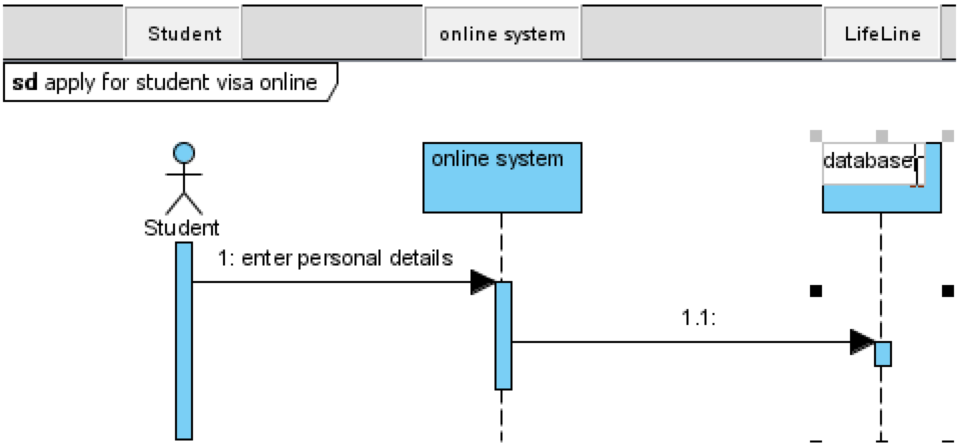


Figure 5.26 – Creation of a sequence diagram

An activity diagram is essentially a flowchart, showing flow of control from one activity to another. Unlike a traditional flowchart, it can model the dynamic aspects of a system because it involves modeling the sequential steps in a computational process. A new activity diagram can be created by right clicking

Activity Diagram on Diagram Navigator and then selecting New Activity Diagram.

The following actions must be done:

1) Select Initial Node on diagram toolbar and drag it to the diagram pane (fig. 5.27).

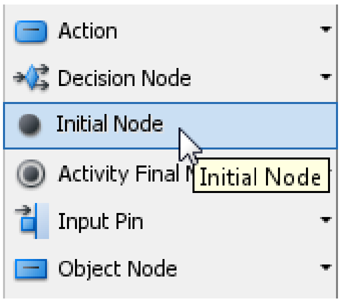


Figure 5.27 – Creating Initial Node

2) Create an action through the resource icon Control Flow -> Action of initial node. Press on the icon and drag to the target position. Name the action (fig. 5.28).

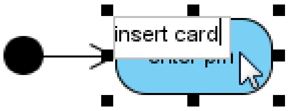


Figure 5.28 – Creating a control flow and an action

3) Terminate the activity by creating an activity final node through the resource icon Control Flow -> Activity Final Node of the final action (fig. 5.29).

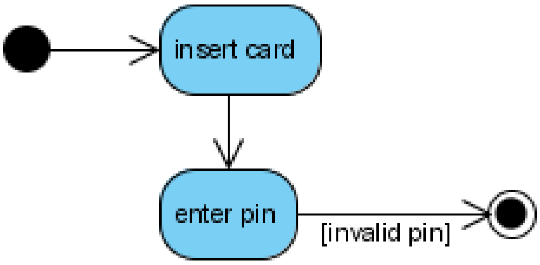


Figure 5.29 – Adding a Final Node

4) You can use swimlane to group actions based on the participants involved. Select Horizontal Swimlane from the diagram and drag it on the diagram pane to create it (fig. 5.30).

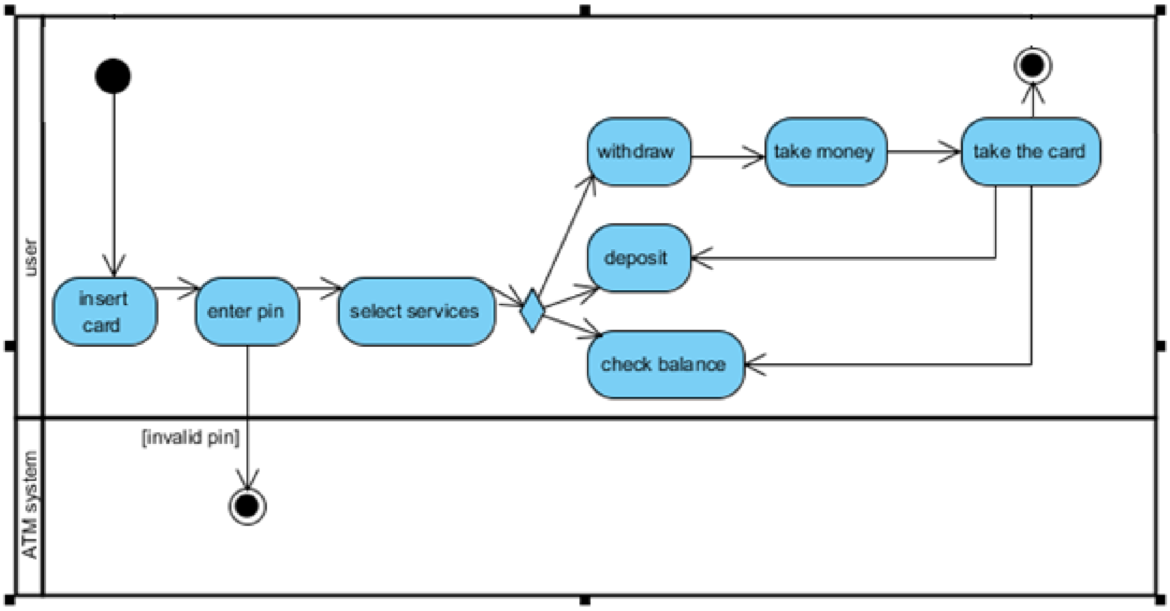


Figure 5.30 – Horizontal Swimlane usage

A class diagram shows the blueprints of objects required by a system and the relationships between them. A new class diagram can be created by right clicking on Class Diagram on Diagram Navigator and then selecting New Class Diagram from the popup menu.

Taking bank account as an example:

1) Select Class on the diagram toolbar and drag it to the diagram pane (fig. 5.31)

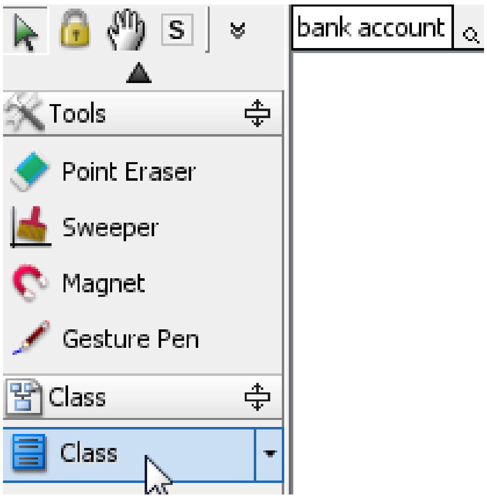


Figure 5.31 – Creating a class

2) Name the class as bank account. To create an attribute, right click the class and select Add > Attribute from the popup menu (fig. 5.32). You can create as many as attributes as you need by pressing enter after adding a new one. In the class of bank account, create attributes name and balance. Similarly, operation can be created by right clicking the class and select Add > Operation from the popup menu. Create operations debit and credit.

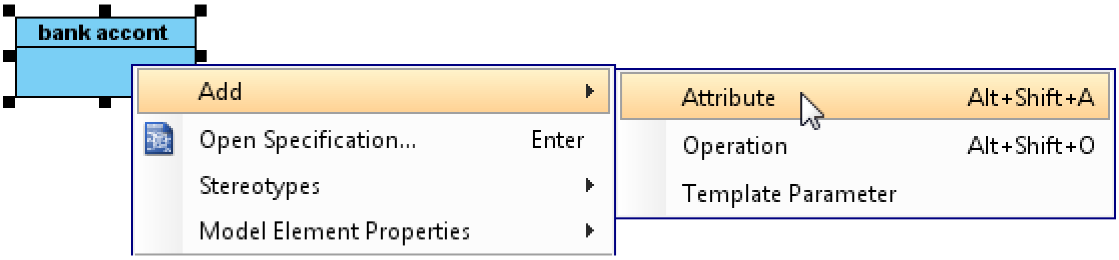


Figure 5.32 – Creating class attributes

3) Generalization is needed when you want to show the subclass out of the super class. It can be created by pointing the super class, selecting the resource Generalization -> Class and dragging it to the place you want the subclass to appear (fig. 5.33).

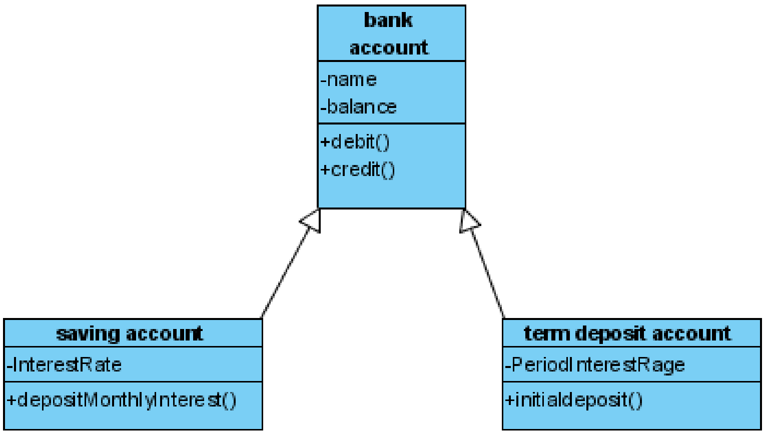


Figure 5.33 – Creating generalization relationship

Let's consider an example of several diagrams elaboration in the Visual Paradigm for UML environment.

It is necessary to create use-case, class, activity and component diagrams for documenting the process of software development for the orders management system. The consumer can make two types of orders that are processed by the system: normal orders and special ones.

The part of orders management system functionality defined by the above description can be presented on the use-case diagram (fig. 5.34).

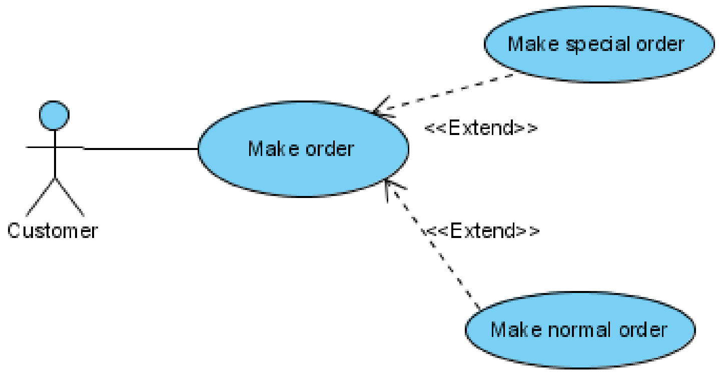


Figure 5.34 – Use-case diagram

Class diagram will look like it is shown on the figure 5.35.

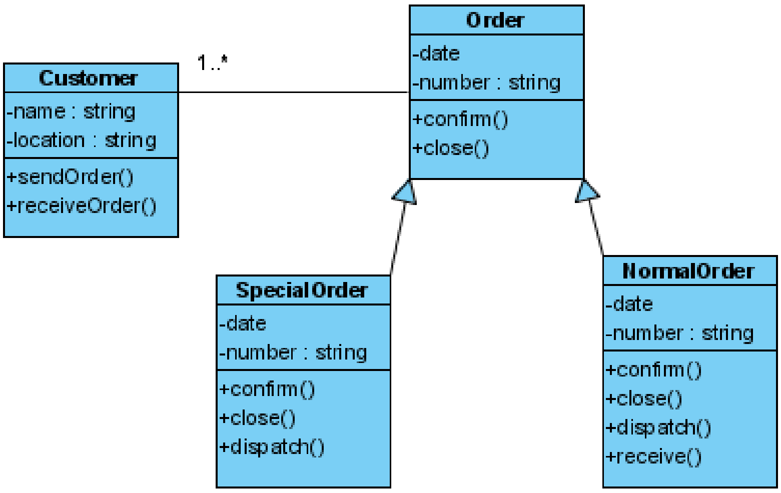


Figure 5.35 – Class diagram

Activity diagram is shown on the figure 5.36. Component diagram is represented on the figure 5.37.

To save your work, select either File > Save Project or File Save Project as…. When you perform saving the first time, you can select to save the project in workspace, or to another directory you preferred (fig. 5.38).

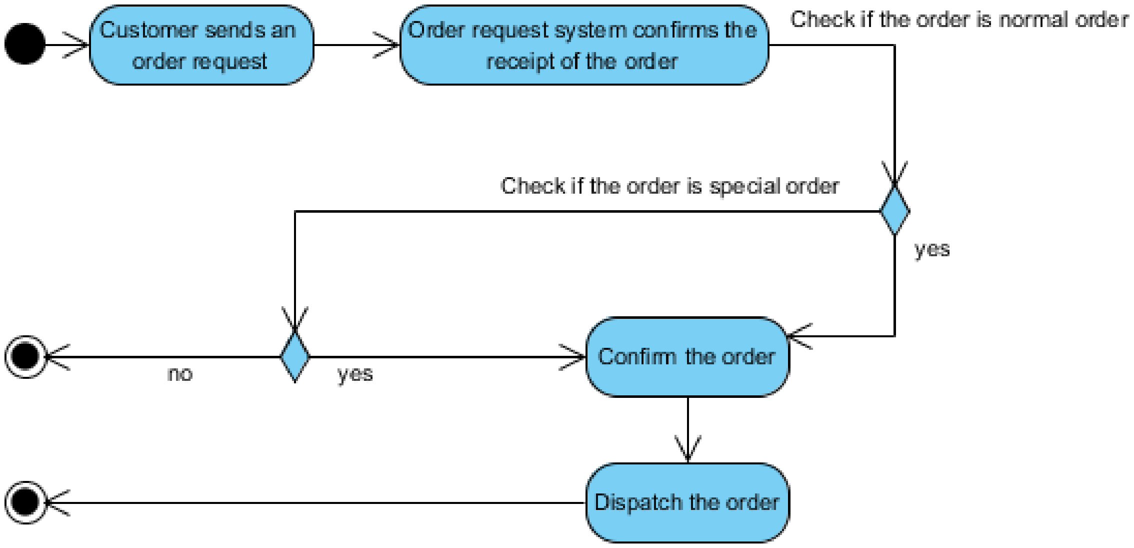

Customer sends an order request
Order request system confirms the receipt of the order
Check if the order is normal order
Check if the order is special order
yes
no
yes
Confirm the order
Dispatch the order

Figure 5.36 – Activity diagram

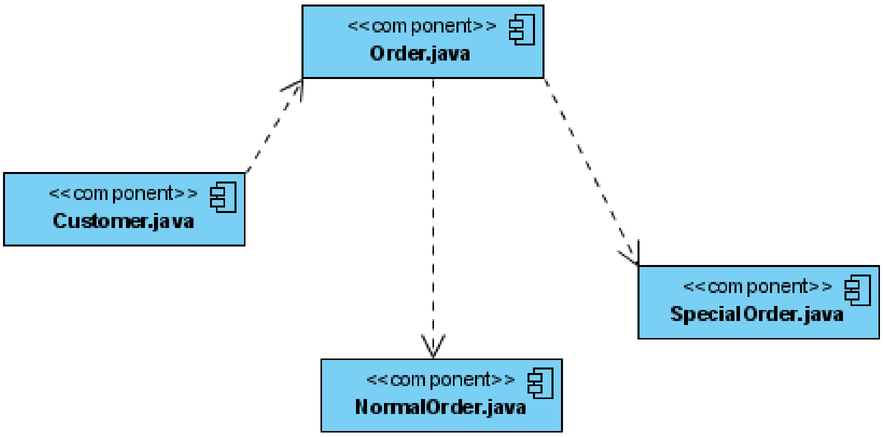
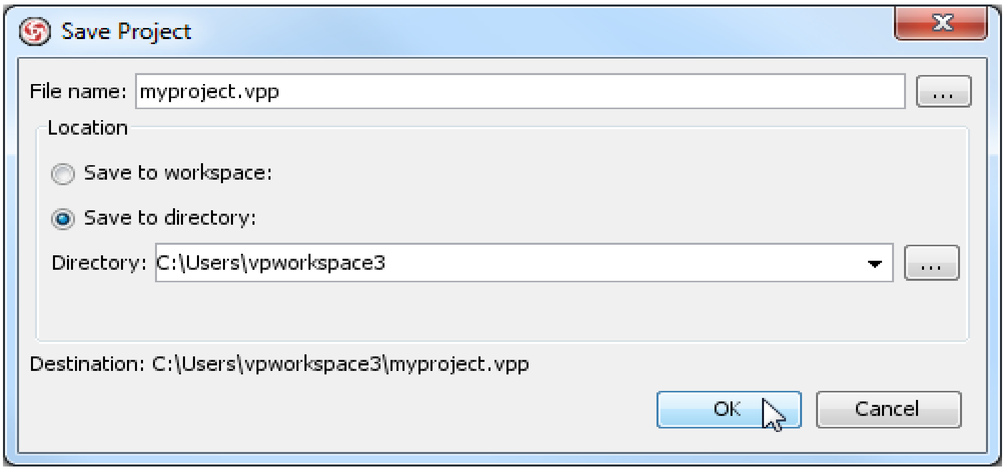


Figure 5.37 – Component diagram



File name: myproject.vpp

Location

○ Save to workspace:

● Save to directory:

Directory: C:\Users\vpworkspace3

Destination: C:\Users\vpworkspace3\myproject.vpp

Figure 5.38 – Saving the project

5.8. Business-process modeling case-study

Let’s consider a process of some discipline study in higher education establishment. A teaching plan of any discipline supposes that studied material is divided into several modules. Each module is finished by some form of control (for example, a control work). A discipline study is supported by two basic forms of material learning: auditorium lessons (lectures, practical classes, laboratory works) and independent work. In addition, a fulfillment of individual home assignments (IHA) may be possible to fix theoretical knowledge and practical skills. As a rule, the final semester control must be done in the end of discipline study. It takes the form of exam or credit.

Let’s suppose that discipline study includes visiting lectures by students, execution of laboratory works and independent work on IHA. All material is divided into two modules. A written module control work is used as a module control form. A student passes an exam in the end of semester.

Let’s build s context diagram in IDEF0 (fig. 5.39).

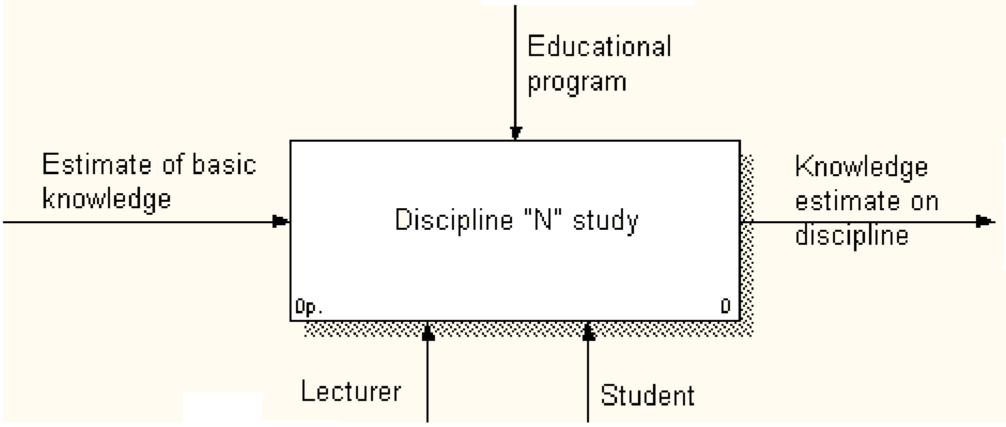


Figure 5.39 – Context diagram of discipline study

Learning of each discipline supposes that a student has some definite knowledge needed for mastering a new material. Discipline study is regulated by educational program, which defines the number of hours, topics, forms of control, etc. As a result o discipline study a student obtains knowledge that is estimated by a lecturer.

During a discipline study a student learns theoretical material, makes laboratory works and work independently (fig. 5.40). The current control means recording of all works passing and obtaining an admission for exam. A final control means exam passing and assessment of knowledge.

An independent work (fig. 5.41) consists of obtaining a task, execution of an IHA and making consultations by a lecturer at particular time. An IHA is given according a student's number in a group. To fulfill an IHA a student uses guidelines.

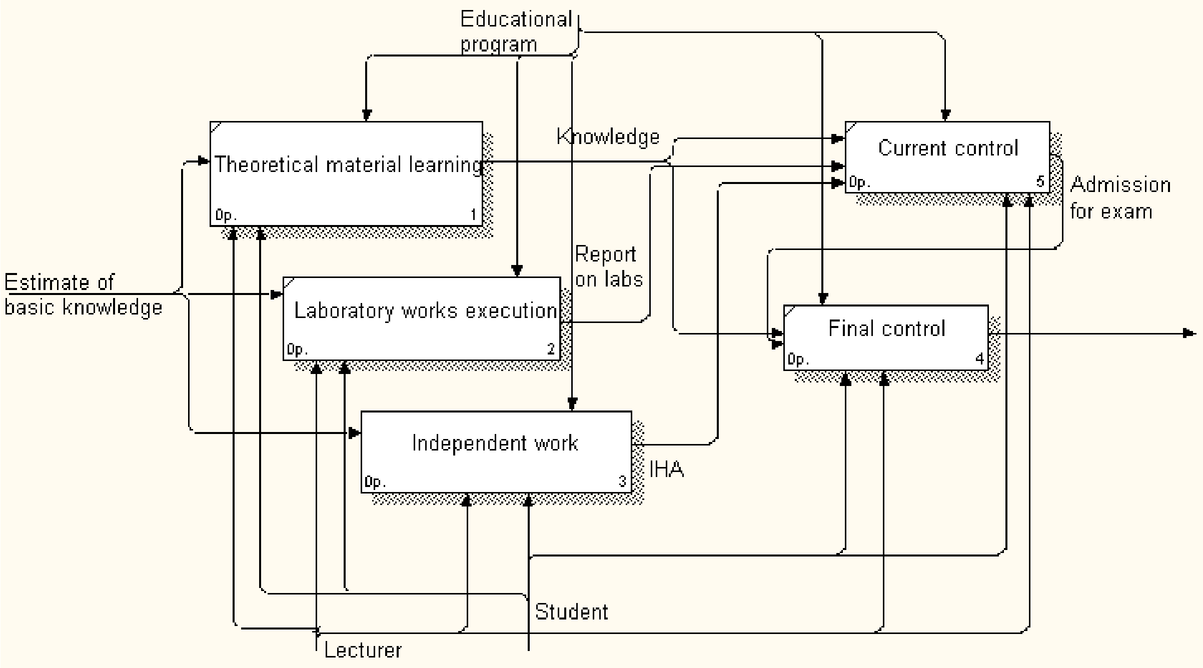


Figure 5.40 – Decomposition of discipline study process in IDEF0

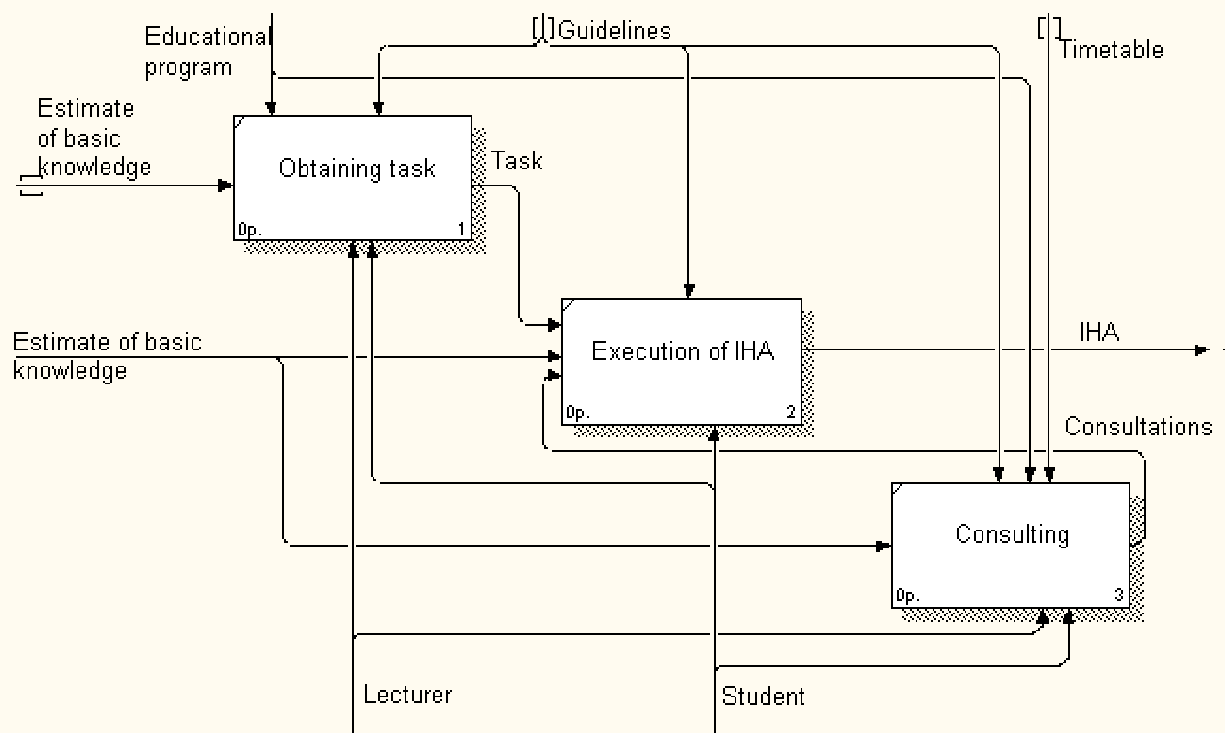


Figure 5.41 – Decomposition of independent work in IDEF0

The current control is made up of module control works, passing of all laboratory works and IHA. A lecturer admits a student to an exam on this basis (fig. 5.42). A lecturer admits a student to an exam or gives him a list of debts on the basis of information about passed modules, laboratory works and IHA stored in a register (fig. 5.43).

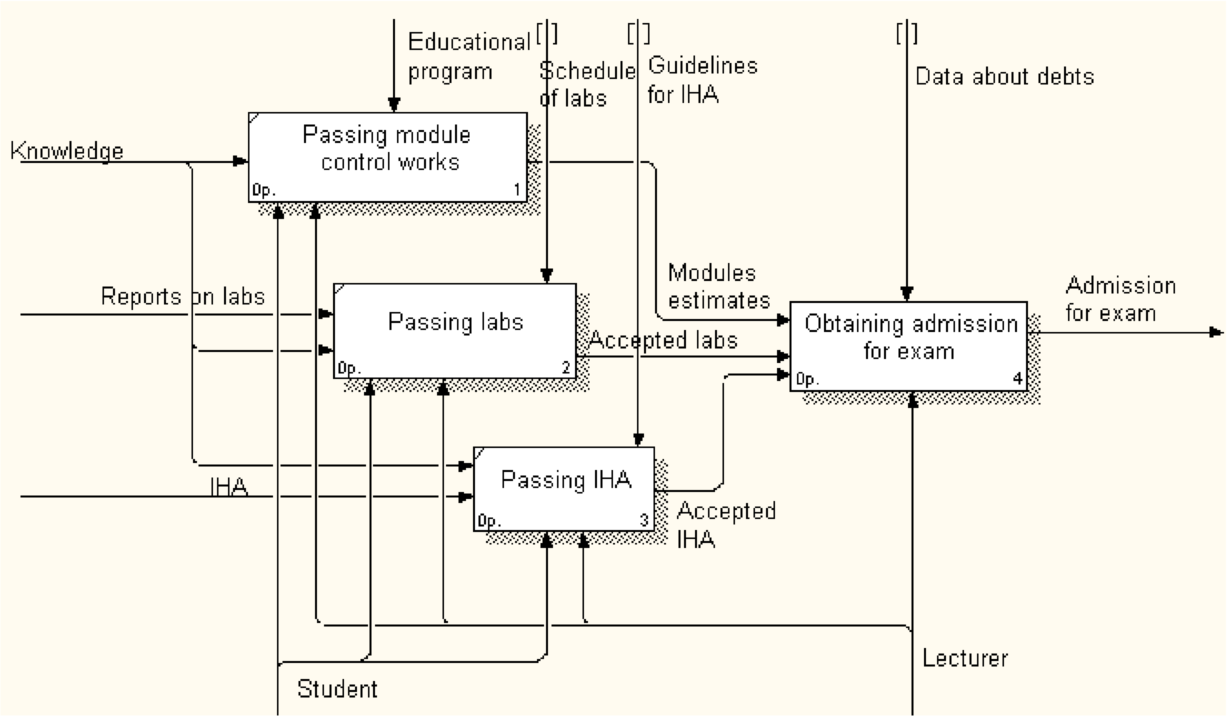


Figure 5.42 – Decomposition of current control process in IDEF0

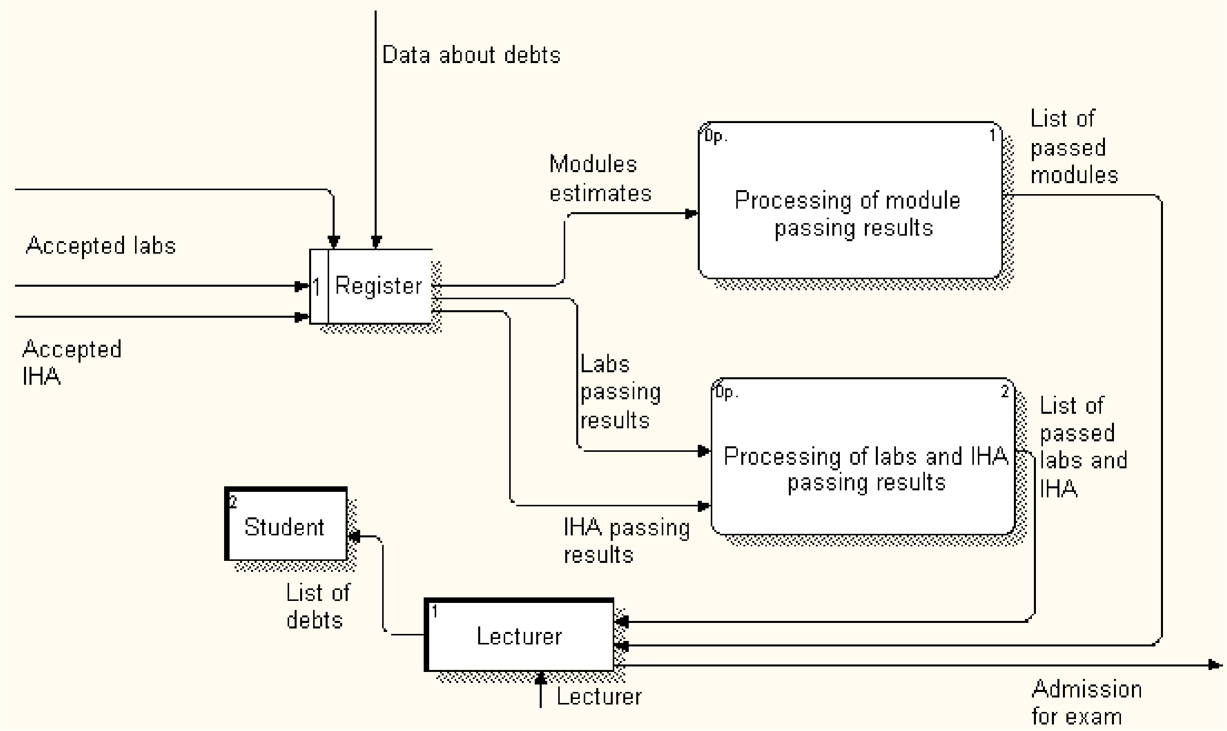


Figure 5.43 – Process of obtaining admission for exam in DFD

During the final control the estimate of student's knowledge in the discipline is defined (fig. 5.44). It may coincide with a preliminary estimate obtained from modules' results. If a student wishes to increase an estimate, he/she can pass an exam and get a final estimate.

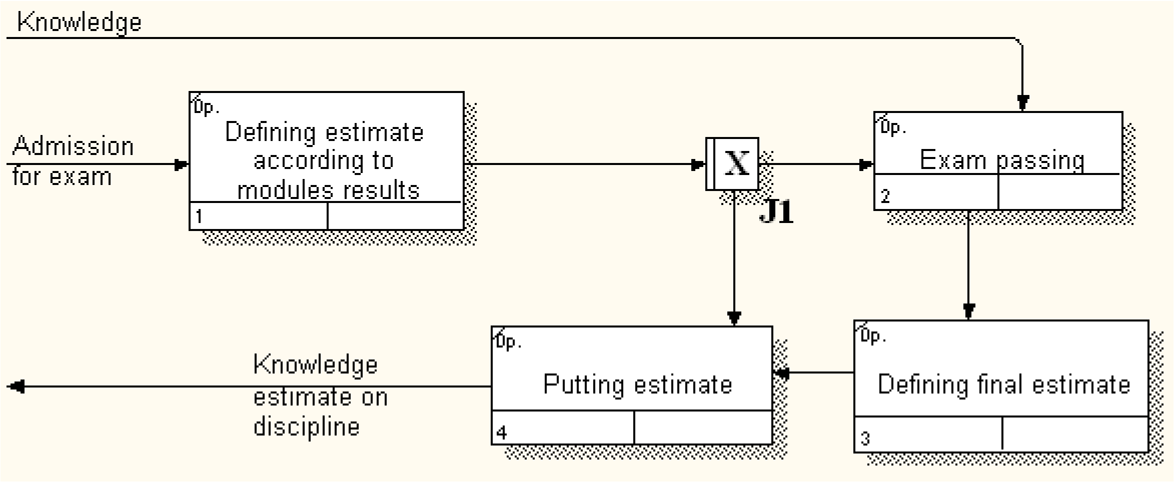


Figure 5.44 – Decomposition of final control in IDEF3

The described process can be modeled using UML standard. In this case participants and main business-functions are represented on the use-case diagrams (fig. 5.45).

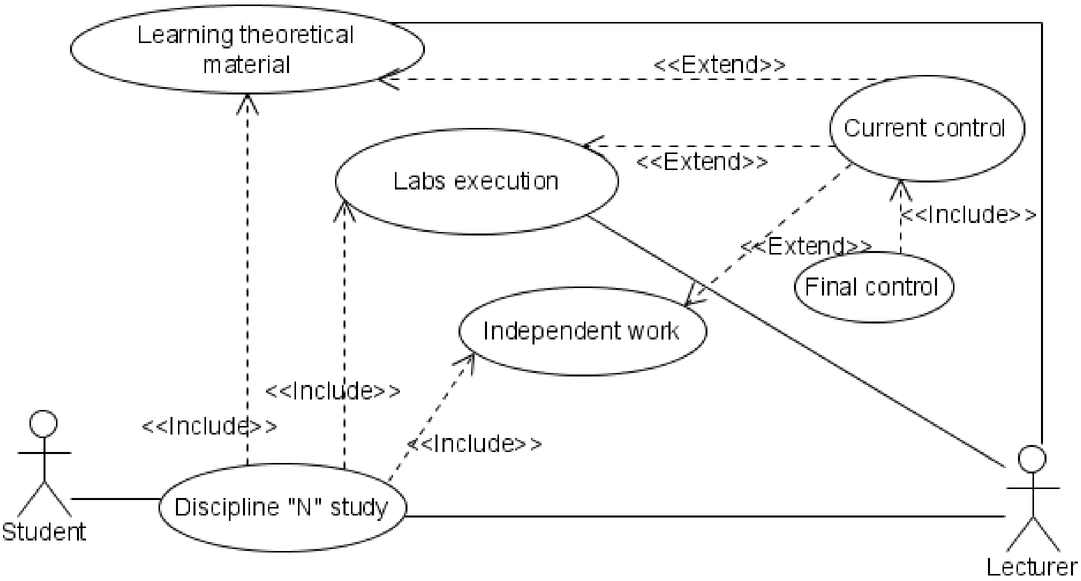


Figure 5.45 – Use-case diagram of discipline study process (UML)

A process of independent work is represented on the activity diagram (fig. 5.46). A student obtains a task, make an IHA, consults with a lecturer. Finished IHA is passed. A lecturer checks it. If IHA is correctly fulfilled, a lecturer makes a correspondent record in a register, otherwise a student corrects it and passes again. A process of admission for exam obtaining is shown on the figure 5.47 in UML notation.

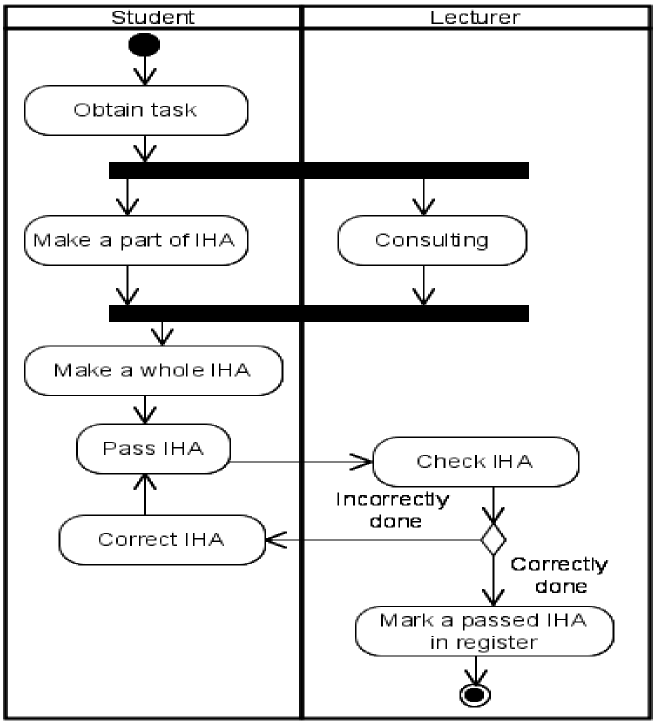


Figure 5.46 – IHA passing process (UML)

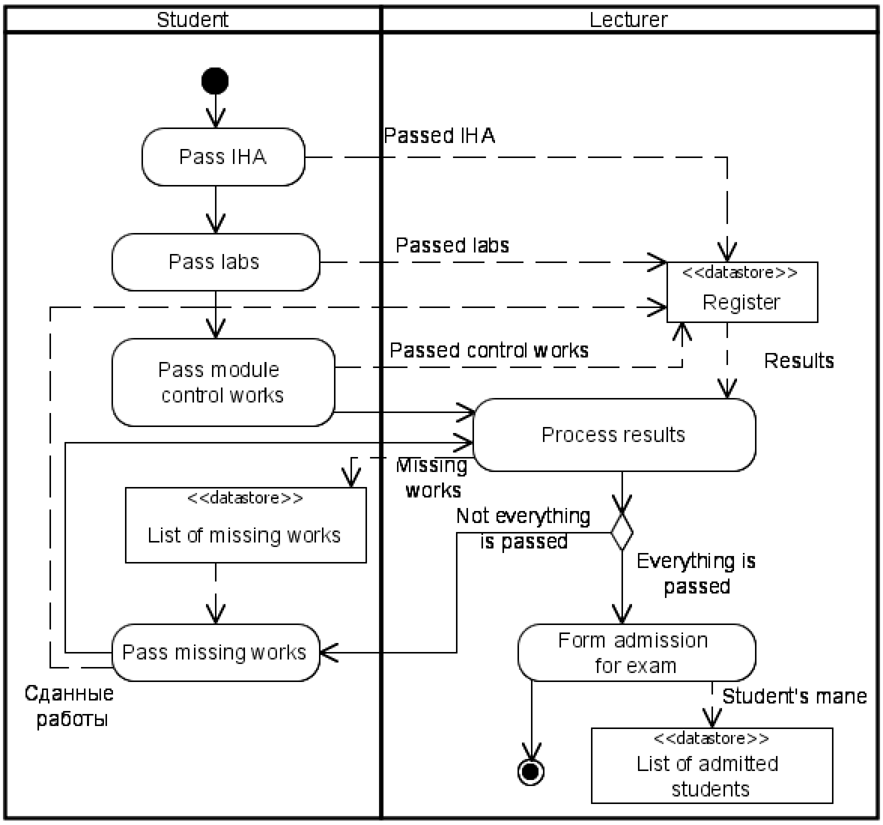


Figure 5.47 – Admission obtaining process (UML)

Let's consider a business-process of discipline study in higher education establishment in BPMN notation (fig. 5.48).

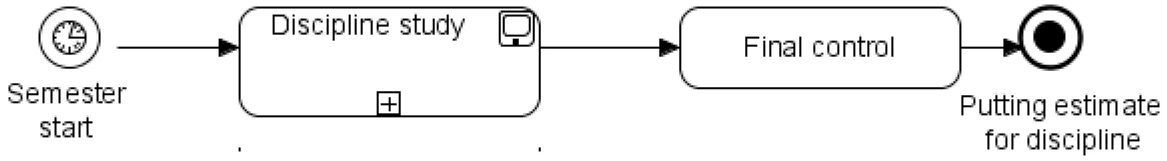


Figure 5.48 – Discipline study (BPMN)

A discipline study subprocess can be decomposed (fig. 5.49). it includes learning theoretical material, making laboratory works and IHA, modules passing. The results of study activities are processed and a student obtains an admission for an exam.

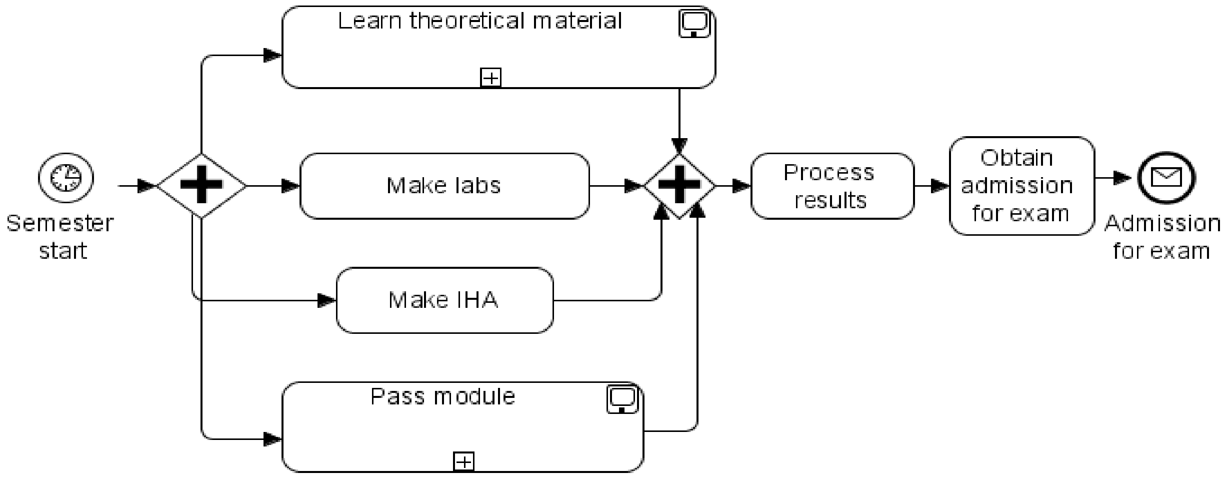


Figure 5.49 – Decomposition of discipline study process (BPMN)

Learning theoretical material supposes that students must visit lectures, that are interrupted by a module week, and must learn additional literature on the discipline by themselves (fig. 5.50).

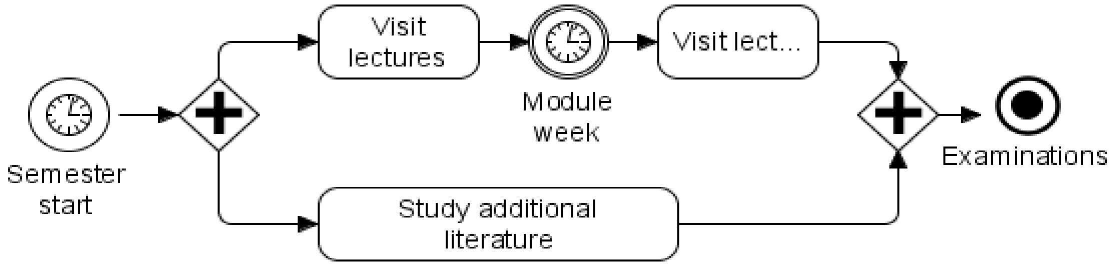


Figure 5.50 – Learning theoretical material (BPMN)

Module passing (fig. 5.51) consists of giving a task for module control work by a lecturer, its execution by a student, and checking. If a lecturer does not accept a control work, a student remakes it or corrects and passes it again. An accepted control work is estimated by a lecturer.

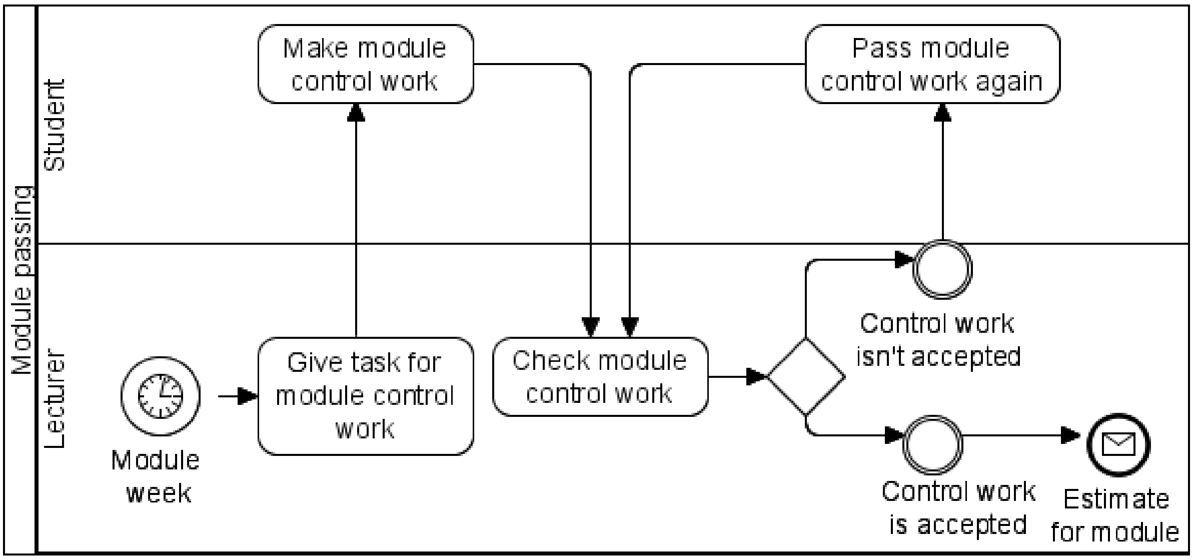

Module passing
Student
Lecturer
Make module control work
Pass module control work again
Module week
Give task for module control work
Check module control work
Control work isn't accepted
Control work is accepted
Estimate for module

Figure 5.51– Module passing (BPMN)

So the considered example shows that different approaches and notations can be used for business-processes formalization. Selection of methodology and CASE-tools depends on the modeling goals and tasks, requirements and restrictions of modeling process, and analyst's qualification and preferences.

Summary

This chapter considers the fundamentals of business-processes modeling. The methodologies of structural (IDEF) and object-oriented (UML) analyses and design of business-processes are presented. The additional attention is paid to graphical representation of business-processes based on BPMN notation. There is a case study of modeling of business-process of discipline learning in the university in IDEF0, IDEF3, DFD, UML, BPMN notations.

Tasks for case-study

1. Construct the models of business-processes which take place in the supermarket using IDEF, DFD, UML and BPMN notations.

2. Construct the models of business-processes which take place in the on-line railway tickets selling company using IDEF, DFD, UML and BPMN notations.
3. Construct the models of business-processes which take place in the on-line hotels booking company using IDEF, DFD, UML and BPMN notations.
4. Construct the models of business-processes which reflect internet-banking functioning using IDEF, DFD, UML and BPMN notations.
5. Construct the models of business-processes which take place in the modern library using IDEF, DFD, UML and BPMN notations.
6. Construct the models of business-processes which reflect on-line shop functioning using IDEF, DFD, UML and BPMN notations.
7. Construct the models of business-processes which reflect call-center functioning using IDEF, DFD, UML and BPMN notations.
8. Construct the models of business-processes which take place in the on-line surveys company using IDEF, DFD, UML and BPMN notations.
9. Construct the models of business-processes which take place in tourist company using IDEF, DFD, UML and BPMN notations.
10. Construct the models of business-processes which reflect mobile operator’s support functioning using IDEF, DFD, UML and BPMN notations.

Check questions

1. Give the definition of notion “business-process”.
2. What is the goal of business-processes modeling?
3. What types of business-processes do you know?
4. What characteristics does business-process have?
5. What does the business-process model consist of?
6. What are the purposes of AS-IS and TO-BE business-process models construction?
7. What are the advantages and disadvantages of SADT models application?
8. What methodologies belong to IDEF?
9. Provide the classification of UML diagrams.
10. What are the advantages and disadvantages of the most widespread CASE-tools for business-process modeling?
11. What for is IDEF0 notation used?
12. What are the peculiarities of system’s behavior modeling in IDEF3 notation?
13. What for data flow modeling is used?
14. How can business-processes be modeled using different UML diagrams?
15. What for can Visual Paradigm for UML be used?

16. What types of diagrams can be built with the help of Visual Paradigm for UML?
17. What notations of visual modeling are supported by Visual Paradigm for UML?
18. Describe the order of actions which are necessary for use-case diagram construction.
19. Describe the order of actions which are necessary for class diagram construction.
20. Describe the order of actions which are necessary for activity diagram construction.
21. Describe the order of actions which are necessary for component diagram construction.
22. Describe the order of actions which are necessary for sequence diagram construction.
23. What elements belong to different categories in BPMN?
24. What types of gates exist in BPMN?

REFERENCES

1. Макаров И.М. Целевые комплексные программы / И.М. Макаров, В.Б. Соколов, А.А. Абрамов. – М.: «Знание», 1980. – 135 с.
2. Месарович М. Общая теория систем и ее математические основы. Исследования по общей теории систем / И.М. Месарович. – М.: «Прогресс», 1969. – 321 с.
3. Акофф Р. Системы, организации и междисциплинарные исследования. Исследования по общей теории систем / Р. Акофф. – М.: «Прогресс», 1969. – 286 с.
4. Поспелов Г.С. Программно-целевое планирование и управление / Г.С. Поспелов, В.А. Ириков. – М.: «Сов. Радио», 1976. – 440 с.
5. Хорафас Д.Н. Системы и моделирование / Д.Н. Хорафас. – М.: «Мир», 1967. – 389 с.
6. Советов Б.Я. Моделирование систем / Б.Я. Советов, С.А. Яковлев. – М.: Высш. школа, 1985. – 315 с.
7. Цвиркун А.Д. Структура сложных систем / А.Д. Цвиркун. – М.: «Сов. радио», 1975. – 200 с.
8. Норенков И.П. Системы автоматизированного проектирования: учеб. пособие для вузов: В 9 кн. Кн.1. Принципы построения и структура / И.П. Норенков. – Мн.: Высш. шк., 1987. – 123 с.
9. Капустин Н.М. Системы автоматизированного проектирования : учеб. пособие для техн. вузов. В 9 кн. Кн. 6. Автоматизация конструкторского и технологического проектирования / Н.М. Капустин, Г.Н. Васильев. – Мн.: Высш. шк., 1988. – 191 с.
10. Фокс А. Вычислительная геометрия. Применение в проектировании на производстве / А. Фокс, М. Пратт. – М.: Мир, 1982. – 304 с.
11. Завьялов Ю.С. Методы сплайн-функций / Ю.С. Завьялов, Б.И. Квасов, В.Л. Мирошниченко. – М.: Наука, 1980. – 352 с.
12. Ларичев О.И. Качественные методы принятия решений / О.И. Ларичев, Е.М. Мовшович. – М.: Наука, 1996. – 268 с.
13. Растригин Л.А. Современные принципы управления сложными объектами / Л.А. Растригин. – М.: Сов. радио, 1980. – 232 с.

14. Бусленко Н.П. Лекции по теории сложных систем / Н.П. Бусленко, В.В. Калашников, И.Н. Коваленко. – М.: Сов. радио, 1973. – 440 с.
15. Красовский А.А. Системы автоматического управления полетом и их аналитическое конструирование / А.А. Красовский. – М.: Наука, 1973. – 356 с.
16. Первозванский А.А. Математическое модели в управлении производством / А.А. Первозванский. – М.: Наука, 1975. – 616 с.
17. Поспелов Д.А. Логические методы анализа и синтеза схем / Д.А. Поспелов. – М.: Энергия, 1968. – 318 с.
18. Закревский А.Д. Алгоритмы синтеза дискретных автоматов / А.Д. Закревский. – М.: Наука, 1971. – 450 с.
19. Вентцель Е.С. Теория вероятностей / Е.С. Вентцель. – М.: Наука, 1969. – 580 с.
20. Гамаюн И.П. Разработка имитационных моделей на основе сетей Петри : учеб. пособие / И.П. Гамаюн. – Харьков: НТУ «ХПИ», 2002. – 143 с.
21. Гнеденко Б.В. Введение в теорию массового обслуживания / Б.В. Гнеденко, И.Н. Коваленко. – М.: Наука, 1966. – 378 с.
22. Бусленко В.Н. Автоматизация имитационного моделирования сложных систем / В.Н. Бусленко. – М.: Наука, 1977. – 240 с.
23. Михалевич В.С. Вычислительные методы исследования и проектирования сложных систем / В.С. Михалевич, В.Л. Волкович. – М.: Наука, 1982. – 286 с.
24. Сергиенко И.В. Математические модели и методы решения задач дискретной оптимизации. – К.: Наукова думка, 1988. – 472 с.
25. Томашевский В.Н., Жданова Е.Г. Имитационное моделирование в среде GPSS. – М.: Бестселлер, 2003. – 416 с.
26. Питерсон Дж. Теория сетей Петри и моделирование систем. – М.: Мир, 1984. – 264 с.
27. Буч Г. Язык UML. Руководство пользователя / Г. Буч, Дж. Рамбо, А. Якобсон. – М.: ДМК Пресс, 2007. – 496 с.
28. Буч Г. UML 2-е издание / Г. Буч, А. Якобсон, Дж. Рамбо. – Санкт-Питербург: Питер, 2006. – 736 с.
29. XML Process Definition Language // http://www.xpdl.org.
30. BPEL: Who Needs It Anyway? // http://www.bpm.com/bpel-who-needs-it.html.
31. Бизнес-процесс //http://www.re-business.ru/standarts/bus_proc /index.html.
32. Основы формальных методов описания бизнес-процессов : учеб. пособие / К.Е. Самуйлов, Н.В. Серебренникова, А.В. Чукарин, Н.В. Яркина. – М.: РУДН, 2008. – 130 с.

33. Маклаков С.В. Моделирование бизнес-процессов с BPWin 4.0 / С.В. Маклаков. – М.: Диалог-МИФИ, 2002. – 209 с.
34. Вендров А.М. Современные технологии создания программного обеспечения (обзор) / А.М. Вендров // Jet Info Информационный бюллетень. – 2004. – №4(131).– с. 2–32.
35. Черемных С.В. Моделирование и анализ систем. IDEF-технологии: практикум / С.В. Черемных, И.О. Семенов, В.С. Ручкин. – М.: Финансы и статистика, 2006. – 192 с.
36. Business Modeling with UML: Business Patterns at Work. – John Wiley & Sons, 2000. – 459 p.
37. W.M.P. van der Aalst. Formalization and Verification of Event-driven Process chains // http://www.alexandria.tue.nl/extra1/wskrap/publichtml/9714860.pdf.
38. Business Process Model and Notation (BPMN). Version 2.0 // http://www.omg.org/spec/BPMN/2.0.
39. Войнов Н. Практика использования BPMN / Н. Войнов // http://nvoynov.blogspot.com/2007/10/bpmn.html.
40. Assaf Arkin. Business Process Modeling Language // http://www.bpmn.org/Documents/BPML-2003.pdf.

Навчальне видання

ГАМАЮН Ігор Петрович
ЧЕРЕДНІЧЕНКО Ольга Юріївна

МОДЕЛЮВАННЯ СИСТЕМ

Навчальний посібник для студентів спеціальностей
6.050103 “Програмна інженерія”
6.050101 “Комп’ютерні науки”

(Англ. мовою)

Відповідальний за випуск *М. Д. Годлевський*

Роботу до видання рекомендував *О. В. Горілий*

Формат 60×84/16. Ум. друк. арк. 13,02. Наклад 300 прим. Зам. № 20_07-1.

Видавництво «Факт»
Україна, 61166, м. Харків, вул. Бакуліна, 11, оф. 4-28.
Тел./факс: (057)756-43-75. E-mail: publish_fakt@mail.ru
Свідоцтво суб’єкта видавничої справи ДК № 3172 від 22.04.2008

Виготовлено у ФОП В. Є. Гудзинський
Україна, 61072, м. Харків, вул. 23-го Серпня, 27.
Тел./факс: (057)340-52-26. E-mail: fop_veg@ukr.net
Свідоцтво суб’єкта видавничої справи ХК № 269 від 23.11.2010