UDC 004.65; 004.91
DOI: 10.15587/1729-4061.2022.269133

DEVELOPMENT OF A METHOD FOR SOLVING THE PROBLEM OF IT PRODUCT CONFIGURATION ANALYSIS

Maksym Ievlanov
Corresponding author
Doctor of Technical Science, Associate Professor*
E-mail: maksym.ievlanov@nure.ua
Nataliya Vasiltcova
PhD, Associate Professor*
Olga Neumyvakina
PhD, Leading Engineer
Educational and Research Department**
Iryna Panforova
PhD, Associate Professor*
*Department of Information Control Systems**
**Kharkiv National University of Radio Electronics
Nauky ave., 14, Kharkiv, Ukraine, 61166

The object of this study is the process of managing the configuration of an IT project.

During the study, the problem of analyzing the configuration of an IT product was solved. Research in this area is mainly aimed at solving the problem of configuration analysis during the refactoring of a monolithic IT product into individual services or microservices. Questions about the methods of decomposing the description of the architecture of the developed IT product into separate functional configuration items (CI) remain almost unexplored.

As a result of the study, a method has been developed that makes it possible to build in the form of a dendrogram all possible options for decomposing the description of the IT product architecture into separate CIs. Unlike existing ones, the proposed method takes into account the degree of repetition of CI descriptions. The method is based on a Smith Maknaoton's divisive algorithm for solving the clustering problem. For its use, when solving the problem, the method for determining the distance between two functional CIs was modified.

The devised method passed an experimental test during the development of the functional task "Formation and maintenance of the individual plan of the scientific and pedagogical employee of the department". As CI, 10 functions of the task were considered. To define these functions, the descriptions of 12 entities in the problem database were used. As a result, a dendrogram was constructed with all possible options for decomposing the description of the task architecture into separate CIs.

Using the results of the study, it is possible to distinguish separate functional CIs and CI groups, the descriptions of which are strongly similar to each other. This allows one to improve the quality of IT product development by assigning such CI groups to one and the same executor of the IT project.

The reported results are used to form backlogs of the IT product and further distribute their elements among the executors of the IT project

Keywords: IT product, architecture description, configuration item, divisive algorithm, Chebyshev distance, Hamming distance

*Received date 05.10.2022
Accepted date 08.12.2022
Published date 30.12.2022*

***How to Cite**: Ievlanov, M., Vasyltsova, N., Neumyvakina, O., Panforova, I. (2022). Development of method for solving the problem of it product configuration analysis. Eastern-European Journal of Enterprise Technologies, 6 (2 (120)), 6–19. doi: https://doi.org/10.15587/1729-4061.2022.269133*

1. Introduction

The current point of view on project management processes singles out the process of managing the configuration of an IT product as one of the processes of integrated control over changes. This process is designed to systematically monitor configuration changes and maintain integrity and trace the configuration throughout the product lifecycle [1].

The main operations of the configuration management process are planning and managing the configuration management process; configuration identification; configuration control; accounting for the state of the configuration; audit of configuration and managing product release and delivery. A special place among these activities is occupied by the work "configuration identification". It defines the elements to be controlled, establishes schemes for identifying elements and their versions, as well as tools and methods that will be used to obtain and manage the selected elements [1].

However, it should be recognized that the definition of the configuration management process formulated in [1] and the specifications of its individual operations are rather arbitrary. Recommendations for the implementation of the configuration management process in the life cycle of an IT product are mainly theoretical and methodological in nature [1]. As one of the reasons for this situation, it is necessary to indicate the uncertainty that arises during the separation of the controlled elements of the IT product. As such an element, it is customary to consider the so-called "configuration item" (CI). A CI may be a set of hardware, software, or both. In this case, each specific CI acts as a unit for configuration management and is considered as a single entity in the configuration management process. At the same time, CI is often considered during the IT product configuration management process as a set of other CIs organized in some way (most often, hierarchically). The consequence of this is the emergence in the course of the work "configuration identification" of a large number of theoretical and applied problems associated with the separation of the optimal number of CI. Optimal here should be understood as such a number of CI that will require minimal time and resources

Copyright © 2022, Authors. This is an open access article under the Creative Commons CC BY license

to perform further work of the IT product configuration management process. Therefore, research in this area should be recognized as promising in both theoretical and applied aspects.

2. Literature review and problem statement

The configuration management process is of particular importance during the design of large systems (the so-called "System of Systems"). Paper [2] shows that most cases of negative impact of local solutions on the overall design and quality of a large software and hardware system were eliminated, in particular, by the early division of the system into separate elements. The reverse cases arose mainly as a result of misinterpretation of the requirements or bias of personal experience [2]. This allows us to conclude that it is advisable to use human-machine or machine methods to identify the configuration of large systems, in which the subjective influence of an individual analyst is minimized.

At the same time, the separation of CI is recognized as an important step in the design of the system based on the results of the analysis of the description of its architecture. In [3], the division of the description of the system architecture into separate microservices is considered. To describe the system-wide architecture, it is proposed to use the domain-specific Silvera language. This language and its compiler allow one, during the design of a software system, to automate the decomposition of the system-wide description of the architecture into descriptions of individual microservices. However, the application of this solution is limited by the following features [3]:

– focusing exclusively on decentralized development of microservices;

– the absence of a description of business logic in the description of the system architecture.

In addition, in [3] the question of the optimal (in terms of the cost of implementing an IT project) number of development teams of the separated microservices is ignored.

Study [4] describes the use of clustering algorithms to solve the problem of microservices separation during refactoring of the source code of a monolithic software application. It is shown that such a way of solving this problem highlights more connected microservices that have fewer intermediate interactions. A similar situation is observed for cases of refactoring of complex multi-level monolithic software systems during their decomposition into microservices [5]. However, such options for separating microservices do not imply a possible dependence of microservices on the functions separated in these software products. It is assumed that the functional decomposition of the description of the architecture of the original software system was carried out during its design and does not change during refactoring.

It is worth noting that the quality of refactoring a monolithic system by dividing its architecture into many microservices can be improved. To do this, in [6] it is proposed to use during decomposition not only a static analysis of the structure of the system but also a dynamic analysis of its actual behavior. However, [6] does not take into account that the general behavior of a system is determined, first of all, by a set of scenarios for the implementation of many separate functional requirements for this system.

In general, studies [4–6] confirm the considerations set forth in [7]. Thus, paper [7] argues that most of the existing approaches to the decomposition of a monolithic architecture into many separate microservices are applicable only under certain conditions. There are probably no universal approaches to solving such a problem. However, such a statement needs additional verification in the case of transferring the solution of the problem to a higher level of abstraction – in the course of separating CIs that implement individual functions of the system.

The issues of separating individual software artifacts that ensure the reliable operation of the designed software system from the description of the meta-architecture of such a system are considered in [8]. It is also noted there that the use of abstractions to describe the architecture of a software system simplifies the solution of the problem of decomposing the description of the system architecture into separate elements. However, the separation from the description of the CI system architecture on a functional basis is not considered in [8].

The considered features of solving the problem of separating individual services or microservices as CI are probably due to the dominant approach to the study of service-oriented systems. In one of the classic works in this area [9], it is noted that solving most of the problems of choosing IT services requires the definition of a set of functionally equivalent IT services before starting to solve such problems. This means that the task of decomposing the description of the system architecture, created on the basis of many functional system requirements, into individual IT services should be solved separately for abstract descriptions of individual services. Such abstract descriptions should not depend on the specifics of the implementation of these services and the non-functional requirements put forward for the services.

A similar approach to the implementation of the functional decomposition of the description of the system architecture into individual elements is shown in [10]. At the same time, the task of configuration analysis is solved in two stages:

– at the first stage, a functional decomposition of the architecture description into separate elements is performed, taking into account the required evolutionary changes;

– at the second stage, the selection of those decomposition options that satisfy the restrictions on the cost of the required changes is carried out.

Based on the results of analysis of works [2–10], the following conclusions were drawn:

a) the objective decomposition of the description of the system architecture into separate CIs is an important task that needs to be addressed in the course of system design;

b) decomposition of the description of the architecture of the designed system directly into individual CIs of a particular supporting system (software services, microservices, etc.) is impossible since it requires taking into account too many factors;

c) it is required to divide this task into two sequentially solved subtasks:

– subtask of the formation of many variants of decomposition of the description of the system architecture into separate functional CIs;

– a subtask of selecting from a plurality of individual functional CIs of such a subset that will satisfy the selection conditions in the best possible way;

d) the features of solving the problem of forming a set of options for decomposing the description of the system architecture into separate functional CI are studied very poorly.

These conclusions allow us to conclude that it is necessary to conduct research into the area of finding such methods for solving the problem of analyzing the configuration of an IT product that can form a set of all possible options for decomposing the description of an IT product architecture into separate CIs.

3. The aim and objectives of the study

The purpose of this study is to develop a method for solving the problem of analyzing the configuration of an IT product. The application of the developed method should ensure the separation of subsets of CI descriptions of the IT product. The condition for such separation is the similarity of these descriptions with each other to a sufficient extent. This makes it possible in the future to form a backlog of the product for teams of IT project executors from CI, whose descriptions are based on similar data structures.

To accomplish the aim, the following tasks have been set:

– to propose a way to determine the distance between the descriptions of the CI architecture of an IT product;

– to modify the divisive clustering algorithm as the basis for devising the developed method;

– to verify the obtained solutions during the analysis of the configuration of a particular IT product.

4. The study materials and methods

The object of this study is the process of managing the configuration of an IT product. The subject of the study is the task of analyzing the configuration of the IT product.

The main hypothesis of this study is the definition of the problem of analyzing the configuration of the IT product as a special case of the clustering problem.

Consequently, the main methods of research will be:

– methods and algorithms for solving the clustering problem;

– methods for determining the distance between the CI of an IT product.

To solve the problem of analyzing the configuration of an IT product, it is proposed to use hierarchical algorithms for solving the clustering problem. This choice is based on the following considerations:

a) as a result of the application of these algorithms, each CI will belong to only one cluster at each specific level of the cluster tree;

b) the application of these algorithms does not impose restrictions on the form of the resulting clusters;

c) the result of the application of these algorithms is a tree of cluster associations, on the basis of which the system architect can choose the final solution taking into account weakly formalized factors.

The first assumption of this study is the separation of divisive algorithms as the preferred algorithms for solving the problem of analyzing the configuration of an IT product. The essence of divisive algorithms is to divide the objects of the source cluster between the two resulting clusters. In general, these algorithms are variations of the following algorithm [11]:

– Step 1. Form one cluster C_1, consisting of all m elements of the original set of clustering objects.

– Step 2. Select an item in cluster C_1 with the highest average distance from other cluster members. The average distance value for cluster element i_p of cluster C_1 can be calculated, for example, as follows:

$$D_{C_1}\left(i_p\right)=\frac{1}{m}\times\sum_{q=1}^{m}d\left(i_p,i_q\right)\forall i_p,i_q\in C_1,p\neq q. \quad (1)$$

Here m is the number of elements in cluster C_1; i_p is the p-th element of cluster C_1; i_q is the q-th element of cluster C_1; $d(i_p, i_q)$ is the distance between elements i_p and i_q.

– Step 3. The item you selected in Step 2 is removed from cluster C_1 and included in the new cluster C_2.

– Step 4. Among the remaining elements of cluster C_1, find one for which the difference between the average distance to the remaining elements of cluster C_1 and the average distance to the elements included in cluster C_2 is positive and maximum.

– Step 5. Remove the item you selected in Step 4 from cluster C_1 and include it in the new cluster C_2.

– Step 6. Continue with Steps 4 and 5 until the mean distance differences are negative, and then complete the execution of the algorithm.

The result of this algorithm is to divide the original cluster C_1 into two child clusters – C_1 and C_2. Next, one of the child clusters is selected and, using this algorithm, is divided into two more child clusters. The partitioning procedure is stopped in one of the following cases [11]:

a) only one member remains in the child cluster;

b) all elements of the child cluster have zero difference from each other.

The second assumption of this study is to describe the CI in the form of sets of entities or classes. Therefore, it is proposed to use methods for determining the distance for objects whose descriptions contain qualitative features. Among these methods, the following are most often distinguished [11]:

– Hamming distance $dH\left(xi,xj\right)=\sum_{l=1}^{n}\left|x_{il}-x_{jl}\right|$;

– distances based on the Gower coefficient;

– distances based on different correlation coefficients (e. g., Pearson, Spearman, or Kendall).

However, these methods do not take into account the peculiarities of the description of the CI of the IT product. Therefore, the use of these methods in its pure form to solve the task is ineffective.

5. Results of the development of a method for solving the problem of analyzing the configuration of an IT product

5. 1. Formation of a way to determine the distance between the descriptions of the configuration items of the IT product architecture

The proposed solution to the problem of analyzing the configuration of an IT product is based on the initial assumption of a unified description of each individual function of this product. Such a unified description implies a clear division of the CI description into the following groups:

a) a group of descriptions of entities or classes that characterize the CI function;

b) a group of descriptions of entities or classes that form input data streams;

c) a group of descriptions of entities or classes that make up the output data streams.

The assumption considered allows us to assert that the descriptions of two CIs should be considered different if they do not coincide in at least one of the above groups. Therefore, to determine the distance between the descriptions of CI, based on this assumption, it is recommended to use the Chebyshev distance. This distance is determined by the formula given in [12]

$$d_\infty\left(i_p,i_q\right)=\max_{1\le l\le m}\left|i_{pl}-i_{ql}\right|, \tag{2}$$

where i_p, i_q are the objects between which the distance is determined; i_{pl} is the l-th coordinate of the object i_p, $1\le l\le m$; i_{ql} is the l-th coordinate of the object i_q, $1\le l\le m$.

For the task of analyzing the configuration of an IT product, the objects i_p, i_q will be the descriptions of the i-th and j-th CI. The coordinates of each specific object will be the groups of function descriptions discussed above, the input and output streams corresponding to CI. Therefore, for the task under consideration, m=3.

However, the Chebyshev distance suggests that coordinate values are numeric for all objects. In the IT Product Configuration Analysis task, the description groups highlighted as coordinates are sets of character strings. In this case, the entities or classes that make up these groups of descriptions are considered as sets, the elements of which are individual attributes (and in the case of classes, methods). It is therefore proposed to introduce a second assumption:

a) the descriptions of each particular entity or class do not change depending on its inclusion in the description of the function, input or output streams of different CIs;

b) each description of an entity or class can be mapped to an identifier.

This assumption allows us to represent groups of descriptions of any functions, input and output flows of CI as sets, the elements of which are the identifiers of the corresponding entities or classes. To determine the distance between such descriptions, it is proposed to modify the definition of the Hamming distance as follows:

$$d_{H_m}\left(i_{pl},i_{ql}\right)=\sum_{k=1}^{n} i_{plk}\oplus i_{qlk}, \tag{3}$$

where i_{plk} is the value of the identifier of the k-th entity or class included in the description of the function, input or output stream i_{pl} CI i_p; i_{qlk} – the value of the identifier of the k-th entity or class included in the description of the function, input or output stream i_{ql} CI i_q; n – the maximum value of the identifier involved in the description of the compared functions, input or output flows CI i_p and i_q; $\oplus$ – operation "sum modulo 2".

The Hamming distance modified in the proposed manner will be equal to 0 in the case when the compared descriptions of functions, input or output streams consist of sets of the same entities or classes. In the case where an entity or class is present in one description and not in another description, the result of the operation from expression (3) will be 1.

Thus, to solve the problem of analyzing the configuration of an IT product, it is proposed to use a method for determining the distance between the descriptions of functions, input and output flows of CI as follows:

$$d_\infty\left(i_p,i_q\right)=\max_{1\le l\le m}\sum_{k=1}^{n} i_{plk}\oplus i_{qlk}. \tag{4}$$

The proposed method for determining the distance makes it possible to determine the degree of proximity of the compared CIs, taking into account their representations in the form of visual models. This allows one, in turn, to develop a method for automated solutions to the problem of analyzing the configuration of an IT product.

5. 2. Modification of the divisive clustering algorithm as the basis for devising the developed method

During the planning of IT projects, the analysis of the configuration of the IT product is carried out strictly after the description of the analyzed configuration is formed. Therefore, the solution of the problem of analyzing the configuration of the IT product is proposed to be considered as a sequential implementation of such stages:

– Step 1. Preparatory stage: selection of sets of analyzed data from the description of the configuration of the IT product, their transformation and coding.

– Step 2. The main stage: solving the problem of analyzing the configuration of the IT product using the divisive algorithm. Completion of the method.

The proposed representation of the solution to the problem assumes the presence of a description of the architecture of the IT product, which formally in the general case can be represented by the expression

$$Arch_{base}=\bigcup_{j=1}^{z} CI_j, \tag{5}$$

where CI_j is a description of the j-th CI of the IT product; z is the number of CIs separated in the description of the IT product. Each description CI_j, in turn, can be formally generally represented as a set

$$CI_j=\left\{W_j,X_j,Y_j\right\}, \tag{6}$$

where W_j is a description of the j-th function of the IT product as a set of entities or classes; X_j – description of all input streams of the j-th function of the IT product as sets of entities or classes that form these flows; Y_j is a description of all output streams of the j-th function of the IT product as sets of entities or classes that form these flows.

The issues of forming a description of the architecture of an IT product from the descriptions of the implemented functional requirements for this product are considered in [13, 14].

Then Step 1 involves performing the following actions:

– Step 1. 1. Generate a set of descriptions of entities or classes based on the structural description of the analyzed architecture of the IT product.

– Step 1. 2. Check for numeric identifiers for each element of the set of entity descriptions or classes formed in Step 1. 1. If the identifier value is not defined for the element of this set to be checked, determine this value yourself.

– Step 1. 3. Select the first of the unconsidered CI_j, which forms a description of the architecture $Arch_{base}$ to be analyzed.

– Step 1. 4. Form a subset W_j by including all entity identifiers or classes that describe the function of the CI_j element.

– Step 1. 5. Generate a subset of X_j by including all entity identifiers or classes that describe each input stream of CI_j.

– Step 1. 6. Generate a subset Y_j by including all entity identifiers or classes that describe each output stream of CI_j.

– Step 1. 7. Repeat Steps 1.3–1.6 until all CI_j that describe the $Arch_{base}$ architecture being analyzed. Complete Step 1.

The result of the implementation of Stage 1 are descriptions of all CIs of the IT product in the form of a set consisting of three subsets. The elements of these subsets are the numeric identifiers of the entities or classes used to structurally describe the architecture of the corresponding *CI*. This makes it possible to use divisive algorithms to solve the problem of analyzing the configuration of an IT product, taking into account those proposed in 5. 1 modification of the method of determining the distance.

Then Step 2 involves performing the following actions:

– Step 2. 1. Generate from the descriptions of CI_j, $j=\overline{1,z}$, the initial cluster C_1 and accept it as the initial cluster C_i, i=1.

– Step 2. 2. For the source cluster, calculate the distances between each pair of elements C_p, C_q using (4).

– Step 2. 3. Determine for each CI_j element of cluster C_i the average distance of Chebyshev according to the formula

$$D_{C_i}\left(CI_i\right)=\frac{1}{m}\times\sum_{p=1}^{m}\max_{1\le l\le 3}\sum_{k=1}^{n}CI_{ilk}\oplus CI_{plk}\forall CI_p\in C_i, j\ne p. \quad (7)$$

– Step 2. 4. Select the CI_p element for which the average distance (7) determined in Step 2. 3 is positive and maximum. If the same maximum distances are defined for two or more CI_p elements, choose from these elements the one whose sum of the Chebyshev distances will be the largest.

– Step 2. 5. Create a child cluster C_{i+1} and include the CI_p element selected in Step 2. 4. Exclude the same item from the original C_i cluster.

– Step 2. 6. Determine, for each CI_j element remaining for consideration in the C_i cluster the average Chebyshev distance $D_{C_i}\left(CI_j\right)$ according to formula (7), taking into account the change in the value of *m* for cluster C_i.

– Step 2. 7. Determine, for each CI_j element remaining for consideration in the C_i cluster the average Chebyshev distance to all elements included in the C_{i+1} child cluster using formula (7), taking into account the *m* value for the C_{i+1} cluster.

– Step 2. 8. For each CI_j element of cluster C_i, calculate the difference in average distances using the formula

$$D\left(CI_j\right)=D_{C_i}\left(CI_j\right)-D_{C_{i+1}}\left(CI_j\right). \quad (8)$$

– Step 2. 9. Select the CI_p element for which difference (8) is positive and maximum. Exclude CI_p from the original C_i cluster and include this element in the C_{i+1} child cluster, and then return to Step 2. 6. If such an element does not exist, create a new child cluster C_{i+2}. In this cluster, include all CI_j elements remaining in the original C_i cluster after you complete Steps 2. 5 through 2. 9.

Step 2. 10. For all child clusters, verify that the algorithm stop conditions are met. If they are not, select the first of the child cluster as the source cluster and go to Step 2. 2. Otherwise, complete Step 2.

The result of the implementation of Stage 2 is a dendrogram of all possible clusters separated for the analyzed description of the architecture of the IT product. Based on this dendrogram, it becomes possible to further analyze and select an option for decomposing the description of the architecture, which will be convenient for the implementation of the corresponding IT project by the existing teams of performers.

5. 3. Verification of the obtained results

Verification of the obtained results was carried out during the planning of the IT project for the development of the functional task "Formation and maintenance of the individual plan of the scientific and pedagogical worker of the department". This task was considered as the development of the information and analytical system "University", operated at the Kharkiv National University of Radio Electronics. Earlier, the functional task "Distribution of the educational load between the teachers of the department" was implemented in the "University" system, the main output document of which is one of the sections of the document "Individual plan of the scientific and pedagogical employee of the department".

The purpose of this IT project was to create a functional task that allows the user to automate the work on the formation and subsequent changes of the document "Individual plan of the scientific and pedagogical worker of the department". At the same time, the number of errors and, consequently, repeated iterations of adjusting the output document should be minimized.

During the planning of this IT project, the following was developed:

a) operational description of the architecture of the functional task «Formation and maintenance of the individual plan of the scientific and pedagogical worker of the department» in the form of a diagram of data flows;

b) structural description of the architecture of the functional task "Formation and maintenance of the individual plan of the scientific and pedagogical worker of the department" in the form of a diagram "essence – connection".

An operational description of the architecture of a functional task is given in Table 1.

Table 1

Structural description of the functional task "Formation and maintenance of the individual plan of the scientific and pedagogical worker of the department" (based on the diagram of data flows)

Operation		Input stream		Output stream	
No.	Name	No.	Name	No.	Name
1	2	3	4	5	6
1	Conversion of the «Educational work» section	1	Teaching load of the lecturer for the academic year	2	Information from the section of the individual plan (IP) «Educational work»
2	Formation of the section «Scientific work»	2	Information about the lecturer	3	Information from the section IP «Scientific work»
		3	Information about the work planned for execution		
		5	Information on recommended works		
		8	Information from the section IP «Scientific work»		
		12	Remaining hours		

Continuation of Table 1

1	2	3	4	5	6
3	Formation of the section «Methodical work»	2	Information about the lecturer	4	Information from the section of IP «Methodical work»
		3	Information about the work planned for execution		
		5	Information on recommended works		
		9	Information from the section of IP «Methodical work»		
		12	Remaining hours		
4	Formation of the section «Organizational work»	2	Information about the lecturer	5	Information from the section of IP «Organizational work»
		3	Information about the work planned for execution		
		5	Information on recommended works		
		10	Information from the section of IP «Organizational work»		
		12	Remaining hours		
5	Formation of a list of positions and long-term assignments	4	Information on positions and long-term assignments	6	Information from the IP section «List of positions and long-term assignments»
		11	Information from the IP section «List of positions and long-term assignments»		
6	Formation of a list of works recommended for implementation	5	Information on recommended works	1	Information on recommended works
7	Formation and maintenance of regulatory and reference information on key performance indicators (KPIs)	6	Information about the key KPIs of the department	7	Information about the key KPIs of the department
8	Formation of KPI of the lecturer and part of the KPI of the department	8	Information from the section IP «Scientific work»	9	Information about the KPI of the lecturer and part from the KPI of the department
9	Generate a pivot table for the academic year	9	Information from the section of IP «Methodical work»	8	Information on the number of hours by IP sections
		7	Information from the section IP «Educational work»		
		8	Information from the section IP «Scientific work»		
		10	Information from the section of IP «Organizational work»		
10	Formation of the output document «IP»	9	Information from the section IP «Methodical work»	10	IP
		7	Information from the section IP «Educational work»		
		8	Information from the section IP «Scientific work»		
		10	Information from the section of IP «Organizational work»		
		11	Information from the IP section «List of positions and long-term assignments»		

Table 1 lists the numbers of works, input and output flows that were generated by the AllFusion Process Modeler CASE tool during the construction of a data flow diagram.

A structural description of the architecture of a functional task is shown in Fig. 1.

During the execution of Stage 1 of the method of solving the problem of analyzing the configuration of the IT product, the following results were obtained.

As a result of performing Steps 1. 1 and 1. 2, a set of descriptions of entities was obtained, given in Table 2.

In Table 2, the numeric identifiers are the entity numbers generated using the AllFusion ERwin Data Modeler CASE tool during development shown in Fig. 1 Entity-Relationship diagram and imported into the Case AllFusion Process Modeler to link the "entity-relationship" diagram to the task data flow diagram.

Table 2

A set of descriptions of functional task entities

Numeric identifier	Name
1	Academic_load
2	Academic
3	Department
4	Individual_plan
5	Academic_section
6	Academic_year
7	Section
8	Recommended_works
9	Type_of_work
10	Section_Pos_Assign_Dept
11	PositionsAssignments
12	KPI

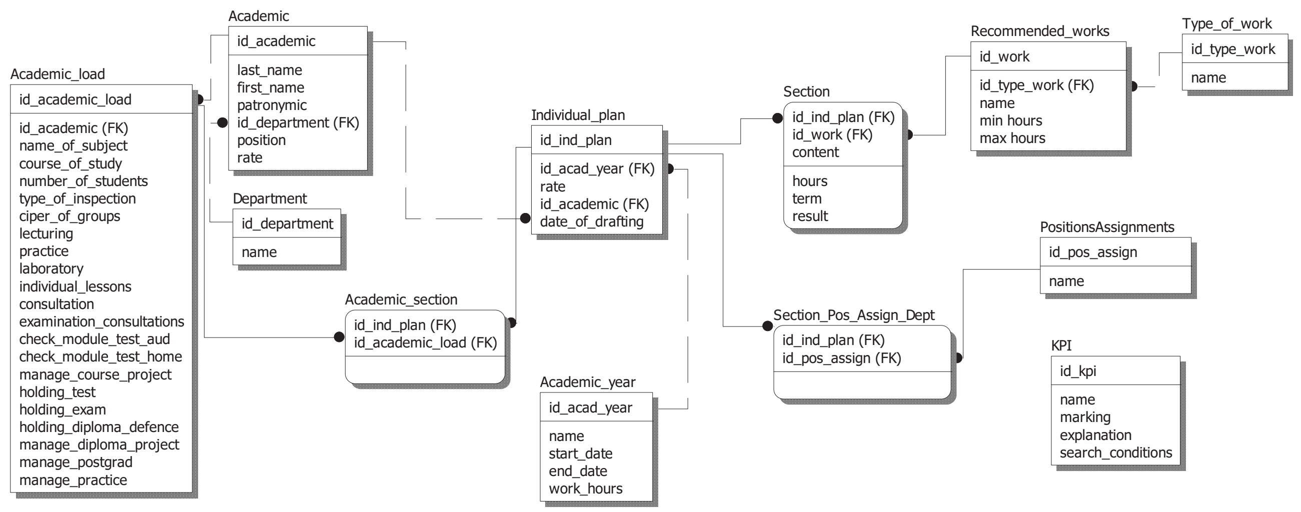

Fig. 1. Structural description of the functional task "Formation and maintenance of the individual plan of the scientific and pedagogical worker of the department" in the form of a diagram "essence – connection"

Basic information for performing Steps 1.3 to 1.7 of the method's Stage 1 is a diagram of the data flows of a functional task, supplemented by an imported diagram "entity – relationship". This addition allows one to assign to each work, input and output flows of the data flow diagram the corresponding subsets of entities from the "entity–relationship" diagram. Therefore, it was proposed to consider as a CI each specific work of the data flow diagram with the accompanying input and output data streams.

An intermediate result of Steps 1. 3–1. 7 is given in Table 3.

Table 3

Initial information for solving a functional problem

No. КЭСІ	Subset identifier	Item No.	Set of entity identifiers that describe an element
1	2	3	4
1	1	1	{1, 2, 3, 4, 5, 6}
	2	1	{1, 2, 3}
	3	2	{1, 2, 4, 5, 6}
2	1	2	{1, 2, 3, 4, 5, 6, 7, 8, 9}
	2	2	{2, 3}
		3	{2, 4, 6, 7, 8, 9}
		5	{8, 9}
		8	{1, 2, 4, 6, 7, 8, 9}
		12	{1, 2, 4, 5, 6, 7}
	3	3	{2, 4, 6, 7, 8, 9}
3	1	3	{1, 2, 3, 4, 5, 6, 7, 8, 9}
	2	2	{2, 3}
		3	{2, 4, 6, 7, 8, 9}
		5	{8, 9}
		9	{2, 4, 6, 7, 8, 9}
		12	{1, 2, 4, 5, 6, 7}
	3	4	{2, 4, 6, 7, 8, 9}
4	1	4	{1, 2, 3, 4, 5, 6, 7, 8, 9}
	2	2	{2, 3}
		3	{2, 4, 6, 7, 8, 9}
		5	{8, 9}
		10	{2, 4, 6, 7, 8, 9}
		12	{1, 2, 4, 5, 6, 7}
	3	5	{2, 4, 6, 7, 8, 9}
5	1	5	{2, 4, 6, 7, 10, 11}
	2	4	{2, 4, 6, 7, 10, 11}
		11	{2, 4, 6, 7, 10, 11}
	3	6	{2, 4, 6, 7, 10, 11}

Continuation of Table 3

1	2	3	4
6	1	6	{8, 9}
	2	5	{8, 9}
	3	1	{8, 9}
7	1	7	{12}
	2	6	{12}
	3	7	{12}
8	1	8	{1, 2, 4, 5, 6, 7, 8, 9, 12}
	2	8	{1, 2, 4, 6, 7, 8, 9}
	3	9	{1, 2, 4, 5, 6, 7, 12}
9	1	9	{1, 2, 4, 5, 6, 7, 8, 9}
	2	7	{1, 2, 4, 5, 6}
		8	{1, 2, 4, 6, 7, 8, 9}
		9	{2, 4, 6, 7, 8, 9}
		10	{2, 4, 6, 7, 8, 9}
	3	8	{1, 2, 4, 5, 6, 7}
10	1	10	{1, 2, 4, 5, 6, 7, 8, 9, 10, 11}
	2	7	{1, 2, 4, 5, 6}
		8	{1, 2, 4, 6, 7, 8, 9}
		9	{1, 4, 6, 7, 8, 9}
		10	{2, 4, 6, 7, 8, 9}
		11	{2, 4, 6, 7, 10, 11}
	3	10	{1, 2, 4, 5, 6, 7, 8, 9, 10, 11}

In Table 3, CI numbers coincide with the work numbers of the data flow diagram specified in Table 1. The identifier of the subset takes the following values: 1 – a subset of the description of the work of the configuration item; 2 is a subset of the description of the input streams of the configuration item; 3 is a subset of the description of the output streams of the configuration item. The contents of the Item Number cells in Table 3 corresponds to work numbers (for subsets 1), input flow numbers (for subset 2), and output stream numbers (for subset 3), taken from Table 1. As identifiers of entities, their numbers from Table 2 are used.

The final result of Steps 1. 3–1. 7 is given in Table 4.

Table 4

Initial information for solving a functional problem

CI No.	CI description
1	{{1, 2, 3, 4, 5, 6}, {1, 2, 3}, {1, 2, 4, 5, 6}}
2	{{1, 2, 3, 4, 5, 6, 7, 8, 9}, {1, 2, 3, 4, 5, 6, 7, 8, 9}, {2, 4, 6, 7, 8, 9}}
3	{{1, 2, 3, 4, 5, 6, 7, 8, 9}, {1, 2, 3, 4, 5, 6, 7, 8, 9}, {2, 4, 6, 7, 8, 9}}
4	{{1, 2, 3, 4, 5, 6, 7, 8, 9}, {1, 2, 3, 4, 5, 6, 7, 8, 9}, {2, 4, 6, 7, 8, 9}}
5	{{2, 4, 6, 7, 10, 11}, {2, 4, 6, 7, 10, 11}, {2, 4, 6, 7, 10, 11}}
6	{{8, 9}, {8, 9}, {8, 9}}
7	{{12}, {12}, {12}}
8	{{1, 2, 4, 5, 6, 7, 8, 9, 12}, {1, 2, 4, 6, 7, 8, 9}, {1, 2, 4, 5, 6, 7, 12}}
9	{{1, 2, 4, 5, 6, 7, 8, 9}, {1, 2, 4, 5, 6, 7, 8, 9}, {1, 2, 4, 5, 6, 7}}
10	{{1, 2, 4, 5, 6, 7, 8, 9, 10, 11}, {1, 2, 4, 5, 6, 7, 8, 9, 10, 11}, {1, 2, 4, 5, 6, 7, 8, 9, 10, 11}}

Consider the first pass of Stage 2 of the proposed method for solving the problem of analyzing the configuration of an IT product. As a result of performing Step 2. 1 and Step 2, we obtained the Chebyshev distance values for each pair of items of the initial C_1 cluster. The values are given in Table 5.

Table 5

Values of Chebyshev distances for each pair of items in cluster C_1

–	CI_1	CI_2	CI_3	CI_4	CI_5	CI_6	CI_7	CI_8	CI_9	CI_{10}
CI_1	0	6	6	6	7	8	7	6	7	9
CI_2	6	0	0	0	7	7	10	4	3	4
CI_3	6	0	0	0	7	7	10	4	3	4
CI_5	6	0	0	0	7	7	10	4	3	4
CI_5	7	7	7	7	0	8	7	7	6	4
CI_6	8	7	7	7	8	0	3	7	7	9
CI_7	7	10	10	10	7	3	0	8	9	11
CI_8	6	4	4	4	7	7	8	0	2	4
CI_9	7	3	3	3	6	7	9	2	0	3
CI_{10}	9	4	4	4	4	9	11	4	3	0

As a result of performing Step 2. 3, the average Chebyshev distance was determined for each element. The values of these distances are given in Table 6.

Table 6

Values of the average Chebyshev distances for each element of the cluster

CI	CI_1	CI_2	CI_3	CI_4	CI_5	CI_6	CI_7	CI_8	CI_9	CI_{10}
Average distance	6.2	4.1	4.1	4.1	6	6.3	7.5	4.6	4.3	5.2

As a result of performing Step 2. 4, CI_7 was selected. This element was excluded from the original C_1 cluster as a result of Step 2. 5 and included in the formed child cluster C_2.

As a result of performing Steps 2 .6–2. 8, new average Chebyshev distances were determined for each element remaining for consideration in cluster C_1, and the difference in these distances was calculated. The results of the calculations are given in Table 7.

During Step 2. 9, CI CI_6 was moved to the child cluster C_2. After that, for CIs CI_1, CI_2, CI_3, CI_4, CI_5, CI_8, CI_9 and remaining in the C_1 cluster, the Steps 2. 6–2. 8 calculations were repeated. The results of these calculations are given in Table 8.

Table 7

Values of the difference in the average Chebyshev distances for each element of the cluster C_1

Chebyshev distance										Average to items C_1	Average to items C_2	Difference
–	CI_1	CI_2	CI_3	CI_4	CI_5	CI_6	CI_8	CI_9	CI_{10}			
CI_1	0	6	6	6	7	8	6	7	9	6.11	7	–0.89
CI_2	6	0	0	0	7	7	4	3	4	3.44	10	–6.56
CI_3	6	0	0	0	7	7	4	3	4	3.44	10	–6.56
CI_5	6	0	0	0	7	7	4	3	4	3.44	10	–6.56
CI_5	7	7	7	7	0	8	7	6	4	5.89	7	–1.11
CI_6	8	7	7	7	8	0	7	7	9	6.67	3	3.67
CI_8	6	4	4	4	7	7	0	2	4	4.22	8	–3.78
CI_9	7	3	3	3	6	7	2	0	3	3.78	9	–5.22
CI_{10}	9	4	4	4	4	9	4	3	0	4.56	11	–6.44

Table 8

Values of the difference in the average Chebyshev distances for each element remaining in cluster C_1

Chebyshev distance									Average to items C_1	Average to items C_2	Difference
–	CI_1	CI_2	CI_3	CI_4	I_5	CI_8	CI_9	CI_{10}			
CI_1	0	6	6	6	7	6	7	9	5.875	7.5	–1.625
CI_2	6	0	0	0	7	4	3	4	3	8.5	–5.5
CI_3	6	0	0	0	7	4	3	4	3	8.5	–5.5
CI_5	6	0	0	0	7	4	3	4	3	8.5	–5.5
CI_5	7	7	7	7	0	7	6	4	5.625	7.5	–1.875
CI_8	6	4	4	4	7	0	2	4	3.875	7.5	–3.625
CI_9	7	3	3	3	6	2	0	3	3.375	8	–4.625
CI_{10}	9	4	4	4	4	4	3	0	4	10	–6

Since all the differences in Table 8 are negative, then a new child cluster C_3 was formed. Items CI_1, CI_2, CI_3, CI_4, CI_5, CI_8, CI_9 and CI_{10} were included in this cluster.

Step 2. 10 found that the stop conditions for clusters C_2 and C_3 are not met. Each of these clusters is not a singleton and the distances between the elements of each of these clusters are not 0. At this point, the first pass of Phase 2 of the method was completed. For the second pass of Step 2 of the method, cluster C_2 is proposed.

As a result of the implementation of Stage 2 of the method for solving the problem of analyzing the configuration of an IT product, a dendrogram of clusters was constructed, shown in Fig. 2.

This dendrogram is the result of solving the problem of analyzing the configuration of an IT product. In the future, it can be used to assign CIs as task list items for teams of performers of an IT project to develop a functional task.

Consider the solution of the same problem in the way set forth in [10]. In accordance with this method, it is proposed to convert the description of the system under study to a set of the following descriptions:

– descriptions of data structures that define state variables;

– descriptions of the operations that are performed on state variables.

At the same time, the interaction of operations and state variables is divided in [10] into two main groups:

– reading a state variable by an operation;

– the operation's writing of values to a state variable.

Further, in [10], it is proposed to convert the set of these descriptions into an undirected graph. The vertices of this graph represent state variables and operations, and the arcs represent the interactions of operations and state variables. In this case, each arc has one of the following weights:

– a weight of 1 if a variable state operation reads;

– a weight of 2 if the operation writes values to a state variable.

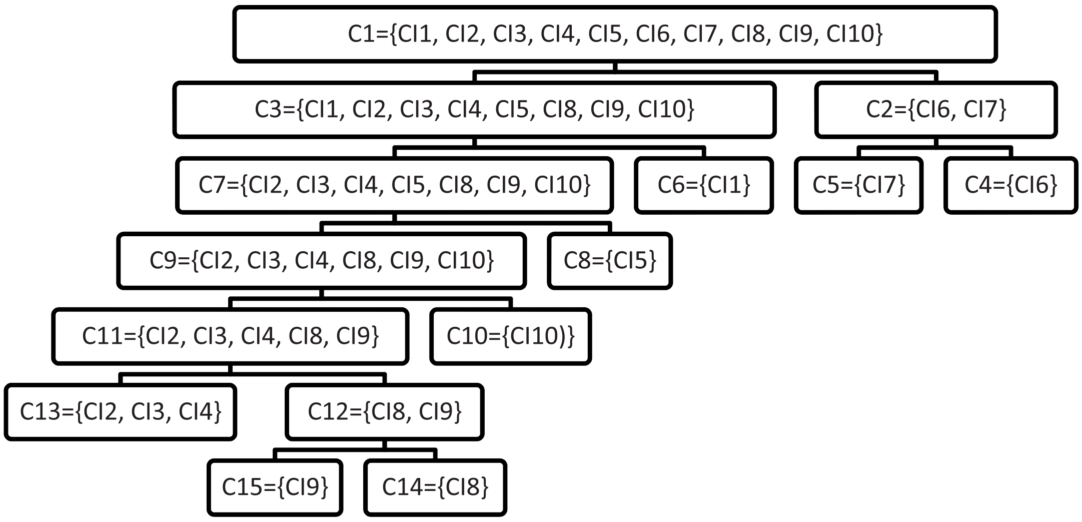
C1={CI1, CI2, CI3, CI4, CI5, CI6, CI7, CI8, CI9, CI10}
C3={CI1, CI2, CI3, CI4, CI5, CI8, CI9, CI10}
C2={CI6, CI7}
C7={CI2, CI3, CI4, CI5, CI8, CI9, CI10}
C6={CI1}
C5={CI7}
C4={CI6}
C9={CI2, CI3, CI4, CI8, CI9, CI10}
C8={CI5}
C11={CI2, CI3, CI4, CI8, CI9}
C10={CI10)}
C13={CI2, CI3, CI4}
C12={CI8, CI9}
C15={CI9}
C14={CI8}

Fig. 2. Dendrogram of clusters of configuration items of the functional task "Formation and maintenance of the individual plan of the scientific and pedagogical worker of the department"

System decomposition is carried out on the constructed graph by selecting separate clusters. Each such cluster is a part of a graph connected to other parts by the smallest possible number of arcs with minimal weights. The rationale for this method of cluster separation is detailed in [10].

It should be noted that the method described in [10] is focused on the use of UML diagrams describing the software system being created. In our case, the description of the functional task is DFD. Therefore, as descriptions of operations, we will use descriptions of works from Table 1. As descriptions of state variables, we will use the descriptions of input and output streams from Table 1. In this case, we will assume the following assumption: the output streams of any work from Table 1 are the input streams of other works from Table 1 if the names of these streams are the same. The results of the separation of operations and variables of the state of the functional task «Formation and maintenance of the individual plan of the scientific and pedagogical worker of the department» are given in Table 9.

Table 9

Description of operations and variables of the state of the functional task "Formation and maintenance of the individual plan of the scientific and pedagogical worker of the department" (based on the diagram of data flows)

Designation	Name
Operation	
O1	Conversion of the «Educational work» section
O2	Formation of the section «Scientific work»
O3	Formation of the section «Methodical work»
O4	Formation of the section «Organizational work»
O5	Formation of a list of positions and long-term assignments
O6	Formation of a list of works recommended for implementation
O7	Formation and maintenance of regulatory and reference information on key performance indicators (KPIs)
O8	Formation and maintenance of regulatory and reference information on key performance indicators (KPIs)
O9	Generate a pivot table for the academic year
O10	Formation of the output document «IP»
State variables	
V1	Teaching load of the lecturer for the academic year
V2	Information about the lecturer
V3	Information about the work planned for execution
V4	Information on positions and long-term assignments
V5	Information on recommended works
V6	Information about the key KPIs of the department
V7	Information from the section IP «Educational work»
V8	Information from the section IP «Scientific work»
V9	Information from the section of IP «Methodical work»
V10	Information from the section of IP «Organizational work»
V11	Information from the IP section «List of positions and long-term assignments»
V12	Remaining hours
V13	Information on the number of hours by IP sections
V14	Information about the KPI of the lecturer and part from the KPI of the department
V15	IP

A description of the interactions between operations and state variables is given in Table 10.

Table 10

Description of interactions between operations and state variables

Operation	State variables														
	V1	V2	V3	V4	V5	V6	V7	V8	V9	V10	V11	V12	V13	V14	V15
O1	r	–	–	–	–	–	w	–	–	–	–	–	–	–	–
O2	–	r	r	–	r	–	–	r, w	–	–	–	r	–	–	–
O3	–	r	r	–	r	–	–	–	r, w	–	–	r	–	–	–
O4	–	r	r	–	r	–	–	–	–	r, w	–	r	–	–	–
O5	–	–	–	r		–	–	–	–	–	r, w	–	–	–	–
O6	–	–	–	–	r, w	–	–	–	–	–	–	–	–	–	–
O7	–	–	–	–	–	r, w	–	–	–	–	–	–	–	–	–
O8	–	–	–	–	–	–	–	r	–	–	–	–	–	w	–
O9	–	–	–	–	–	–	r	r	r	r	–	–	w	–	–
O10	–	–	–	–	–	–	r	r	r	r	r	–	–	–	w

The result of the graph construction according to the method described in [10] is shown in Fig. 3. Each edge has its weight indicated. This assumes that the weight of the edge, which describes both reading and writing, is 3 ((r=1)+(w=2)=3).

Most of the vertices of the constructed graph are strongly related to each other. This means that the studied description of the architecture of a functional task is strongly monolithic. Therefore, during the selection of clusters on the constructed graph, the following will be formed:

– relatively small clusters, including one operation;

– one large cluster that includes a large number of operations that will have to be implemented in the form of a monolithic software module.

Solutions for separating clusters on the constructed graph (Fig. 3) are given in Table 11.

The method described in [10] requires minimizing the number of arcs to be cut. However, the fulfillment of this condition (Solution 1 in Table 11) leads to the issuance of a task for the creation of a large monolithic software module (C2 cluster). Such a solution is ineffective for further division of work between the teams of IT project executors.

[10] does not specify the condition for stopping the solution of the clustering problem on the graph description of the system architecture. Therefore, in Table 11 are those separation options for which the number of cut arcs of the graph does not exceed 4.

Comparison of the results of solving the problem of analyzing the configuration of the IT product on the example of the functional task «Formation and maintenance of the individual plan of the scientific and pedagogical worker of the department» (Fig. 2 and Table 11) allows us to draw the following conclusions:

a) when separating clusters consisting of one CI, both the method developed, and the method described in [10] give the same results;

b) when separating clusters consisting of several CIs, the method described in [10] is less accurate than the developed method;

c) if the number of connections between CIs that need to be cut when decomposing the graph into separate clusters increases, the developed method and the method described in [10] give very similar results (Solution 7 in Table 11 is completely the same as clusters C3, C4 and C5, highlighted in Fig. 2).

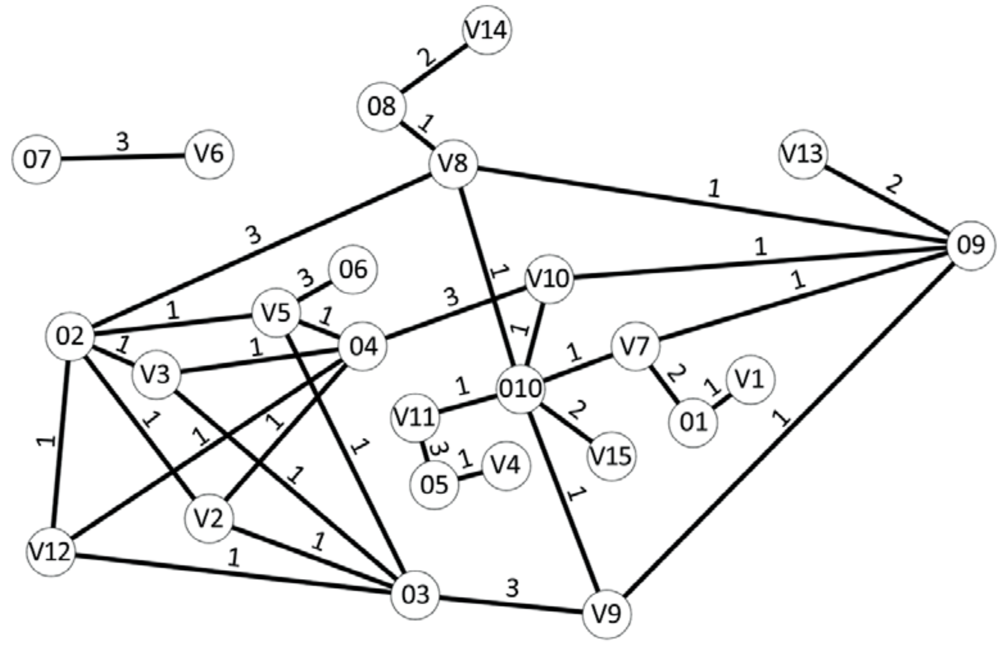

Fig. 3. Type of graph describing the interaction of operations and variables of the state of the functional task "Formation and maintenance of the individual plan of the scientific and pedagogical worker of the department"

Table 11

Description of the results of solving the problem of analyzing the configuration of an IT product in the manner set forth in [10]

Cluster designation	Designations of the vertices that are part of the cluster	The CI symbols of the functional task corresponding to the operations	Designation of a similar cluster on the dendrogram
1	2	3	4
Solution 1 (number of arcs of the minimum weight cut – 0)			
C1	O7, V6	CI7	C5
C2	O1, O2, O3, O4, O5, O6, O8, O9, O10, V1, V2, V3, V4, V5, V7, V8, V9, V10, V11, V12, V13, V14, V15	CI1, CI2, CI3, CI4, CI5, CI6, CI8, CI9, CI10	There is no complete analog, the closest is C3
Solution 2 (number of arcs of minimum weight cut – 1)			
C1	O7, П6 V6	CI7	C5
C2	O8, П14 V14	CI8	C14
C3	O1, O2, O3, O4, O5, O6, O9, O10, V1, V2, V3, V4, V5, V7, V8, V9, V10, V11, V12, V13, V15	CI1, CI2, CI3, CI4, CI5, CI6, CI9, CI10	There is no complete analog, the closest is C3
Solution 3 (number of arcs of minimum weight cut – 1)			
C1	O7, V6	CI7	C5
C2	O5, V4, V11	CI5	C8
C3	O1, O2, O3, O4, O6, O8, O9, O10, V1, V2, V3, V5, V7, V8, V9, V10, V12, V13, V14, V15	CI1, CI2, CI3, CI4, CI6, CI8, CI9, CI10	There is no complete analog, the closest is C3
Solution 4 (number of arcs of minimum weight cut – 2)			
C1	O7, V6	CI7	C5
C2	O8, V14	CI8	C14
C3	O5, П4, V4, V11	CI5	C8
C4	O1, O2, O3, O4, O6, O9, O10, V1, V2, V3, V5, V7, V8, V9, V10, V12, V13, V15	CI1, CI2, CI3, CI4, CI6, CI9, CI10	There is no complete analog, the closest are C3, C7 and C9
Solution 5 (number of arcs of minimum weight cut – 2)			
C1	O7, П6 V6	CI7	C5
C2	O1, П1, П7 V1, V7	CI1	C6
C3	O2, O3, O4, O5, O6, O8, O9, O10, П2, П3, П4, П5, П8, П9, П10, П11, П12, П13, П14, П15 V2, V3, V4, V5, V8, V9, V10, V11, V12, V13, V14, V15	CI2, CI3, CI4, CI5, CI6, CI8, CI9, CI10	There is no complete analog, the closest is C7

Continuation of Table 12

1	2	3	4
Solution 6 (number of arcs of minimum weight cut – 3)			
C1	O7, V6	CI7	C5
C2	O8, V14	CI8	C14
C3	O1, V1, V7	CI1	C6
C4	O2, O3, O4, O5, O6, O9, O10, V2, V3, V4, V5, V8, V9, V10, V11, V12, V13, V15	CI2, CI3, CI4, CI5, CI6, CI9, CI10	There is no complete analog, the closest is C7
Solution 7 (number of arcs of minimum weight cut – 3)			
C1	O7, V6	CI7	C5
C2	O5, V4, V11	CI5	C8
C3	O1, V1, V7	CI1	C6
C4	O2, O3, O4, O6, O8, O9, O10, V2, V3, V5, V8, V9, V10, V12, V13, V14, V15	CI2, CI3, CI4, CI6, CI8, CI9, CI10	There is no complete analog, the closest is C9
Solution 8 (number of arcs of minimum weight cut – 4)			
C1	O7, V6	CI7	C5
C2	O8, V14	CI8	C14
C3	O5, V4, V11	CI5	C8
C4	O1, П1, П7 V1, V7	CI1	C6
C5	O2, O3, O4, O6, O9, O10, V2, V3, V5, V8, V9, V10, V12, V13, V15	CI2, CI3, CI4, CI6, CI9, CI10	There is no complete analog, the closest is C9
Solution 9 (number of arcs of minimum weight cut – 4)			
C1	O7, V6	CI7	C5
C2	O6, V5	CI6	C4
C3	O1, O2, O3, O4, O5, O8, O9, O10, V1, V2, V3, V4, V7, V8, V9, V10, V11, V12, V13, V14, V15	CI1, CI2, CI3, CI4, CI5, CI8, CI9, CI10	C3

It should be noted that the increase in the number of connections between CIs, which must be cut when decomposing the graph into separate clusters, leads to an increase in problems in the further complexion of the system from individual CIs. Therefore, it is necessary to recognize the developed method for solving the problem of analyzing the configuration of the IT product more accurate and more convenient for use during the initial planning of an IT project. If it is necessary to replan the ongoing IT project due to making changes, the method outlined in [10] may be more convenient.

6. Discussion of results of the development of a method for solving the problem of analyzing the configuration of an IT product

In the course of the study, a method was developed to determine the distance between the descriptions of the CI architecture of the IT product (4). This method is the Chebyshev distance, in which, instead of the difference in individual coordinates of objects, the Hamming distance modified by us is used. This solution is explained by common descriptions of functional CI as a process for converting input streams into outputs. In this case, the descriptions of the input and output streams, as well as functions, are represented as a set of database entities or software classes of these CIs.

Further, taking into account the developed method of determining the distance, the divisive clustering algorithm was modified as the basis for creating the method being developed. The proposed modification allows us to identify all possible options for dividing the description of the architecture of the IT product into groups of individual CIs. This decision is explained by the lack of satisfactory formal criteria for selecting the most effective option for decomposing the description of the IT product architecture into separate CIs.

Our results were verified in the course of solving the problem of analyzing the configuration of the functional task "Formation and maintenance of the individual plan of the scientific and pedagogical worker of the department". During verification, a dendrogram was formed (Fig. 2), which shows all possible options for dividing the description of the task architecture into separate functional CIs. The presence in the dendrogram of the daughter cluster C_{13} with three CIs (CI_2, CI_3 and C_4) is explained by the fact that the descriptions of the CI data completely coincide with each other.

In contrast to the method proposed in [8] for separating individual software artifacts, the developed method allows us to consider as CI precisely the descriptions of individual functions. This feature allows one to take into account in the course of solving the problem of analyzing the configuration of the IT product not only technical but also business features of the analyzed product. A similar solution was proposed in [10]. However, unlike that solution, the decomposition of the description of the product architecture took into account the measure of the proximity of their descriptions in the form of a set of data structures. The use of such a measure allows one to improve the quality of an IT project by assigning to the executor similar in their description CIs.

The construction of the developed method on the basis of the Smith Maknaoton divisive algorithm allows us to distinguish a set of all possible options for decomposing the description of the architecture of the IT product into separate CIs. This set is presented to the analyst in the form of a dendrogram. Based on this dendrogram, the analyst further solves the problem of selecting for the executors of the IT project such subsets of functional CIs that will satisfy the basic (in terms of quality, content, duration, and cost) limitations of the IT project.

The following limitations of this study should be recognized:

– using as descriptions of the architecture of the developed functional task its data flow diagram and the «entity-relationship» diagram;

– the use of a divisive algorithm for solving the clustering problem, which leads to the need to identify requirements for all functions of the IT product before forming a description of the architecture of this product.

The following disadvantages of this study should be recognized:

– the need to construct a data flow diagram with a detailed description of the data structures for each data flow and each work of the diagram;

– the need to build at least at the conceptual level a diagram "essence – connection" for the analyzed IT product;

– the numerical coding method used in the experiment does not take into account such an aspect as the semantic proximity of the encoded names of functions and input and output data streams.

Further development of this study can be carried out in several directions. Thus, the question of the applicability of the proposed method for processing descriptions of the architecture of an IT product made using object-oriented visual models remains open and requires further research. More promising, although more complex, seem to be research aimed at forming and tracing chains of concepts with similar semantics. The expected results of such studies will greatly simplify the search, selection, and reuse of code fragments and database elements that ensure the implementation of individual concepts of the subject area.

The question of the expediency of separating the task of distributing the results of the analysis of the configuration of the IT product among the teams of IT project executors into a separate task also requires further research. The possibility of a combined solution that seeks the optimal design-cost partitioning of the architecture description into CI clusters (e. g., using various modifications of the fuzzy *c*-means algorithm) also remains unclear. The main difficulty of this solution is to build an objective function that could take into account the main project constraints that affect the choice of the number of performers of the IT project and the way individual performers are combined into teams.

7. Conclusions

1. A method for determining the distance between the descriptions of the CI architecture of an IT product is proposed. The essence of this technique is to use the distance of the modified Chebyshev distance. In the proposed modification, a modified Hamming distance is used to determine the distance between the individual elements of the compared functions. The proposed method for determining the distance allows processing data from visual models of descriptions of the architecture of the IT product and individual CIs.

This greatly simplifies the further implementation of the proposed method within the framework of specialized information technologies for design automation and design management of IT products.

2. We have modified a divisive clustering algorithm as the basis for creating the developed method. The essence of the modification is to use the modified Chebyshev distance to determine the distances between individual CIs in the course of solving the clustering problem. This modification allows us to form all possible options for describing the architecture of the developed IT product into separate CIs and their sets. Such a representation of the solution to the problem of analyzing the configuration of the IT product allows one to further automate the solution of the problem of distributing the separated CIs and their subsets between the teams of IT project executors.

3. An experimental verification of the results obtained was carried out. This check was carried out during the planning of the IT-project of the development of the functional task "Formation and maintenance of the individual plan of the scientific and pedagogical worker of the department". As a description of the architecture, such visual models of a functional task as a diagram of data flows and an entity-relationship diagram were used. The results of the test allow us to assert that the solution to the problem of analyzing the configuration of the IT product can be performed both manually and in an automated way. In addition, the application of the developed method makes it possible to isolate during the clustering of the description of the architecture of the IT product not only individual CIs but also sets of CIs, the descriptions of which are strongly similar to each other. The separation of such sets and their subsequent assignment to the same executor of the IT project allows one to improve the quality of an IT product by repeatedly using the same solutions to implement similar CIs.

Conflicts of interest

The authors declare that they have no conflict of interest in relation to this research, whether financial, personal, authorship or otherwise, that could affect the research and its results presented in this paper.

Financing

The study was conducted without financial support.

Data availability

The data will be provided upon reasonable request.

Acknowledgments

The authors of this paper thank a student of the UPGITm-20-1 group at the Kharkiv National University of Electronics (Ukraine) Elizaveta Kuzma for the help in conducting practical testing of the results of the study.

References

1. Bourque, P., Fairley, R. E. (Eds.) (2014). Guide to the Software Engineering Body of Knowledge. Version 3.0. IEEE Computer Society, 335.
2. Cadavid, H., Andrikopoulos, V., Avgeriou, P., Broekema, P. C. (2022). System and software architecting harmonization practices in ultra-large-scale systems of systems: A confirmatory case study. Information and Software Technology, 150, 106984. doi: https://doi.org/10.1016/j.infsof.2022.106984
3. Suljkanović, A., Milosavljević, B., Inđić, V., Dejanović, I. (2022). Developing Microservice-Based Applications Using the Silvera Domain-Specific Language. Applied Sciences, 12 (13), 6679. doi: https://doi.org/10.3390/app12136679
4. Sellami, K., Saied, M. A., Ouni, A. (2022). A Hierarchical DBSCAN Method for Extracting Microservices from Monolithic Applications. The International Conference on Evaluation and Assessment in Software Engineering 2022, 201–210. doi: https://doi.org/10.1145/3530019.3530040
5. Krause, A., Zirkelbach, C., Hasselbring, W., Lenga, S., Kroger, D. (2020). Microservice Decomposition via Static and Dynamic Analysis of the Monolith. 2020 IEEE International Conference on Software Architecture Companion (ICSA-C). doi: https://doi.org/10.1109/icsa-c50368.2020.00011
6. Matias, T., Correia, F. F., Fritzsch, J., Bogner, J., Ferreira, H. S., Restivo, A. (2020). Determining Microservice Boundaries: A Case Study Using Static and Dynamic Software Analysis. 14th European Conference on Software Architecture, ECSA 2020, 315–332. doi: https://doi.org/10.1007/978-3-030-58923-3_21
7. Fritzsch, J., Bogner, J., Zimmermann, A., Wagner, S. (2019). From monolith to microservices: A classification of refactoring approaches. 1st International Workshop on Software Engineering Aspects of Continuous Development and New Paradigms of Software Production and Deployment, DEVOPS 2018, 128–141. doi: https://doi.org/10.1007/978-3-030-06019-0_10
8. Shahin, R. (2021). Towards Assurance-Driven Architectural Decomposition of Software Systems. 40th International Conference on Computer Safety, Reliability and Security, SAFECOMP 2021 held in conjunction with Workshops on DECSoS, MAPSOD, DepDevOps, USDAI and WAISE 2021, 187–196. doi: https://doi.org/10.48550/arXiv.2106.09237
9. Reiff-Marganiec, S., Tilly, M. (Eds.) (2012). Handbook of Research on Service-Oriented Systems and Non-Functional Properties: Future Directions. Hershey: IGI Global, 521. doi: https://doi.org/10.4018/978-1-61350-432-1
10. Faitelson, D., Heinrich, R., Tyszberowicz, S. (2017). Supporting Software Architecture Evolution by Functional Decomposition. Proceedings of the 5th International Conference on Model-Driven Engineering and Software Development, 435–442. doi: https://doi.org/10.5220/0006206204350442
11. Wierzchoń, S., Kłopotek, M. (2018). Modern Algorithms of Cluster Analysis. Cham: Springer, 441. doi: https://doi.org/10.1007/978-3-319-69308-8
12. Barsegian, A. A., Kupriianov, M. S., Kholod, I. I., Tess, M. D., Elizarov, S. I. (2009). Analiz dannykh i protcessov. Saint Petersburg: BKhV-Peterburg, 512.
13. Yevlanov, M. V., Vasyltsova, N. V., Panforova, I. Yu. (2015). Modeli i metody syntezu opysu ratsionalnoi arkhitektury informatsiinoi systemy. Visnyk naukovoho universytetu «Lvivska politekhnika». Seriia «Informatsiini systemy ta merezhi», 829, 135–152. Available at: https://science.lpnu.ua/sites/default/files/journal-paper/2018/jun/12881/9ievlanovmvvasilcovanv.pdf
14. Evlanov, M. V. (2016). Development of the model and method of selecting the description of rational architecture of information system. Eastern-European Journal of Enterprise Technologies, 1 (2 (79)), 4–12. doi: https://doi.org/10.15587/1729-4061.2016.60583