***Розглянуто задачу розробки математичних моделей патернів проектування вимог до системи. Розроблені моделі дозволяють формалізувати представлення вимог до системи на рівні знань. Використання даних моделей дозволяє підвищити ефективність виконання IT-проектів створення інформаційних систем за рахунок повторного використання елементів, що були розроблені в попередніх IT-проектах***

***Ключові слова: IT-послуга, потреби, функціональні вимоги, повторне використання, патерн проектування вимоги, фрейм, кортеж***

***Рассмотрена задача разработки математических моделей паттернов проектирования требований к системе. Разработанные модели позволяют формализовать представления требований к системе на уровне знаний. Применение данных моделей позволяет повысить эффективность выполнения IT-проектов создания информационных систем за счет повторного использования элементов, разработанных в предыдущих IT-проектах***

***Ключевые слова: IT-услуга, потребности, функциональные требования, повторное использование, паттерн проектирования требования, фрейм, кортеж***

UDC 044.03; 658.11.05.06
DOI: 10.15587/1729-4061.2017.110586

# DEVELOPING THE MODELS OF PATTERNS IN THE DESIGN OF REQUIREMENTS TO AN INFORMATION SYSTEM AT THE KNOWLEDGE LEVEL

**V. Levykin**
Doctor of Technical Science, Professor*
E-mail: viktor.levykin@nure.ua
**M. Ievlanov**
Doctor of Technical Science, Associate Professor*
E-mail: maksym.ievlanov@nure.ua
**O. Neumyvakina**
PhD, Leading Researcher*
E-mail: olga.neumyvakina@nure.ua
*Department of Information Control Systems
Kharkiv National University of Radio Electronics
Nauky ave., 14, Kharkiv, Ukraine, 61166

## 1. Introduction

Current interest in what lies behind the concept of "architecture of a system", in contrast to academic interest of the previous years, acquires a distinctly applied character. Companies want to reduce unproductive expenses for purchasing and implementation of individual elements of information systems (IS). Authorities consider subdivisions, supporting and operating IS as centers of costs and seek to exclude "extra" costs from total costs of management of business processes of an enterprise. These and other factors determine the need to address the problem of architectural integration of non-homogeneous elements of IS at a formal level. This level is available for implementation of modern IS as a separate component. The main purpose of this component is to reduce the costs of introduction and operation of particular functional elements of IS through automating of execution of operations of integration, search and elimination of contradictions between these elements.

The most significant economic benefit from integration of elements into a single unified IS occurs if this operation is performed during formation and analysis of demands for this IS. In addition, in the course of this operation, it becomes possible to solve the problem of determining a possibility to reuse the elements of previously created IS for the creation of a new IS. Reuse of the elements makes it possible to reduce significantly the cost of creation IS and particular kinds of services (mainly information and software).

The aspect of reuse of requirements is becoming particular significant in terms of designing, implementation, operation and modernization of IS as a set of IT-services. These services are provided to consumers for automating of business processes of enterprises or managing these processes. Effect of reuse of IT-services, subject to minimum changes of a providing part, can be considered directly proportional to the degree of reuse requirements for these IT-services, including transformations of descriptions in terms of other subject areas. It should be taken into account that changes of requirements, arising in the course of IS creation, are inevitable. Such a situation will demand formal representations of requirements and mechanisms of their management, capable to be implemented in specific information technologies, controlling integration processes of heterogeneous IS elements. That is why the problem of creation of such concepts and mechanisms is still relevant and requires solution.

## 2. Literature review and problem statement

The focus of researchers in the field of IS requirements descriptions today is on development of methods and tools, allowing creation and processing of formal models of requirements to a system [1–3]. By a model of requirement, we imply a complex of requirements' descriptions in ways that enable us, based on these descriptions, to perform necessary operations within the LC of IS using specific methodologies and information technologies [4]. Examples of such studies include:

a) description of application of the developed method for analysis of requirements for ERP-IS, presented in [5];

b) problems of development and application of the method for scenario-based analysis of requirements for a system, considered in [6];

c) application of a process model for identification of requirements for specialized IS of detection and processing of knowledge from business information, described in [1];

d) a variant of solution of the problem of automation of operations on formalizing requirements for IS, explored in [7].

However, analysis of the above examples of research proves the following: Would-be users have non-formal needs when it comes to created IS and technologies. And the problem of transformation of these needs into formal models of requirements for created IS and technologies is solved primarily at a conceptual level.

One of the promising directions in this field is exploration of models and methods for extracting knowledge from descriptions of needs and requirements for a system [8]. One of the options of formalization of a subject area (SA) of corporative IS through semantic modeling is discussed in [9]. Paper [8] also examines the issue of knowledge mining techniques for conversion of expressed needs into a set of descriptions of IS requirements in healthcare.

However, an analysis of studies, devoted to mining knowledge from requirements, shows that the problem of transformation of informal requirements into formal requirements models is solved primarily at the conceptual level [10]. The lack of a unified definition of "a requirement for a system" should be considered one of the main reasons of it [11].

At the same time, there is considerable practical experience in development and operation of IT-products, used for generation, analysis and management of requirements for a system – requirement management systems [12–15]. However, analysis of this experience in research [16] shows that none of the existing requirement management systems is oriented towards automation of synthesis of description of the architecture of a created system. In addition, the problem of decision making about reusing of previously implemented requirements in new IT projects remains practically unresolved in these systems.

On the whole, according to results of conducted analysis, it is possible to derive the following conclusion: so far, the problem of formal description of requirements, methods of formation and analysis – also at the level of knowledge, mined form requirements – does not have any acceptable solution. The same formal representations of requirements, which are used to develop systems of requirement management, place emphasis on description of particular requirements as artifacts of the IS, which is being created or updated. That is why, attempts of IS synthesis based on solutions, implementing particular requirements, most often give rise to the effect of "IT-blindness", which is the inability of existing IS and IT to "see" and assess actual processes in the environment, they belong to [17].

## 3. The aim and objectives of the study

The aim of present research is to develop formal description of representation of knowledge, mined from requirements and operations on these representations. In this case, operations on formal descriptions of knowledge representations are proposed to be considered as mappings, transferring the mentioned descriptions into each other. This will make it possible to describe artifacts and operations of information technology of IS requirements management based on a single mathematical apparatus. Application of these artifacts and operations will allow reduction of labor costs and time consumption due to identifying of previously implemented requirements in the course of initiation and early planning of new IT projects of IS creation.

To accomplish the set goal, the following tasks had to be solved:

– development of models of structural IS design requirements patterns at the knowledge level;

– development of models of behavioral IS design requirements patterns at the knowledge level.

## 4. Modified frame-based knowledge model

It is proposed to regard ontologies of an element of a controlled object or a process, IS element or IS as a whole, for which a requirement is posed, as an IS requirement representation at the knowledge level. Specific features of representation of requirements for particular elements of the created IS are addressed in [18]. At the same time, the desire to reuse IS requirements necessitates additional separation of ontologies of IS elements or IS as a whole, implemented in previously completed IS creation projects. On the whole, it is proposed to consider the following types of ontologies:

a) ontologies of a subject-area (SA), which represent knowledge of automated objects and business process (BP), obtained in the course of identification and analysis of IS requirements;

b) ontologies of implemented IS requirements that represent knowledge of structures of data and IP processes, IT-products and IT-services, created within the frameworks of previous projects;

c) ontologies of requirements for a created IS that represent knowledge of structures of data and processes of IS, IT-products and IT-services, separated in the course of formation and analysis of requirements within the frameworks of the current IS creation project.

These ontologies are supposed to be formed based of the frame-based knowledge model. Application of this model is caused by the following considerations:

– usage of a frame-based knowledge model allows application of a unified mathematical apparatus for describing knowledge about SA with a view to formal IS requirements representation;

– usage of a frame-based knowledge model allows application of a unified mathematical apparatus for describing knowledge, implemented in IS elements in the form of IS models of this system;

– usage of a frame-based model allows implementation of one-to-one mapping of requirements representation for created knowledge-based IS, in software elements of this IS.

The majority of modern database management systems (DBMS) are based on a relational data model. That is why solution of the problem of object-relational mapping with the use of a frame-based knowledge model of allows subsequent one-to-one mapping of requirements for SA and software elements into IA of the created IS [19, 20].

A frame is a data structure for representation of a stereotyped situation. In methodology of object-oriented programming (OOP), this notion corresponds to class [21].

As a rule, frame SA models are represented as a frame network comprising nodes and different relationships between them [21]:

$$M =< FR, C, G >, \quad (1)$$

where

$$FR = \{fr_1, ..., fr_h\}$$

is the set of information units (frames);

$$C = \{C_1, C_2, ..., C_n\}$$

is the set of links and relations between information units (hierarchical, reference, etc.); $G$ is the set of mappings that assign relations from the assigned set $\{C_1, C_2, ..., C_n\}$ (both hierarchical inheritance relations, and horizontal relations of frame associations) among information units, belonging to set $FR$; $G_i \in G =< fr_{i1}, fr_{i2}, C_i >$ [22].

Frames are divided into frames-prototypes (correspondent to classes in OOP and tables in databases) and frames-instances (correspondent to instances of classes – objects and entries in database tables). By the nature of relationships, frames are classified in the following way: "subframes, frames and superframes are hierarchically ordered elements, forming frame systems". In OOP technology, there are also hierarchical and referential relationships, and subclasses (descendants-classes) and superclasses (parents-classes) correspond to concepts of "subframe" and "superframes". Similar to a frame model, it is possible to represent a hierarchy of classes (a class diagram in UML), in which parents-classes define a set of fields and functionality, inherent in all classes-descendants in the form of a network [21, 23–26].

Frame $fr \in FR$ can be described by a structured set, having the form [24]:

$$fr = \{n, [(ns_1, vs_1, ps_1), (ns_2, vs_2, ps_2), ..., (ns_k, vs_k, ps_k)]\}, \quad (2)$$

where $n$ is the name of a frame; *(ns, vs, ps)* is the slot of a frame; $k$ is the number of frame's slots; $ns_j$ is the name of a slot, $i = \overline{1,k}$; $vs_i$ is the value of the slot, $i = \overline{1,k}$; $ps_i$ is the name of the attached procedure, $i = \overline{1,k}$.

A subprogram of the procedural type is used as the value of slot "the name of the attached procedure". In OOP, methods of classes are associated with attached procedures, in relational databases, they are associated with triggers, procedures and functions, related to tables [27].

However, a frame-based model in the classic form, represented by expression (2), does not exactly correspond to special features of representation of knowledge about SA, and especially on the created IS. Similar features are mainly determined by paradigms of system designing that a Provider puts as a base of the vast majority of design decisions of IS as a whole and separate IT-services. Structural and object-oriented approaches are typically specified as such paradigms. These approaches necessitate upgrading of frame-based model of knowledge representation. The objective of this upgrading is to use a frame-based model for description of transformation of knowledge about SA, IS, individual IT-products and services in patterns and specific elements of the types of support, created by IS.

First of all, it is proposed to expand the basic concept of "frame" by separation as an individual frame's unit of a totality of all methods (attached procedures) $Mt = \{mt_1, ..., mt_z\}$, associated with a frame as a whole, rather than with specific slots.

In addition, it is proposed to expand the concept of "frame" by introduction of the additional concept of "frame interface", corresponding to the concept of "class interface" in OOP methodology [27]. It should be noted that some programming languages (such as Object Pascal) support the possibility to define properties in interfaces of classes. In other programming languages (in particular, Java, C++), properties can be declared by assigning *GetPropertyName* and *SetPropertyName* methods. These methods provide access to fields of a class. Interfaces, like classes, can contain reference properties and be inherited with formation of hierarchies [23, 26, 28].

In relational databases, there is no concept, which completely corresponds to the term "class interface". The term "representation" is the closest to it in terms of participation in organization of interaction between various IS components.

The above allows us to define the term "frame interface" as declarative announcement of a set of properties and methods without detailed description (as well as without detailed description of attached procedures). Each interface of a frame is supposed to describe a separate point of view on the frame or a subset of frames, and this point of view may not perceive these frames or a subset of frames completely.

According to the proposed definition, formalized description of frame interface *if* will be a structured set in the form of [27]:

$$if =< g, \{ns\_if_1, ..., ns\_if_n\}, \{nm_1, ..., nm_s\} >, \quad (3)$$

where $g$ is the Globally Unique IDentifier (GUID); $\{ns\_if_1, ..., ns\_if_n\}$ is the set of declarations of slots in the frame's interface; $\{nm_1, ..., nm_s\}$ is the set of declarations of methods in the frame's interface.

It should be noted that for any declared slot of the frame's interface, it is possible to match a set of values both of a separate frame $fr \in Fr$, and of a separate slot $ns_i \in fr$. Subsequently, the frame, participating in formation of interface *if*, will be a generating frame

The process of formation of interface *if* from a subset of generating frames $FR_{par} \subset FR$ can be described by mapping $F^{if}_{FR_{par}} : FR_{par} \to if$. This mapping is bi-active, since each specific element of interface *if* can have only one frame or a frame's slot from subset $Fr_{par}$ as a parent.

Strictness of selection of a subset of generating frames is caused by inappropriateness of appearance of frame's interface of "God Almighty", i. e. containing elements of all the frames that make up a specific network.

Then formalized description of the concept of "frame" (2) taking into account the proposed modifications will take the form [27]:

$$fr = \{n, [(ns_1, vs_1, ps_1), ..., (ns_k, vs_k, ps_k)], \{if_1, ..., if_n\}, \{mt_1, ..., mt_z\}\}, \quad (4)$$

where $\{if_1, ..., if_n\}$ *is* the set of interfaces, used by frame *fr* (can be empty).

Usage of the modified formalized description of frame (4) allows making the following statement. A network of frames (1) will include:

– knowledge of data structures, presented in the form of frames;

– knowledge of processes, in the course of which the above-mentioned structures will interact.

Such knowledge is represented in the network in the form of frames' interfaces and methods. In addition, the use of a modified description of a frame allows us to formalize descriptions of knowledge about SA and created IS, as well as descriptions of one-to-one mappings of this knowledge.

## 5. Results of development of models of information system requirement design patterns at the knowledge level

### 5. 1. Results of development of models of structural information system requirement design patterns at the knowledge level

To develop a model of a frame network as IS requirement representation at knowledge level, we will use a generalized model of formulated IS requirements $L_{IS}^{tr}$, described in [28]. Then the model of IS requirement representation at the knowledge level should be represented as a formalized description of each element of subclass $K_{IS}^{tr}$ with subsequent refinement after categorizing this element to one of the following subclasses: $K_{IS}^{B}$ (a subclass of business requirements), $K_{IS}^{IB}$ (a subclass of requirements for IS as an aspect of business), $K_{IS}^{s}$ (a subclass of requirements for IS as a whole), $K_{IS}^{f}$ (a subclass of functional requirements), $K_{IS}^{nf}$ (a subclass of non-functional requirements), $I_{IS}^{fw}$ (a subclass of functional requirements for IT-services), $K_{IS}^{nfw}$ (a subclass of non-functional requirements for IT-services). Based on this view, the model of formulated IS requirements representation at the knowledge level will be a tuple of attributes, which is structurally divided into two parts:

$$M_{K_{IS}^{tr}} = \left\langle < M_K^{At_{Pr}} >, < M_K^{At_{gr}} > \right\rangle, \quad (5)$$

where $M_{K_{IS}^{tr}}$ is the model of a subclass of formulated IS requirements representations at the knowledge level; $< M_K^{At_{Pr}} >$ is the tuple of elements of attributive model of IS requirements, which are determined by IS design requirements pattern at knowledge level and are compulsory for requirements of any group; $< M_K^{At_{gr}} >$ is the tuple of elements of attributive IS requirement model, which are determined based on individual features of executing of processes by a Provider and a Consumer, directly working with requirements representations of a specific group at the knowledge level.

Decision on formation of the above types of ontologies based on a modified frame model of knowledge (4) as a network of frames (1) necessitates separation of the following structural IS requirement design patterns at the knowledge level:

a) structural frame design pattern;

b) structural frame's interface design pattern;

c) structural pattern of designing relations between nodes of frames' network;

d) generalized structural frames' network design pattern.

The model, describing the structural frame design pattern, $Pt_{fr\_str}$, in a general case will take the form:

$$Pt_{fr\_str} =< At_n, At_{el\_fr}, At_{el\_fr\_t}, < at_n, at_{el\_fr}, at_{el\_fr\_t} >>, \quad (6)$$

where $At_n$ is the tuple of attributes, describing the name of a frame; $At_{el\_fr}$ is the tuple of attributes, describing the element of a frame (slot, interface, method); $At_{el\_fr\_t}$ is the tuple of attribute, describing the type of the frame's element; $at_n$ is the attribute, identifying the name of a frame; $at_{el\_fr}$ is the attribute, identifying the element of a frame; $at_{el\_fr\_t}$ is the attribute, identifying the type of the element of a frame.

The model, describing a structural frame's interface design pattern $Pt_{if}$, in a general case will take the form:

$$Pt_{if} =< At_g, At_{el\_if}, At_{el\_if\_t}, < at_g, at_{el\_if}, at_{el\_if\_t} >>, \quad (7)$$

where $At_g$ is the tuple of attributes, describing globally unique identifier of the frame's interface; $At_{el\_if}$ is the tuple of attributes, describing the element of frame's interface (slot, method); $At_{el\_if\_t}$ is the tuple of attributes, describing the type of the element of the frame's interface; $at_g$ is the attribute, identifying globally unique identifier of the frame's interface; $at_{el\_if}$ is the attribute, identifying the element of the frame's interface; $at_{el\_if\_t}$ is the attribute, identifying the type of the element of the frame's interface.

The model, describing the structural pattern of designing relations between nodes on a network of frames, $Pt_{fr\_rel}$ in a general case will take the form:

$$Pt_{fr\_rel} =< At_{fr\_rel\_n}, At_{el\_fr\_rel}, At_{el\_fr\_rel\_t}, < at_{fr\_rel\_n}, at_{el\_fr\_rel}, at_{el\_fr\_rel\_t} >>, \quad (8)$$

where $At_{fr\_rel\_n}$ is the tuple of attributes, describing the name of relationship; $At_{el\_fr\_rel}$ is the tuple of attributes, describing the element of description of relationship; $At_{el\_fr\_rel\_t}$ is the tuple of attributes, descri bing the type of the element of description of relationship; $at_{fr\_rel\_n}$ is the attribute, identifying the name of relationship; $at_{el\_fr\_rel}$ is the attribute, identifying the element of description of relationship; $at_{el\_fr\_rel\_t}$ is the attribute, identifying the type of element of description of relationship.

Descriptions of possible types of relationships between objective and structural models of entities are listed in [29].

Based on these models of structural patterns of designing of elements of frame's network, we can conclude that the model, describing the structural frame's network design pattern (1), $Pt_{net\_fr}$ in a general case will take the form:

$$Pt_{net\_fr} =< Pt_{fr\_str}, Pt_{if}, Pt_{fr\_rel}, < at_n^1, at_n^2, at_{if}, at_{fr\_rel\_n} >>, \quad (9)$$

where $at_n^1$ is the attribute, identifying the name of the first frame, which can participate in relationship formation (may be undefined); $at_n^2$ is the attribute, identifying the name of the second name, which can participate in relationship formation (may be undefined).

Attribute $at_{if}$ can also be undefined, since each particular relationship can exist only between two frames, or between one frame and an interface. Condition that restricts existence of relationships between nodes of a frame network will take the form:

$$\forall (at_n^1, at_n^2, at_{if}) \in < at_n^1, at_n^2, at_{if}, at_{fr\_rel\_n} > (at_n^1 \bigcap at_n^2) \oplus (at_n^1 \bigcap at_{if}) \oplus (at_n^2 \bigcap at_{if}) \oplus \oplus (at_{if} \bigcap at_n^1) \oplus (at_{if} \bigcap at_n^2) = 1. \quad (10)$$

Models (6)–(9) form a set of structural IS requirements design pattern at the knowledge level as a subclass of objects of categorical-theoretic model of IS requirements design patterns $K_{IS}^{Pt}$ [30], assigning a particular type of element $< M_K^{At_{Pt}} >$ of model (5). This subclass of patterns in general case will take the form:

$$K_{IS}^{Pt} = \{Pt_{fr\_str}, Pt_{if}, Pt_{fr\_rel}, Pt_{net\_fr}\} =$$
$$= \{< At_n, At_{el\_fr}, At_{el\_fr\_t}, < at_n, at_{el\_fr}, at_{el\_fr\_t} >>,$$
$$< At_g, At_{el\_if}, At_{el\_if\_t}, < at_g, at_{el\_if}, at_{el\_if\_t} >>,$$
$$< At_{fr\_rel\_n}, At_{el\_fr\_rel}, At_{el\_fr\_rel\_t},$$
$$< at_{fr\_rel\_n}, at_{el\_fr\_rel}, at_{el\_fr\_rel\_t} >>,$$
$$< at_n^1, at_n^2, at_{if}, at_{fr\_rel\_n} >\}. \quad (11)$$

**5. 2. Results of development of models of behavioral information system requirements design patterns at the knowledge level**

Subclass of morphisms $H(K_{IS}^{Pt})$ of categorical-theoretic model of IS requirements design patterns, explored in [30], describes a set of behavioral IS requirements design patterns at the knowledge level and consists of the following types of morphisms:

a) single morphisms $1_{Pt_{fr\_str}}$, $1_{Pt_{if}}$, $1_{Pt_{fr\_rel}}$, representing structural patterns $Pt_{fr\_str}$, $Pt_{if}$ and $Pt_{fr\_rel}$, respectively, in themselves;

b) morhisms, establishing relationships between structural patterns $Pt_{fr\_str}$, $Pt_{if}$, $Pt_{fr\_rel}$ and $Pt_{net\_fr}$.

Single morphisms do not exist for structural pattern $Pt_{net\_fr}$, because this pattern inherit characteristics of patterns $Pt_{fr\_str}$, $Pt_{if}$ and $Pt_{fr\_rel}$.

For the model of pattern $Pt_{fr\_str}$, conditions of existence of single morphisms are similar to conditions, described in [28]. Models of these morphisms will, accordingly, take the form:

$$1_{Pt_{fr\_str}}^{add} : [At_n \oplus At_x] \oplus [At_{el\_fr} \oplus At_x] \oplus$$
$$\oplus [At_{el\_fr\_t} \oplus At_x] \oplus [\varnothing \oplus At_x], \quad (12)$$

$$1_{Pt_{fr\_str}}^{upd} : [(At_n \setminus At_x) \cup At'_x] \oplus [(At_{el\_fr} \setminus At_x) \cup At'_x] \oplus$$
$$\oplus [(At_{el\_fr\_t} \setminus At_x) \cup At'_x], \quad (13)$$

$$1_{Pt_{fr\_str}}^{del} : [At_n \setminus At_x] \oplus [At_{el\_fr} \setminus At_x] \oplus [At_{el\_fr\_t} \setminus At_x]. \quad (14)$$

For a model of pattern $Pt_{if}$, conditions of existence of single morphisms are similar to conditions, described in [28], and models of these morphisms will respectively take the form:

$$1_{Pt_{if}}^{add} : [At_g \oplus At_x] \oplus [At_{el\_if} \oplus At_x] \oplus$$
$$\oplus [At_{el\_if\_t} \oplus At_x] \oplus [\varnothing \oplus At_x], \quad (15)$$

$$1_{Pt_{if}}^{upd} : [(At_g \setminus At_x) \cup At'_x] \oplus [(At_{el\_if} \setminus At_x) \cup At'_x] \oplus$$
$$\oplus [(At_{el\_if\_t} \setminus At_x) \cup At'_x], \quad (16)$$

$$1_{Pt_{if}}^{del} : [At_g \setminus At_x] \oplus [At_{el\_if} \setminus At_x] \oplus [At_{el\_if\_t} \setminus At_x]. \quad (17)$$

For model of pattern $Pt_{fr\_rel}$, conditions of existence of single morphisms are similar to conditions, described in [28] and models of these morphisms will respectively take the form:

$$1_{Pt_{fr\_rel}}^{add} : [At_{fr\_rel\_n} \oplus At_x] \oplus [At_{el\_fr\_rel} \oplus At_x] \oplus$$
$$\oplus [At_{el\_fr\_rel\_t} \oplus At_x] \oplus [\varnothing \oplus At_x], \quad (18)$$

$$1_{Pt_{fr\_rel}}^{upd} : [(At_{fr\_rel\_n} \setminus At_x) \cup At'_x] \oplus$$
$$\oplus [(At_{el\_fr\_rel} \setminus At_x) \cup At'_x] \oplus$$
$$\oplus [(At_{el\_fr\_rel\_t} \setminus At_x) \cup At'_x], \quad (19)$$

$$1_{Pt_{fr\_rel}}^{del} : [At_{fr\_rel} \setminus At_x] \oplus [At_{el\_fr\_rel} \setminus At_x] \oplus$$
$$\oplus [At_{el\_fr\_rel\_t} \setminus At_x]. \quad (20)$$

Of all morphisms, establishing relationships between models of structural patterns $Pt_{fr\_str}$, $Pt_{if}$, $Pt_{fr\_rel}$ and $Pt_{net\_fr}$, only morphisms $H(Pt_{fr\_str}, Pt_{if})$, $H(Pt_{fr\_str}, Pt_{fr\_rel})$, $H(Pt_{fr\_str}, Pt_{net\_fr})$, $H(Pt_{if}, Pt_{fr\_str})$, $H(Pt_{if}, Pt_{fr\_rel})$, $H(Pt_{if}, Pt_{net\_fr})$, and $H(Pt_{fr\_rel}, Pt_{net\_fr})$ can exist. Existence of these morphisms is caused by impossibility of appearance of descriptions of relationships between nodes of the network of frames without prior appearance of description of separate frames and interfaces, that are the nodes of this network.

Morphism $H(Pt_{fr\_str}, Pt_{if})$ in a general case takes the form:

$$H(Pt_{fr\_str}, Pt_{if}) : \{At_n^{fr\_str}, At_{el\_fr}^{fr\_str}, At_{el\_fr\_t}^{fr\_str}\} \rightarrow$$
$$\rightarrow \{At_{el\_if}^{if}, At_{el\_if\_t}^{if}\} \quad (21)$$

and exists only if the condition is satisfied

$$\forall \{At_n^{fr\_str}, At_{el\_fr}^{fr\_str}, At_{el\_fr\_t}^{fr\_str}\}$$
$$[\{At_n^{fr\_str}\} \rightarrow \{At_{el\_if}^{if}\}] \oplus [\{At_{el\_fr}^{fr\_str}, At_{el\_fr\_t}^{fr\_str}\} \rightarrow$$
$$\rightarrow \{At_{el\_if}^{if}, At_{el\_if\_t}^{if}\}] = 1, \quad (22)$$

where $At_n^{fr\_str}$ is the subset of attributes, describing the name of the frame in the model of structural pattern $Pt_{fr\_str}$; $At_{el\_fr}^{fr\_str}$ is the subset of attribute, describing the element of the frame in the model of structural pattern $Pt_{fr\_str}$; $At_{el\_fr\_t}^{fr\_str}$ is the subset, describing the type of the element of frame in the model of structural pattern $Pt_{fr\_str}$; $At_{el\_if}^{if}$ is the subset of attributes, describing the element of frame's interface in the model of structural pattern $Pt_{if}$; $At_{el\_if\_t}^{if}$ is the subset of attributes, describing the type of element of frame's interface in the model of structural pattern $Pt_{if}$.

Morphism $H(Pt_{fr\_str}, Pt_{fr\_rel})$ in a general case takes the form:

$$H(Pt_{fr\_str}, Pt_{fr\_rel}) : \{At_n^{fr\_str}, At_{el\_fr}^{fr\_str}, At_{el\_fr\_t}^{fr\_str}\} \rightarrow$$
$$\rightarrow \{At_{el\_fr\_rel}^{fr\_rel}, At_{el\_fr\_rel\_t}^{fr\_rel}\}, \quad (23)$$

where $At_{el\_fr\_rel}^{fr\_rel}$ is the subset of attributes, describing the element of relationship description in the model of structural pattern $Pt_{fr\_rel}$; $At_{el\_fr\_rel\_t}^{fr\_rel}$ is the subset of attributes, describing the type of relationship description element in the model of structural pattern $Pt_{fr\_rel}$.

Morphism $H(Pt_{fr\_str}, Pt_{net\_fr})$ in a general case takes the form:

$$H(Pt_{fr\_str}, Pt_{net\_fr}) : \{At_n^{fr\_str}, At_{el\_fr}^{fr\_str}, At_{el\_fr\_t}^{fr\_str}\} \rightarrow$$
$$\rightarrow \{At_n^{net\_fr}, At_{el\_fr}^{net\_fr}, At_{el\_fr\_t}^{net\_fr}\}, \quad (24)$$

where $At_n^{net\_fr}$ is the subset of attributes, describing the name of the frame in the model of structural pattern $Pt_{net\_fr}$; $At_{el\_fr}^{net\_fr}$ is the subset of attributes, describing the element

of the frame in the model of structural pattern $Pt_{net\_fr}$; $At^{net\_fr}_{el\_fr\_t}$ is the subset of attributes, describing the type of the element of the frame in the model of structural pattern $Pt_{net\_fr}$.

Morphism $H(Pt_{if}, Pt_{fr\_str})$ in a general case takes the form:

$$H(Pt_{if}, Pt_{fr\_str}): \{At^{if}_{g}\} \rightarrow \{At^{fr\_str}_{el\_fr}\}, \quad (25)$$

where $At^{if}_{g}$ is the subset of attributes, describing globally unique identifier of interface of a frame in the model of structural pattern $Pt_{if}$.

Morphism $H(Pt_{if}, Pt_{fr\_rel})$ in a general case takes the following form:

$$H(Pt_{if}, Pt_{fr\_rel}): \{At^{if}_{g}, At^{if}_{el\_if}, At^{if}_{el\_if\_t}\} \rightarrow$$
$$\rightarrow \{At^{fr\_rel}_{el\_fr\_rel}, At^{fr\_rel}_{el\_fr\_rel\_t}\}. \quad (26)$$

Morphism $H(Pt_{if}, Pt_{net\_fr})$ in a general case takes the form:

$$H(Pt_{if}, Pt_{net\_fr}): \{At^{if}_{g}, At^{if}_{el\_if}, At^{if}_{el\_if\_t}\} \rightarrow$$
$$\rightarrow \{At^{net\_fr}_{g}, At^{net\_fr}_{el\_if}, At^{net\_fr}_{el\_if\_t}\}, \quad (27)$$

where $At^{net\_fr}_{g}$ is the subset of attributes, describing globally unique identifier of interface of a frame in the model of structural pattern $Pt_{net\_fr}$; $At^{net\_fr}_{el\_if}$ is the subset of attributes, describing the element of frame's interface in the model of structural pattern $Pt_{net\_fr}$; $At^{net\_fr}_{el\_if\_t}$ is the subset of attributes, describing the type of the element of the frame's interface in the model of structural pattern $Pt_{net\_fr}$.

Morphism $H(Pt_{fr\_rel}, Pt_{net\_fr})$ in a general case takes the form:

$$H(Pt_{fr\_rel}, Pt_{net\_fr}): \{At^{fr\_rel}_{fr\_rel\_n}, At^{fr\_rel}_{el\_fr\_rel}, At^{fr\_rel}_{el\_fr\_rel\_t}\} \rightarrow$$
$$\rightarrow \{At^{net\_fr}_{fr\_rel\_n}, At^{net\_fr}_{el\_fr\_rel}, At^{net\_fr}_{el\_fr\_rel\_t}\}, \quad (28)$$

where $At^{net\_fr}_{fr\_rel\_n}$ is the subset of attributes, describing the name of relationship in the model of structural pattern $Pt_{net\_fr}$; $At^{net\_fr}_{el\_fr\_rel}$ is the subset of attributes, describing the element of description of relationships in the model of structural pattern $Pt_{net\_fr}$; $At^{net\_fr}_{el\_fr\_rel\_t}$ is the subset of attributes, describing the type of element of description of relationship in the model of structural pattern $Pt_{net\_fr}$.

It should be noted that for the case of using particular versions of IS requirements, morphisms, making up subclass $H(K^{Pt}_{IS})$, will take a similar form.

## 6. Discussion of results of development of models of information system requirements design pattern at the knowledge level

Model (6)–(9) define the basic structural features and basic content of data showcases, designed to store the following information:

a) historical information about frames as network nodes (model (6));

b) historical information about frames' interfaces as nodes of frames' network (model (7));

c) historical information about relationships between nodes of frames' networks (model (8));

d) historical information about variants of frames' network of created IS (model (9)).

Model (11) defines the basic structural features and basic content of the data storage fragment, providing storage of historical information about IS requirements representation at the knowledge level. The scheme of this fragment is the result of integration of schemes of data showcases, described above.

Models of single morphisms (expressions (12)–(14), (15)–(17) and (18)–(20)) define the main features of the following operations

a) addition of new elements to models of structural requirements design patterns (expressions (12), (15) and (18));

b) modification of descriptions of elements of models of structural requirements design patterns (expressions (13), (16) and (19));

c) deletion of unused elements of models of structural requirements design patterns (expressions (14), (17) and (20)).

The issue of existence of morphisms $H(Pt_{net\_fr}, Pt_{fr\_str})$, $H(Pt_{net\_fr}, P_{tif})$ and $H(Pt_{net\_fr}, Pt_{fr\_rel}$ requires further research. The existence of these morphisms can be attributed to special methods of automatic knowledge mining based on the results of analysis of the generated network of frames, describing requirements for created IS.

An equally important problem, requiring subsequent study, is also the problem of existence of morphisms of type $H(Pt_{net\_fr}, Pt'_{net\_fr})$ and $H(Pt'_{net\_fr}, Pt_{net\_fr})$ where $Pt_{net\_fr}$ is the structural pattern of designing a network of frames, describing formulated requirements for created IS and $Pt'_{net\_fr}$ is the structural pattern of designing a network of frames, describing SA or implemented requirements for IS. The existence of these morphisms can be explained by availability of methods for automatic generation or modification of a network of frames. These methods can be used to solving the following tasks:

a) formation or modification of a network of frames, describing formulated requirements for created IS, according to results of solution of problems of analysis of conformity of specific formulated requirements for created IS with implemented requirements for earlier developed IS;

b) formation or modification of networks of frames, describing SA or implemented requirements for IS, based on results of successful completion of IS creation project, for which knowledge about formulated of implemented requirements are described in the form of a correspondent network of frames.

Solution of the problem of the first type allows automation of formation of a network of frames, which describes formulated requirements for created IS. In the course of formation of this network of frames, we take into account whether reuse of requirements that were implemented in previously completed IT projects of IS creation is worthwhile.

Solution of the problem of the second type allows automation of processes of expansion or modification of a network of frames, describing SA and implemented requirements for IS, taking into account knowledge, obtained from requirements of the implemented IS creation project.

## 7. Conclusions

1. To develop a model of IS requirements design patterns at the knowledge level, it was proposed to use a modified frame-based knowledge model. The essence of the modification is to extend formal frame description with the following elements:

a) description of totality of all methods, associated with a frame as a whole;

b) description of frame interface as declaration of a set of properties and methods without detailed description.

Proposed modification allows us to describe both knowledge of data structures, and knowledge about the processes of interaction of these structures with the help of a frame-based model. This means that almost any IS function, based on reusable concepts of SA, can be expressed in the form of interfaces of reusable frames. In addition, proposed modification of the frame-based knowledge model allows setting and solving the problem of automation of synthesis of architecture and the providing part of IS as one-to-one mapping of frame descriptions of SA, IA and PA of the created IS.

2. We developed theoretical-multiple models of structural IS requirements design patterns (6)–(9) and behavioral IS requirements design patterns ((12)–(14), (15)–(17), (18)–(20), (21), (23)–(28)). Models of structural patterns allow formalization of the process of IS architecture design, taking into account reusable components already in the course of formation and analysis of requirements for the created system. These models can be implemented as a specialized data showcase, storing knowledge of previously created IS and individual elements of systems. Models of behavioral patterns (12)–(14), (15)–(17) and (18)-(20) describe operations of addition, modification and deletion of elements of structural IS requirement design patterns at the knowledge level. Models of behavioral patterns (21) and (23)–(28) describe operations on formation of knowledge-oriented description of IS architecture in the form of a network of frames based on knowledge, derived from IS requirements. The obtained results allow us:

a) to unify descriptions of typical operations on structural IS requirements design patterns as a description of operations of changing the scheme of specialized data showcases, storing knowledge of previously created IS and elements of created systems;

b) to establish basic requirements for implementation of these operations as a set of SQL commands or special software components.

References


1. Mansilla, D. A Proposal of a Process Model for Requirements Elicitation in Information Mining Projects [Text] / D. Mansilla, M. Pollo-Cattaneo, P. Britos, R. García-Martínez // Lecture Notes in Business Information Processing. – 2013. – P. 165–173. doi: 10.1007/978-3-642-36611-6_13
2. Berkovich, M. A requirements data model for product service systems [Text] / M. Berkovich, J. M. Leimeister, A. Hoffmann, H. Krcmar // Requirements Engineering. – 2012. – Vol. 19, Issue 2. – P. 161–186. doi: 10.1007/s00766-012-0164-1
3. Lucassen, G. Improving agile requirements: the Quality User Story framework and tool [Text] / G. Lucassen, F. Dalpiaz, J. M. E. M. van der Werf, S. Brinkkemper // Requirements Engineering. – 2016. – Vol. 21, Issue 3. – P. 383–403. doi: 10.1007/s00766-016-0250-x
4. Modelirovanie trebovaniy pol'zovateley [Electronic resource]. – Microsoft Developer Network. – Available at: https://msdn.microsoft.com/ru-ru/library/dd409376.aspx
5. Vilpola, I. H. A method for improving ERP implementation success by the principles and process of user-centred design [Text] / I. H. Vilpola // Enterprise Information Systems. – 2008. – Vol. 2, Issue 1. – P. 47–76. doi: 10.1080/17517570701793848
6. Sutcliffe, A. Scenario-based requirements analysis [Text] / A. Sutcliffe // Requirements Engineering. – 1998. – Vol. 3, Issue 1. – P. 48–65. doi: 10.1007/bf02802920
7. Lipko, Yu. Algoritm formalizacii trebovaniy pri razrabotke informacionnyh sistem [Text] / Yu. Lipko // Izvestiya Yuzhnogo federal'nogo universiteta. Tekhnicheskie nauki. – 2014. – Issue 6 (155). – P. 153–158.
8. Cleland-Huang, J. Mining Domain Knowledge [Requirements] [Text] / J. Cleland-Huang // IEEE Software. – 2015. – Vol. 32, Issue 3. – P. 16–19. doi: 10.1109/ms.2015.67
9. Tyurganov, A. G. Osobennosti formalizacii predmetnoy oblasti korporativnyh informacionnyh sistem [Text] / A. G. Tyurganov // Vestnik Ufimskogo gosudarstvennogo aviacionnogo tekhnicheskogo universiteta. – 2007. – Vol. 9, Issue 5. – P. 72–76.
10. Yue, T. A systematic review of transformation approaches between user requirements and analysis models [Text] / T. Yue, L. C. Briand, Y. Labiche // Requirements Engineering. – 2010. – Vol. 16, Issue 2. – P. 75–99. doi: 10.1007/s00766-010-0111-y
11. Ralph, P. The illusion of requirements in software development [Text] / P. Ralph // Requirements Engineering. – 2012. – Vol. 18, Issue 3. – P. 293–296. doi: 10.1007/s00766-012-0161-4
12. Rational Requisite Pro [Electronic resource]. – IBM developerWorks. – Available at: https://www.ibm.com/developerworks/community/wikis/home?lang=en#!/wiki/Wbcd69e09400c_4f72_9665_66f116225986/page/Rational%20RequisitePro
13. IBM Rational DOORS Next Generation. An efficient requirements management tool for complex systems [Electronic resource]. – IBM. – Available at: http://www-01.ibm.com/common/ssi/cgi-bin/ssialias?subtype=SP&infotype=PM&appname=SWGE_RA_IR_USEN&htmlfid=RAD14128USEN&attachment=RAD14128USEN.PDF
14. Cradle Overview [Electronic resource]. – 3SL. – Available at: https://www.threesl.com/en/cradle/index.php
15. Sistema upravleniya trebovaniyami Devprom Requirements [Electronic resource]. – DEVPROM. – Available at: http://devprom.ru/features/Система-управления-требованиями-Devprom-Requirements
16. Madorskaya, Yu. M. Sistemy upravleniya trebovaniyami: chto i zachem? [Electronic resource] / Yu. M. Madorskaya // ReqCenter.pro. Soglasovannye znaniya dlya prakticheskogo ispol'zovaniya. – Available at: http://edu.reqcenter.pro/?p=2433
17. Luckham, D. The Beginnings of IT Insight: Business Activity Monitoring [Electronic resource] / D. Luckham // Real Time Intelligence & Complex Event Processing. – Available at: http://complexevents.com/media/articles/cep-article-three.pdf
18. Levykin, V. M. Patterny proektirovaniya trebovaniy k informacionnym sistemam: modelirovanie i primenenie [Text]: monografiya / V. M. Levykin, M. V. Evlanov, M. A. Kernosov. – Kharkiv: OOO «Kompaniya «Smit», 2014. – 320 p.
19. Lassila, O. Frames or Objects, or Both? [Text] / O. Lassila // Workshop Notes from the Eight National Conference on Artificial Intelligence (AAAI-90). – Object-Oriented Programming in AI, Boston (Massachusetts, U.S.A.), 1990. – 8 p. – Available at: https://pdfs.semanticscholar.org/c357/adb27b4564751552ff490a04703b39f4620b.pdf

20. Wu, X. A Comparison of Objects with Frames and OODBs [Text] / X. Wu // Object Currents. – 1996. – Vol. 1, Issue 1.
21. Minskiy, M. Freymy dlya predstavleniya znaniy [Text] / M. Minskiy. – Moscow: Energiya, 1979. – 152 p.
22. Iskusstvennyy intellekt. Kn. 2. Modeli i metody [Text]: spravochnik / D. A. Pospelov (Ed.). – Moscow: Radio i svyaz', 1990. – 304 p.
23. Maciaszek, L. A. Requirements Analysis and System Design [Text] / L. A. Maciaszek. – 2nd ed. – Reading: Addison Wesley, Harlow England, 2005. – 504 p.
24. Gavrilov, A. V. Sistemy iskusstvennogo intellekta [Text] / A. V. Gavrilov. – Novosibirsk: NGTU, 2004. – 59 p.
25. Savitch, W. Java: An Introduction to Computer Science and Programming [Text] / W. Savitch. – 2nd ed. – Pearson: Prentice Hall, Inc, 2001. – 1039 p.
26. Deitel, H. M. C++ How to Program [Text] / H. M. Deitel, P. J. Deitel. – 5th ed. – Pearson: Prentice Hall, Inc, 2005 – 1536 p.
27. Levykin, V. M. Issledovanie i razrabotka freymovoy modeli struktury dokumenta [Text] / V. M. Levykin, M. A. Kernosov // Novi tekhnolohyi. – 2008. – Issue 1 (19). – P. 149–154.
28. Evlanov, M. V. Modeli patternov proektirovaniya trebovaniy k informacionnoy sisteme na urovne dannyh [Text] / M. V. Evlanov // Radioelektronni i kompiuterni systemy. – 2014. – Issue 1 (65). – P. 128–138.
29. Levykin, V. M. Parallel'noe proektirovanie informacionnogo i programmnogo kompleksov informacionnoy sistemy [Text] / V. M. Levykin, M. V. Evlanov, V. S. Sugrobov // Radiotekhnika. – 2006. – Issue 146. – P. 89–98.
30. Yevlanov, M. V. Paterny proektuvannia vymoh do informatsiynoi systemy [Text] / M. V. Yevlanov // Visnyk natsionalnoho universytetu «Lvivska politekhnika». – 2014. – Issue 783. – P. 429–434.



***Розроблено алгоритм навчання багатошарового екстрактора ознак, що використовує принципи нейронного газу та розрідженого кодування. Запропоновано інформаційно-екстремальний метод двійкового кодування ознакового подання для побудови вирішальних правил. Це дозволяє зменшити вимоги до обсягів навчальних даних і обчислювальних ресурсів та забезпечити високу достовірність прогнозування порушення умов договору про рівень обслуговування в хмарному середовищі***



***Ключові слова: датацентр, розріджене кодування, нейронний газ, інформаційний критерій, машинне навчання, ройовий алгоритм***



***Разработан алгоритм обучения многослойного экстрактора признаков, использующий принципы нейронного газа и разреженного кодирования. Предложен информационно-экстремальный метод двоичного кодирования признакового представление для построения решающих правил. Это позволяет уменьшить требования к объемам обучающих данных и вычислительных ресурсов и обеспечить высокую достоверность прогнозирования нарушения условий договора об уровне обслуживания в облачной среде***



***Ключевые слова: датацентр, разреженное кодирование, информационный критерий, машинное обучение, роевой алгоритм***

UDC 004:891.032.26:616.127–073.7
DOI: 10.15587/1729-4061.2017.110073


# DEVELOPMENT OF THE METHOD OF FEATURES LEARNING AND TRAINING DECISION RULES FOR THE PREDICTION OF VIOLATION OF SERVICE LEVEL AGREEMENT IN A CLOUD-BASED ENVIRONMENT


**V. Moskalenko**
PhD, Associate Professor*
E-mail: systemscoders@gmail.com
**A. Moskalenko**
Assistant*
E-mail: a.moskalenko@cs.sumdu.edu.ua
**S. Pimonenko**
Postgraduate student*
E-mail: pstsnet@gmail.com
**A. Korobov**
Postgraduate student*
E-mail: artemkorr@gmail.com
*Department of Computer Science
Sumy State University
Rimskoho-Korsakova str., 2, Sumy, Ukraine, 40007


## 1. Introduction

The increase in popularity of cloud-based services stimulates spreading of distributed centers of data processing on the global scale, which leads to numerous problems in terms of resource planning for different administrative domains. Effective resource planning implies simultaneous provision of minimized violation of Service Level Agreement, SLA,