Formal Data Integration Models Development for Intelligent Electronic Commerce Systems

Victoria Vysotska[1], Andrii Berko[1], Lyubomyr Chyrun[2], Sofia Chyrun[1], Olena Havrylyshyn[3], Oksana Smirnova[1], Nataliia Sokulska[4], Olena Sokhatska[5] and Iryna Shakleina[1]

[1] *Lviv Polytechnic National University, Stepan Bandera Street, 12, Lviv, 79013, Ukraine*
[2] *Ivan Franko National University of Lviv, University Street, 1, Lviv, 79000, Ukraine*
[3] *Ukrainian Academy of Printing, Pidholosko St., 19, Lviv, 79020, Ukraine*
[4] *Hetman Petro Sahaidachnyi National Army Academy, Heroes of Maidan street, 32, Lviv, 79026, Ukraine*
[5] *West Ukrainian National University, Lvivska Street, 11, Ternopil, 46004, Ukraine*

Abstract
The problem of creation and application of methods and means of information technologies of electronic commerce for various subject areas and applications has been studied, as the problems of developing mathematical models, solution methods and instrumental means for the integration of information resources and the functioning of intelligent electronic commerce systems with the use of effective intelligent models have been solved. Processes of modelling and design of business analytics tools for processing heterogeneous information resources based on ontologies are described. To solve the problem, several scientific tasks were performed, in particular, a classification of intelligent electronic commerce systems and means of processing heterogeneous distributed information resources of business analytics was proposed, a formal model of intelligent electronic commerce systems is developed using ontologies, its components, a structural model of information resources, methods and algorithms for designing intelligent electronic commerce systems based on the apparatus of ontologies and integration of information resources.

Keywords
Intelligent system, electronic commerce, Data Integration, system model, process model

1. Introduction

The processes of data integration have a fairly wide scope of practical applications. In particular, in areas such as construction of DS of various types and directions, development of corporate management systems, information Web systems, electronic business systems, computer monitoring, etc. The information resources of such systems provide for the simultaneous use of a significant number of various forms, structures, content, methods of presentation and application of data [1-2]. The purpose of developing the method of multi-level data integration is to build and justify a single generalized approach to solving the given task and determining ways of implementation that will ensure its interoperability and invariance to the nature, content, specificity, and order of application of the integrated data. This is especially important in operational integration processes, in which these data properties are often not predetermined and may change during the integration procedures themselves. The basis for solving the problems of this section is the formal presentation of data as a system, the syntax, structure and

COLINS-2024: 8th International Conference on Computational Linguistics and Intelligent Systems, April 12–13, 2024, Lviv, Ukraine

victoria.a.vysotska@lpnu.ua (V. Vysotska); andrii.y.berko@lpnu.ua (A. Berko); Lyubomyr.Chyrun@lnu.edu.ua (L. Chyrun); sofiia.chyrun.sa.2022@lpnu.ua (S. Chyrun); havrylyshynolena@gmail.com (O. Havrylyshyn); oksana.y.smirnova@lpnu.ua (O. Smirnova); natalya.sokulska@gmail.com (N. Sokulska); o.sokhatska@wunu.edu.ua (O. Sokhatska); iryna.o.shakleina@lpnu.ua (I. Shakleina)

0000-0001-6417-3689 (V. Vysotska); 0000-0001-8653-1520 (A. Berko); 0000-0003-3140-3788 0000-0002-9448-1751 (L. Chyrun); 0000-0002-2829-0164 (S. Chyrun); 0000-0001-7181-4421 (O. Havrylyshyn); 0000-0002-1314-0489 (O. Smirnova); 0000-0002-3425-5517 (N. Sokulska); 0000-0002-6535-549X (O. Sokhatska); 0000-0003-0809-1480 (I. Shakleina)

semantics of which elements are described with the help of special tools suitable for software perception and processing. The main task of data integration is the formation of a complete and consistent output set based on a set of disparate input data obtained from various sources. To achieve the final goal of integration, it is necessary to ensure a coordinated combination in the single formation of their syntax, structure and semantics [3-4]. In the course of solving this kind of problem, several problematic moments arise, which are manifested in various kinds of conflicts, and contradictions due to inconsistencies of input local data [5-6]. At the level of data syntax integration, the following contradictions arise [1-2]: ambiguity or contradiction of alphabets, mismatch of data types and formats, and mismatch of syntactic constraints. At the level of integration of data structures, the following are typical contradictions: inconsistency in methods of defining data units, contradictions in the types and methods of building connections, and a variety of ways to organize data [7]. The semantic component of the integration process is one of the most important and complex, since the problems of syntax and structure, in general, are solved at the technical and technological levels. Formation of an agreed interpretation of integrated data is impossible without human participation, as well as the application of methods and means of intelligent data processing. At the level of integration of semantics, conflict situations [7] arise as a result of the following factors:

- contradictions in the definition of concepts,
- ambiguity or different readings of names,
- use of incompatible metrics when forming data values,
- contradictions in defining relationships between data,
- contradictions of limitations and axioms of data interpretation,
- ambiguous interpretation of data values.

Eliminating the listed contradictions and conflicts between input data is one of the tasks of the data integration method. The multi-level data integration method is based on the multi-level data model developed in the previous section and involves the decomposition of the overall process into sub-processes of value integration, data syntax, structure and semantics. A key element of this approach to integration processes is the possibility of their implementation at the level of data meta-schemas, which allows to reduce the number of references to, in fact, data, the volumes of which can be significant. Due to this execution of data integration procedures, they are transferred to the meta-level, operating instead of data with their formalized description. Similar principles of replacing operations on information resources with operations on metadata that specify them are used in the concept of the "semantic web", which is part of the general concept of Web 2.0, data spaces [8] and DS of the second generation [9].

The purpose of the method of multi-level data integration is to determine the principles, composition and content of actions for the formation of the information resource of open information systems and the order of their implementation. Since the object of application of the method is information resources, it is advisable to organize the process of its development according to a set of requirements [9-10], which are applied to the design processes of information systems. The most acceptable is the application of the popular FURPS+ requirements model [11] defined according to RUP (Rational Unified Process) specifications and IEEE Std 1233a-1998 [12], IEEE Std 610.12-1990 [13] standards, which today are typical in the field of creating information systems and their components. Such a model provides for the formulation of basic and additional requirements for the final result of the development process. The main requirements that must be met by the method of multi-level data integration, according to the chosen approach, form a set that will be formulated as follows [1-2].

1. *Functionality* is compliance of the functionality of the multi-level data integration method with the requirements and needs of users of the final result.
2. *Usability* is the possibility of applying the method for implementation using an open information system.
3. *Reliability* is the ability of the method to provide the appropriate level of quality indicators of the results under the specified conditions during the time of its application.

4. *Performance* is the ratio between the level of costs for the implementation of the actions provided for within the method and the weight of the results obtained.
5. *Supportability* is the ability of the method to be applied in all situations and conditions in which the means of the corresponding information system function.

In addition to the set of basic requirements, the FURPS+ model provides for the formulation of additional requirements, which, unlike the basic set of FURPS requirements, are not unified and are formulated to reflect the specifics of the area and subject of the application. For the method of multi-level data integration, focused on open information systems, additional requirements are as follows [1-2].

6. *Portability* is the ability to move the tools that implement this method from one application environment to another without rebuilding them.
7. *Interoperability* is the ability to jointly apply the method and means that implement it with other methods and means of forming IRP (information resource processing) of open systems.
8. *Unification* is the use of typical concepts, objects and tools and the formation of results, by the uniform requirements of IS (intelligent system).

The fulfilment of such a set of requirements aims to ensure the appropriate level of quality of the method being developed and advantages over other known methods of data integration.

2. Related works

The problem of data syntax integration (syntactic integration) is fundamental to the integration of other components of their general description. Solving the problems of building a generalized structure and semantics of data is possible only based on a single agreed system of notation. The concept of data syntax itself is complex and takes into account various aspects of its representation in documents, DB, DS data repositories, etc. [14]. Taking this into account, the data syntax is presented as a combination of three components $G=<A, T, R>$, where A is an alphabet, T is a set of data types, and R is a set of syntactic restrictions [1-2, 15-18]. An alphabet defines a set of symbols that are used to represent data values in a defined environment. As a rule, the alphabet consists of letters, numbers, and special and service symbols. However, the definition of the alphabet is influenced, in particular, by such factors as the localization of the data processing environment to the language of the users, the nature of the tasks for which the data are used, the peculiarities of the processes of their storage, transmission and processing, the specifics of interpretation and the application of various data values. Along with traditional means of marking data, modern systems widely use graphics, sound, multimedia and other elements for their display and processing, as well as data of complex and complex types, streaming and active data, which creates additional difficulties in producing a single, consistent presentation of data [1-2]. The concept of data type is defined as the result of the classification of values according to the methods of representation and processing [15-18]. Today, along with such classic types as numerical, symbolic, logical, date-time, etc., specific types of data are widely used, which reflect the peculiarities of their content, processing and application. These are, in particular, such scalar types as "hyperlink", "currency", "object", "locator" and other, complex (aggregate) types - "array", "record", "set", "XML document " etc., object types, and user-defined data types. Such a variety of data types, on the one hand, creates additional opportunities for the image and processing of information resources, on the other hand, it complicates the means of supporting the data storage environment, the procedures for their joint application, transformation and unification. Constraints, as an element of data syntax, are used to unify forms of data presentation and create values adequate to the concepts and values they represent. Syntax restrictions are set in the form of quantitative indicators, dimensions, formats, templates, rules for forming values, defining a subset of permissible characters, etc. Such restrictions can be defined both at the IS/IT (information technology) level of data support and at the user level. Therefore, it is advisable to decompose the data syntax integration problem into the problems of alphabet integration, type integration, and constraint integration. The ratio of these tasks and the results of their execution are presented in Fig. 1.

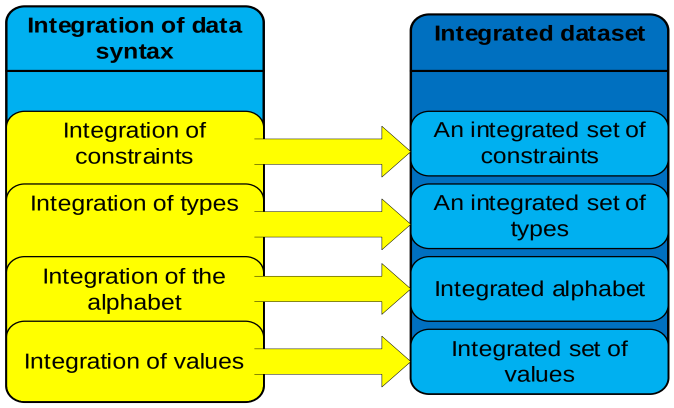
Integration of data syntax
Integration of constraints
Integration of types
Integration of the alphabet
Integration of values
Integrated dataset
An integrated set of constraints
An integrated set of types
Integrated alphabet
Integrated set of values

Figure 1: Schematic of the data syntax integration process

According to this scheme, the syntax of the image of the values of the integrated data set G^I is presented as a combination of three components $G^I = <A^I, T^I, R^I>$, where $A^I = I^A(A_1, A_2, \ldots, A_N)$ is the alphabet of the integrated data set, formed by integrating the input alphabets data sets $A_1, A_2, \ldots, A_N$; $T^I = I^T(T_1, T_2, \ldots, T_N)$ is the set of data types used in the integrated set, obtained as a result of the integration of the data types defined for the input data; $R^I = I^R(R_1, R_2, \ldots, R_N)$ is the set of constraints of the integrated data set formed by the integration of the constraints applied to the input data; I^A, I^T, I^R are integration operators, respectively, of alphabets, data types, and constraints. Each of these operators describes the mapping, respectively, of I^A is sets of input alphabets into the output global alphabet of the integrated data set, I^T is sets of local input sets of data types into the output global set of data types of the integrated set, I^R is sets of local input sets of syntactic constraints into the output global a set of syntactic restrictions of data of an integrated set [1-2, 15-18].

3. Models and methods

3.1. Basic principles of the extended data integration model

Further development of the concept of modelling data integration processes is possible due to the transition in the formal model from the concept of a scheme as an object of integration to the concept of a data set. Each data set is a combination of a scheme, as some formalized description of the composition and structure of data and a set of values (constants) formed according to the requirements of the scheme. In this way, the formal objects of the model are a set of input (local) data sets, an output (global) set of integrated data and a mapping that establishes correspondence between the elements of the input and output sets (Fig. 2a). Formally, such a model is presented as a triple of the form [1-2]: $<DS^L, Map(DS^L, DS^I), DS^I>$, where $DS^L = \{<D_i, \Sigma_i> \mid i = 1, \ldots N\}$ is a set of local input data sets; Σ_i is the data scheme of the i-th input set is made in terms of the input scheme description language L^L, Di is a set of values (constants) formed based on a set of characters of the input alphabet A^L; $DS^I = <D^I, \Sigma^I>$ is global output set of integrated data; Σ^I is the scheme of the global set of integrated data is made in terms of the description language of the original schemes L^I, D^I is the set of values of the original data set given by the symbols of the original alphabet A^I; $Map(DS^L, DS^I)$ is mapping of local input data into a global output set of integrated data [1-2]. The fundamental difference between this model and the formal model of M. Lenzerini is the concept of a global set of integrated data as a result of the integration process. At the same time, this set can be formed both by moving the values of the input data into the global environment and by mapping through virtual structures and data elements. In general, the proposed model corresponds to the real processes of integration to a greater extent than the formal model. Using such a model, it is possible to formulate a sufficiently accurate and detailed formal description of the main typical methods of data integration, such as consolidation, federalization, replication, hybrid integration and collage [1-2].

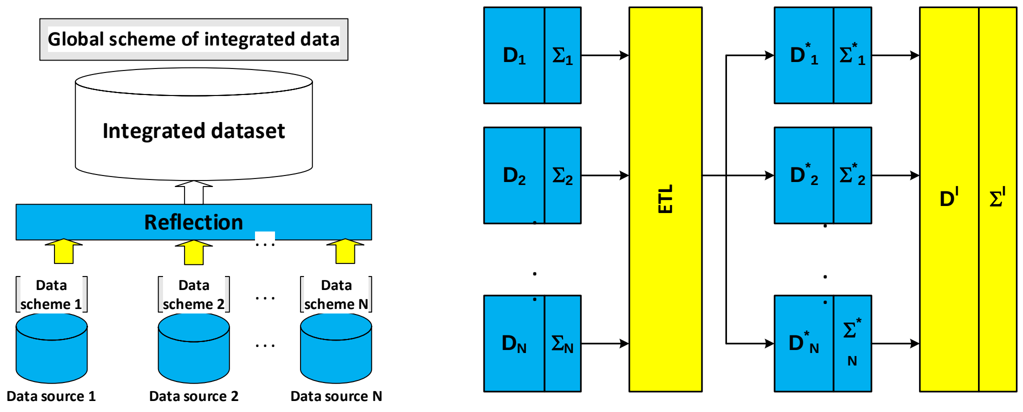

Figure 2: According to the improved model and data consolidation

3.2. Modelling the data consolidation process

A feature of the data consolidation method is the application of data extraction, transformation and loading procedures as the basis of the data integration process. The result of consolidation is a global set of integrated data, which has its scheme, which summarizes the composition and content of the schemes of the input sets. A description of the process of data consolidation according to the proposed generalized model is given in Fig. 2b [1-2]. The formal model of the data consolidation process has the form of a tuple $< \{<D_i, \Sigma_i > \mid i=1,\dots, N\}, ETL(<D_i, \Sigma_i >) \mid i=1,\dots, N,$ $<D^I, \Sigma^I> >$, where $<\{<D_i, \Sigma_i > \mid i=1,\dots N\}$ is a set of input data sets, each of which is given by a scheme Σ_i and a set of values D_i; $<D^I, \Sigma^I>$ is a global set of integrated data, with the scheme Σ^I and a set of D^I values; $ETL(<D_i, \Sigma_i>)$ is display of input data sets into the output by applying extraction procedures, loading conversion [1-2]. The key element of such a model is a mapping, which transforms each ith input set of the form $<D_i, \Sigma_i >$ into an intermediate data set of the form $<D^*_i,$ $\Sigma^*_i>$. The data set formed as a result of such a transformation differs from the initial one, primarily the fact that its composition, scheme and format are built by the requirements of a global integrated data storage environment. The next step is to move the intermediate data set to the global environment and merge the set of its values with the values set of the integrated data set.

3.3. Modelling the data federalization process

The method of data federalization differs in the way of forming a set of integrated values (Fig. 3a) [1-2]. Unlike consolidation, this method involves the formation of an integrated data set as some virtual image based on a set of local data sets. When accessing the integrated data, the corresponding image elements are implemented by substituting real values obtained from local sources. In this way, the integration process is implemented only at the scheme level, using as values the data placed at the local level. A formal model of data federation can be represented as an expression of the type

$$<\{<D_i,\Sigma_i > \mid i=1,\dots, N\}, View(<D_i,\Sigma_i >) \mid i=1,\dots, N, <D^I,\Sigma^I>>, \qquad (1)$$

where $< \{<D_i, \Sigma_i > \mid i=1,\dots N\}$ is a set of input data sets, each of which is given by a scheme Σ_i and a set of values D_i; $<D^I, \Sigma^I>$ is a global set of integrated data, with the scheme Σ^I and a virtual set of D^I values; $View(<D_i, \Sigma_i >)$ is mapping of the scheme of the input data set to the global scheme of integrated data. The key principle of such mapping is the formation of a description of a subset of data of the local input set in terms and composition that meets the requirements of the global scheme, while the set of values described by the new scheme Σ^*_i is a subset of the input local set of values $<D_i, \Sigma_i>$. The result of the mapping is a global scheme of integrated data, formed as a union of mappings of local schemes [1-2]: $\Sigma^I= \Sigma^*_1 \cup \Sigma^*_2 \cup \dots \cup \Sigma^*_N$, where N is the number of input local data sets. The set of values of the global initial set of integrated data is formed as a union of the set of projections of local data sets $\{D^*_i \mid i=1,\dots N\}$, built according to the set of schemes $\{\Sigma^*_i \mid$ $i=1,\dots, N\}$, each of which is formed by displaying $View(\Sigma_i, \Sigma^*_i)$: $D^I= D^*_1 \cup D^*_2 \cup \dots \cup D^*_N$.

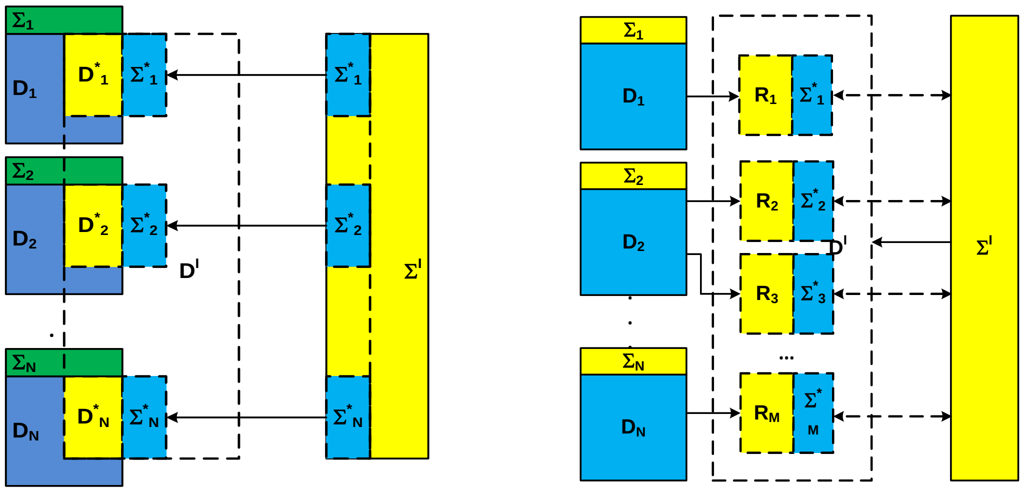

Figure 3: Data federalization and data replication

3.4. Modelling the data replication process

Data integration using the replication method involves the formation of a certain mapping (projection) of the local input data set according to a given mechanism, similar to the federalization method. The fundamental difference is that the result of displaying input data is not a virtual set of values, but some intermediate set of data that has its physical image formed according to some scheme, as in the case of data consolidation. But at the same time, the data set created in this way - a replica, cannot be moved to a specially defined storage environment. An advantageous global set of integrated data is formed as a union of a set of replicas. The general scheme of data integration by the replication method is shown in Fig. 3b [1-2]. The formal model of the data integration process using the replication method can be described as follows [1-2]: $<\{<D_i,\Sigma_i>|i=1,\ldots,N\}, Replicate(<D_i,\Sigma_i>) \mid i=1,\ldots,N, <D^I,\Sigma^I>>$, where $\{<D_i,\Sigma_i> / i=1,\ldots,N\}$ is a set of input data sets, each of which is given by a scheme Σ_i and a set of values D_i; N is the number of incoming local data sets; $Replicate(<D_i,\Sigma_i>)$ is display of the input data set, which forms a new set of values – a replica, which is a subset of the set of values of this set, formed according to the replica scheme, the replica scheme is a subset of the global integrated data scheme; $j=1,\ldots,M$, where M is the number of replicas, the number of which may differ from the number of data sources, since one or more replicas can be formed on the basis of one input local set, the result of mapping $Replicate(<D_i,\Sigma_i>)$ is a set data of the form $<R_j,\Sigma^*_j>$, where Σ^*_j is a replica scheme, R_j is a set of values; $<D^I,\Sigma^I>$ is a global set of integrated data, with a scheme Σ^I and a set of D^I values, while the scheme Σ^I is a union of the schemes of all replicas $\Sigma^I = \Sigma^*_1 \cup \Sigma^*_2 \cup \ldots \cup \Sigma^*_M$, and a set of D^I values by combining sets of replica values – $D^I = R_1 \cup R_2 \cup \ldots \cup R_M$.

3.5. Modeling the hybrid data integration process

A feature of data integration using the hybrid method (Fig. 4) is the combination of the possibilities of the three methods described above – consolidation, federalization and replication – in one process. In this case, the global initial set of integrated data is formed as a heterogeneous entity that combines several segments, each of which is formed based on different methods and technologies [1-2]. In general, the hybrid integration model can be described by a tuple of the form $<\{<D_i,\Sigma_i> / i=1,\ldots N\}, Map_i(<D_i,\Sigma_i>) \mid i=1,\ldots,N), <D^I,\Sigma^I>>$, where $<\{<D_i,\Sigma_i> \mid i=1,\ldots,N\}$ is a set of input data sets, each of which is given by the scheme Σ_i and a set of values D_i, N is the total number of input local data sets, the set of input local data sets is divided into three subsets, according to the integration methods applied to them; $<D_C,\Sigma_C>$ is input local data sets to which the data consolidation method is applied; $<D_R,\Sigma_R>$ is input local data sets to which the data replication method is applied; $<D_F,\Sigma_F>$ is input local data sets to which the data federalization method is applied; $Map_i(D_i, D^I)$ is mapping of the input local data set to the global set of integrated data, the type of mapping is different for different data sets, depending on the integration methods applied to it – consolidation, federalization or replication; $<D^I,\Sigma^I>$ is a global set of integrated data, with a Σ^I scheme and a set of D^I values, while the Σ^I scheme is a union of schemes formed by different integration methods $\Sigma^I = \Sigma^*_C \cup \Sigma^*_F \cup \Sigma^*_R$, where Σ^*_C is a data scheme formed as a result of the consolidation of input local data, Σ^*_F is a data scheme formed by federalization, Σ^*_R is replication, the set of values of the global initial set of integrated data is formed as a union of three segments [1-2]: $D^I = D^*_C \cup D^*_F \cup R^*$, where D^*_C is the set of values formed as a result of consolidation of input local data, D^*_F is the set of values formed by federalization, R^* is replication.

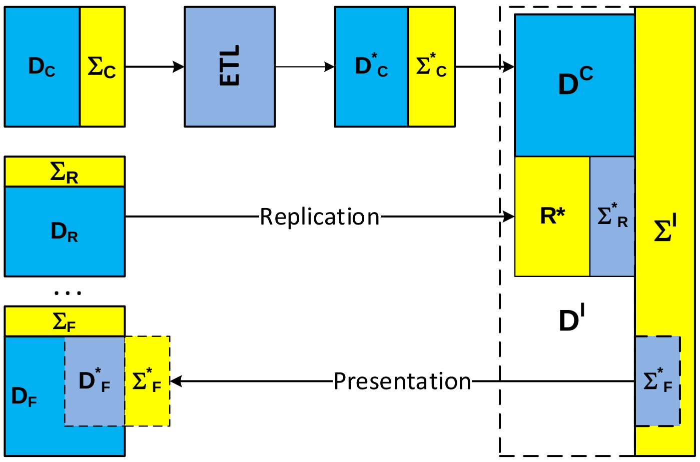

Figure 4: Hybrid data integration process model

3.6. Modelling the data collage process

Collage (mashup), as a method of integration, is most often used in Web-systems to combine in a single presentation of data received from different sources, different in form, structure, and methods of representation, but combined by a common content/application. The peculiarity of collage is the absence of a permanent scheme of integrated data and the dynamic formation of a set of values with each access to resources of this type. At the same time, the initial data are combined in various ways, forming, as a result, arbitrarily structured hybrid data. The general scheme of the data collage process is shown in Fig. 5a [1-2].

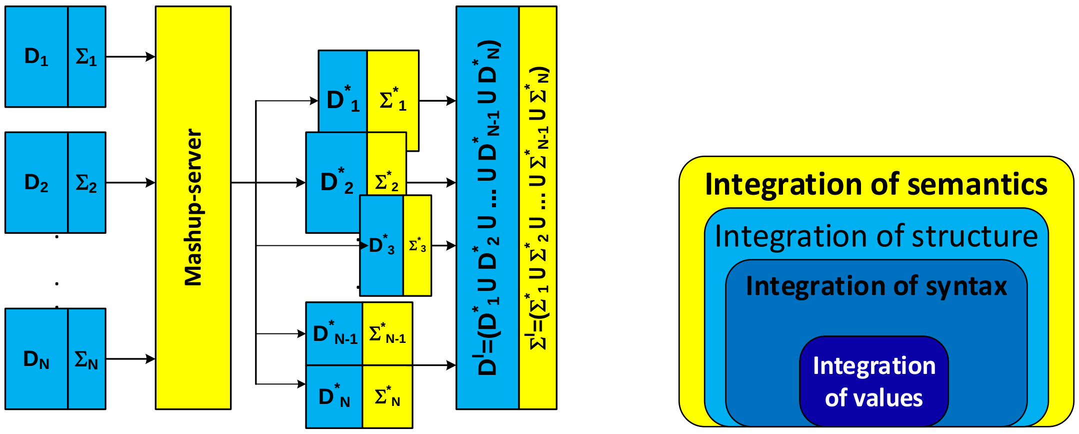

Figure 5: Model of the data collage process and multilevel model of integration

The formal model of the data collage process is described as a tuple:

$$\{<D_i,\Sigma_i> \mid i=1,\ldots N\},\ Mashup_i(<D_i,\Sigma_i>),\ <D^I,\Sigma^I>>, \qquad (2)$$

where $\{<D_i, \Sigma_i> / i=1,\ldots N\}$ is the input local data set, with scheme Σ_i and set of values D_i; $<D^I, \Sigma^I>$ is the initial global set of integrated data; $Mashup_i(<D_i, \Sigma_i>)$ is a mapping that forms a data collage element for further combining parts of input local data sets into a single view. In the collage process, some subset of D^*_i values is selected from each input local set, which is described by the scheme Σ^*_i. From these parts, by combining and superimposing different types of data and forming a global scheme as a combination of schemes, a single integrated data set $<D^I, \Sigma^I>$ is formed for presentation to the user. The difference between integration by collage and other methods is the absence of physical storage of integration results and the dynamic formation of a global scheme of integrated data upon user request [1-2].

3.7. Formal modelling of data integration processes and results

The analysis of the results of modelling data integration processes using various methods and methods using an extended formal model allows the following conclusions to be drawn [1-2]:

- the extended formal model of data integration can be applied to model resource-centric and schema-centric data integration by methods of consolidation, federalization, replication, hybrid integration and collage. Therefore, the proposed model is invariant to the methods and paradigms of data integration, which allows us to conclude about its universality;
- both the data itself in the form of a set of values (constants) and their formalized description – a scheme – appear in the integration processes. Integration involves performing a series of isomorphic transformations over the input schemas to form a global output schema of integrated data and transformations of sets of values of input data to form a set of values of the output set of integrated data;
- in the process of integration, operations of moving, reformatting, selecting, projecting, combining, superimposing, etc. are performed on the input data. as a result, new sets of data are created, which differ from the input ones in composition, content, structure, presentation and methods of application;
- the listed features of integration processes are common to various integration methods and paradigms, which allows us to conclude the possibility of creating a single generalized apparatus for describing data integration processes, independent of integration technologies, subject area, content, purpose and order of application of integrated data.

The general conclusion regarding the modelling of data integration processes is that as a result of integration, new data values, new forms and presentation formats, new data structures, new content and new purpose of data are created [1-2]. So, data integration has technical, syntactic, structural, semantic and pragmatic aspects. Accordingly, each of these aspects involves the use of its methods and means of data description in integration processes, which allows dividing the overall integration process into several sub-processes that implement one of the above-mentioned aspects. This is reflected in the generalized model of data integration processes, which is proposed to be called a multi-level formal model of integration.

3.8. Multilevel data integration model

The results of the analysis of formal models of data integration using various methods show that in the process of integration, significant transformations of the composition, content and form of data occur. This means generating, based on input sets, a set of new final data that have fundamentally different properties. This creates the basis for further development and improvement of the formal model of the data integration process by introducing into its composition elements that describe the main properties of the data and the order of their change. According to the concept of presenting data as a formal system, the data form some formal language that is used to denote a set of values and concepts from a certain SA in the environment of the information system [1-2]. The basis of the construction of language structures is a certain set of symbols - the alphabet. Mandatory and integral properties of data in such a data representation are their syntax, semantics, and structure. At the same time, syntax is used to determine the order of presentation of lexical constructions (constants), for the presentation of real values, and the order of formation of new lexical units based on given ones. Semantics provides an ordered and unified description of the ways of interpreting data, that is, it connects them with the actual values that take place in the subject area, forming, due to this, the content of the data and their pragmatics. With the help of the structure, the order of formation of data units, their combination and arrangement is described. The structure, in turn, determines not only the order of presentation and storage of data but also the methods of its processing and application [1-2]. In the general case, the definition of an arbitrary data set DS forms a system of the form $DS=<D, G, S, H>$, where D is a set of values that represent a set of concepts of some subject area, G is a formalized representation of the data syntax, S is a formalized description of the structure data, H is a formalized presentation of data semantics. In this way, the formal presentation of a data set as a tuple of the form $<D, \Sigma>$, where D is a set of values, Σ is a data scheme, is changed to a tuple of the form $<D,\Theta>$, where $\Theta=<G, S, H>$, formal presentation of the syntax, structure and semantics of the data in this set, which, in the future, we will call its meta-schema. A meta-schema is an extension of the concept of a scheme by supplementing the description of the structure and constraints of data with a formalized description of their syntax and semantics. The introduction of the concept of a meta-scheme makes it possible to build a much broader and more detailed description of data properties in integration processes, compared to a scheme. In general, the process of data integration involves several actions related to their transformation and the formation of new data based on the initial ones. It is considered a sequence of actions involving matching, transformation, merging and filtering of data, and aims to form a final set of DS data based on a set of initial sets, it is formally represented by an expression of the form [1-2]:

$$DS=I(DS_1, DS_2, \ldots, DS_N), \qquad (3)$$

where I is the data integration operator, $DS_1, DS_2, \ldots, DS_N$ is the set of input initial data sets, and N is the number of data sets participating in the integration process. In general, such data sets may contain repeated values, i.e. [1-2]:

$$D_1 \cap D_2 \cap \ldots \cap D_N \neq \varnothing. \qquad (4)$$

Given the data model, which is based on the specification of their syntax, semantics and structure $DS=<D,\Theta>=<D, G, S, H>$, the formal definition of the integration process can be reduced to actions on these components, replacing the DS^I value with a detailed description all components of data definition as follows [1-2]:

$$<D^I,\Theta^I>=<D^I, G^I, S^I, H^I>=I(<D_i,\Theta_i> \mid i=1,\ldots N)= \qquad (5)$$
$$=I(<D_1, G_1, S_1, H_1>,<D_2, G_2, S_2, H_2>, \ldots, <D_N, G_N, S_N, H_N>),$$

where $<D_i, G_i, S_i, H_i>$, $i=1,2, \ldots, N$ is the detailed formal representation of the i^{th} data set.

In this way, the problem of data integration can be decomposed into separate problems of data value integration, syntax integration, structure integration, and semantic integration. The general data integration operator I is presented as a combination $I=<I^D, I^G, I^S, I^H>$, where I^V is the value integration operator, I^G is the syntax integration operator, I^S is the data structure integration operator, and I^H is the semantics integration operator. At the same time, the integration process will be decomposed into corresponding sub-processes, which can be described by a formal scheme of the form [1-2]:

$$<D, G, H, S> =<I^D(D_1, D_2, \ldots, D_N), I^G(G_1, G_2, \ldots, G_N), I^S(S_1, S_2, \ldots, S_N), I^H(H_1, H_2, \ldots, H_N)>. \qquad (6)$$

The mutual relationship of these processes and their classification by levels are shown in Fig. 5b. According to such a scheme, each subsequent level of integration is based on the results of the previous one. Thus, the semantic integration of data is possible only after the integration of their structure, which, in turn, requires the construction of an integrated syntax that defines the methods of data representation and the integrated set [1-2]. The presentation of data in integration processes as a formal system allows to develop and improve the theoretical conceptual foundations of data integration due to a higher level of abstraction and the possibility of creating integration models that do not depend on the nature, content, subject area, methods and technologies. As a result of the study of formal models of data integration using the methods of consolidation, federalization, replication, collage, and the hybrid method, it was found that the basic principles and concepts are common to all methods, which makes it possible to build a unified approach and method to data integration that will generalize the methods known today. In the process of integration, not just a mechanical combination of data is performed, but the formation of new data, which has fundamentally new properties, differs from the input data in syntax, structure, semantics and the order of application. This makes it possible to distinguish the processes of integration of data values, their syntax, structure and semantics [1-2]. The model developed in the way described above defines and substantiates the possibility of creating a universal method of data integration, which summarizes the capabilities of currently known approaches, and also creates an opportunity to move the integration processes from the procedures for processing the actual data and their schemes to the procedures for manipulating metadata that describe the properties and specifics of the set data, which is the object of integration.

4. Experiments, results and discussion

4.1. Syntactic integration of data

4.1.1. Integration of alphabets

The integration of alphabets at the stage of designing a unified integrated data processing environment consists in creating a consistent set of symbols for representing values from the resulting data set - the integrated A^I alphabet, such that for each symbol of the input alphabet A_i, which is used to represent the value of the input data set D_i (i=1,2, ..., N), there is a unique mapping α_i: $A_i \rightarrow A^I$, which matches each symbol of the input alphabet of the ith data set $\sigma_i(A_i)$ with a symbol of the integrated alphabet – $\sigma(A^I)$. The following ratios of input and integrated data alphabets are possible (Fig. 6) [1-2]:

- the input alphabet is a subset of the integrated alphabet and has no intersections with other input alphabets (A_1);
- the input alphabet is a subset of the integrated alphabet and has a non-empty intersection with another alphabet that is a subset of the integrated alphabet (A_5);
- the input alphabet is a subset of the integrated alphabet and has a non-empty intersection with another alphabet that has a partial intersection with the integrated alphabet (A_2);
- the input alphabet has a non-empty intersection with the integrated alphabet and an alphabet that is a subset of the integrated alphabet (A_3);
- the input alphabet is not a subset of the integrated alphabet and has a non-empty intersection with another input alphabet, which, in turn, has a partial intersection with the integrated alphabet (A_4);
- the input alphabet is not a subset of the integrated alphabet and does not have non-empty intersections with another input alphabet (A_6);
- the input alphabet is not a subset of the integrated alphabet, but at the same time has a non-empty intersection with other input alphabets (A_7, A_8).

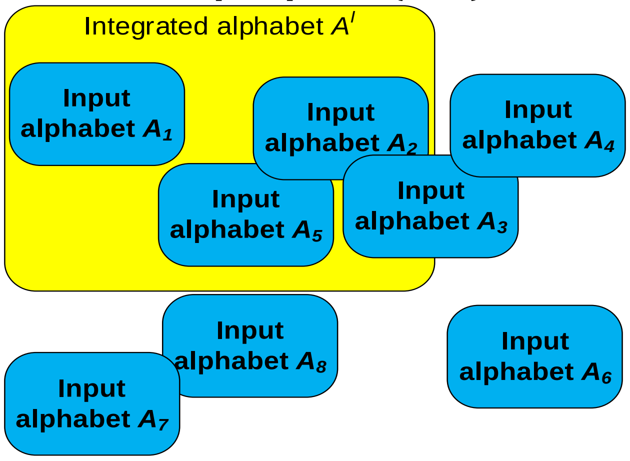

Figure 6: Diagram of the ratio of input and integrated alphabets

We present the process of building an integrated alphabet as a sequence of solving interconnected problems according to the following scheme [1-2, 15-18].

1. Let A^0 be some initial set of symbols of the integrated alphabet.
2. For each of the input alphabets A_i, i=1,2,..., N, the ratio $A_i \subseteq A^0$ is checked. If it is fulfilled, then all characters of the alphabet A_i used to represent the values of the data set D_i are also acceptable for representing the corresponding values in the integrated data set D. Therefore, it can be assumed that the input data can be included in the integrated set without changing the form of their presentation.
3. In this case, the phenomenon of polysemy of symbols is possible. We will call polysemous symbols that have the same shape and reflect different meanings, for example, the Ukrainian letter "I", the Latin letter "I", the Roman numeral "I" (1), the Latin letters A-F, which are used to represent both letters and numbers in the 16th number system, etc. Such a phenomenon may cause an ambiguous interpretation of data values and their content in the future. The problem of polysemic characters has the following solutions:

- banning the use of the same symbols to denote different concepts - this method involves defining a single image for all symbols that have the same shape; this option for solving the problem of polysemic symbols is possible in cases when they are used for formal meanings that do not have additional (phonetic, lexical or substantive) interpretation (for example, the same type of use of Latin and Cyrillic letters that match the spelling in car registration numbers); in this case, problems are possible when interpreting, reading or phonetizing data values;
- the use of polysemic symbols without restrictions - for each of the symbols that have the same form, they retain their method of application; in this case, the problem of polysemy of symbols of the integrated alphabet is not solved in the process of integration, but is transferred to the level of data application;
- replacement of identically shaped symbols with an alternative image - transliteration; this transformation allows you to eliminate the polysemy of characters without narrowing the possibilities of data presentation.

4. If the set of characters of the input alphabet is not a subset of the integrated alphabet – $A_i \not\subset A^0$, then it is divided into two subsets – $A_i^1=A_i \cap A^0$ and $A_i^2=A_i \backslash A^0$. The first includes symbols that are elements of the integrated alphabet, the process of integration in this case is described above. The set A_i^2 includes characters of the input alphabet that are not elements in the current state of the integrated alphabet A^0. In such a situation, character polymorphism is possible. We will consider polymorphic symbols to be symbols that differ in image form and are used to denote the same concepts. For example, upper/lower case letters in words represent the same sounds, numerical values can be represented in different number systems, using Arabic, Latin numbers or letters, etc. The appearance of polymorphic symbols in alphabets is a possible cause of ambiguous perception and interpretation of data values during their processing. As for solving the problem of processing polymorphic symbols, the following solutions are possible [1-2, 15-18]:

- replacing polymorphic symbols with homomorphic images, i.e. bringing symbols of different shapes to a single form, for example, using only uppercase or lowercase letters, only Arabic numerals, which replace similar Roman numerals, etc.;
- parallel application of polymorphic images of synonymous symbols without restrictions, in which case their interpretation will depend on the context and application of data values;
- creating your interpretation and rules for using polymorphic symbols - this path requires a detailed analysis of their properties, but allows you to significantly expand the capabilities of the integrated alphabet in terms of displaying data, for example - proper names begin with capital letters, operators or operations are denoted by special symbols, Arabic numerals are used to represent quantitative values, and Roman – ordinal, etc.

Regarding the set of symbols $A_i^2 = A_i \backslash A^0$, the following options are possible

- prohibiting the use of symbols from the A_i^2 set to represent integrated data;
- expansion of the integrated alphabet due to the inclusion of a set of symbols A_i^2 in its structure, with the formation of the next version $A^1 = A^0 \cup A_i^2$;

- transliteration – replacing symbols that are not elements of the integrated alphabet with symbols from the A^0 alphabet.

5. As a result of iterative repetition of the described sequence of actions, an integrated alphabet $A^I=I_A(A_1, A_2, ..., A_N)$ is formed, which defines a set of symbols for presenting data values in an integrated set.

4.1.2. Integration of data types

The integration of data types in the construction of the output integrated set consists in the formation of a set of data types T^I, such that for each of the types applied in the input data sets there is a mapping $\tau: T_i \rightarrow T$, which establishes a one-to-one correspondence between the data types $t(T_i)$ applied in the input set D_i (i=1,2, ..., N) and $t(T^I)$ are the data types used in the integrated data set D^I. The mutual relationship of different sets of data types in the process of integration is shown in Fig. 7. As can be seen from the diagram, similar to the process of integration of alphabets, input sets of data types may or may not have full or partial intersections with the integrated, and may or may not have mutual intersections with each other. The process of forming an integrated set of types involves the sequential execution of such actions [1-2, 15-18].

1. Let $T^0=\{ t_1(T^0), t_2(T^0), ..., t_n(T^0)\}$ be some initial set of types that are defined in the integrated data set forming the initial state of the T^I type set.
2. For each set of input data D_i (i=1,2, ..., N), check the ratio $T_i \subseteq T^0$, which determines the agreement of the types of the input set with the types of the integrated data set D^I.
3. Fulfilment of this condition does not guarantee that only types allowed for use in the integrated set are used to represent the data of the input set D_i since there is a possibility of type polysemy. We will call polysemous types that have the same designation and differ in implementation methods. For example, a value of the "date/time" type can be represented by both numeric and character values, the "text" type in some applications represents character strings, in others - notes, values of the logical type are represented as numeric or bit, etc. Discrepancies of this nature are a potential factor in possible errors in data processing, ambiguous interpretation and obtaining incorrect results. This kind of inconsistency of data types has, in particular, the following solutions [1-2]:

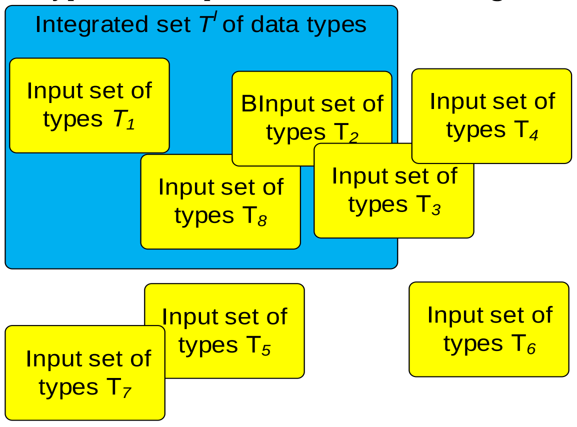

Figure 7: Ratio of input and integrated data type sets

- replacement of homonymous data types with new ones, which by definition do not coincide with others;
- reformatting the values of the input data set to the format of the corresponding types defined in the integrated data set;

4. If the set of data types T_i, which are applied in the input set D_i, exceeds the set of data types of the integrated set D, i.e. $T_i \not\subset T^0$, this indicates the presence in the input set of data belonging to such types that are not valid data types of the original integrated set. Therefore, the data types of the input set are divided into 2 such subsets:

- a subset of types $T_i^1 = T_i \cap T^0$, which are included in the set of types of the integrated data set;
- a subset of types $T_i^2 = T_i \setminus T^0$, which are not included in the set of types of the integrated data set.

For the subset of T_i^1 types, the type-matching procedure is as described above. In the case when the data of some input set D_i belong to types that are not supported in the original integrated data set D^1, the following variants of further transformations are possible [1-2, 15-18]:

- expansion of the set of data types of the integrated set by supplementing it with a subset T_i^2 of the data types of the set D_i, which is implemented by constructing the next version of the set of permissible data types of the integrated set $T^1 = T^0 \cup T_i^2$;
- conversion of data from the format of the types of the set T_i^2 to the corresponding types from the set T^0, that is, replacing each data value of the type $t(T_i^2) \in T_i^2$ with a similar value represented according to the requirements of the type $t(T^0) \in T^0$.

5. Another contradiction in the processes of integration of data types is the occurrence of polymorphism of data types in the integrated set and input sets. We will call polymorphic data types that differ in form of representation but are identical in interpretation (for example, REAL and FLOAT, BOOLEAN and LOGICAL types, etc.). In this case, situations may arise in which data of the same actual type will be incompatible with each other when performing actions on them, which, in turn, is a potential cause of errors and contradictions in the data. There are two possible ways to resolve this contradiction [1-2, 15-18]:

- bringing polymorphic types to a single method of their determination due to the removal of such types that repeat others;
- compatible application of all possible options for defining data types due to the creation of additional means of maintaining the polymorphism of data types and their coordination.

The first way is easier to implement, and the second - expands the possibilities for describing and manipulating data in an integrated set.

6. The result of the steps described above is a generalized and agreed list of data types that are used in determining the units of the integrated set $T^I = I_T(T_1, T_2, ..., T_N)$ [1-2, 15-18].

4.1.3. Integration of syntactic data constraints

The integration of restrictions, which are used when forming data values of some input sets, involves the formation of such a set of restrictions $R^I=(r_1(R^I), r_2(R^I), ..., r_m(R^I))$ that for each restriction $r(R_i) \in R_i$, applied to some input data set D_i (i=1,2, ..., N) there is a one-to-one correspondence given by the mapping $\rho: r(R_i) \rightarrow r(R^I)$. However, unlike alphabets and data types, the restrictions are not free elements, they are formulated and applied only to specific data types, categories or values. Therefore, each restriction that is applied to a certain set of data D_i is defined as a condition of the form $r(R_i, t(T_i), D^j_i)$, which is determined by such factors as belonging to a set of restrictions R_i, binding to a certain type of data – $t(T_i)$, and the scope is some subset of the data set $D^j_i \in D_i$. Therefore, the problem of integrating the set of constraints of the input data sets into a single set of constraints of the integrated set can be solved only after performing the integration of the alphabet and data types. The ratio of the sets of input data constraints and the integrated data set is shown in Fig. 8. The input sets of constraints can have partial intersections with each other, be subsets of each other, be completely independent, be fully or partially part of the integrated set formed by their integration, or not have an intersection and not be part of it. The general sequence of the process of integration of input syntactic constraints, the purpose of which is to create a single, consistent and complete set of constraints applied to values from the integrated data set, is presented in the form of a scheme of actions [1-2, 15-18].

1. Let $R^0=(r_1(R^0), r_1(R^0), ..., r_1(R^0))$ be initial set of constraints of some integrated data set D.
2. For each of the sets of restrictions R_i of the input data sets D_i (i=1,2,..., N), we check the condition $R_i \in R^0$.

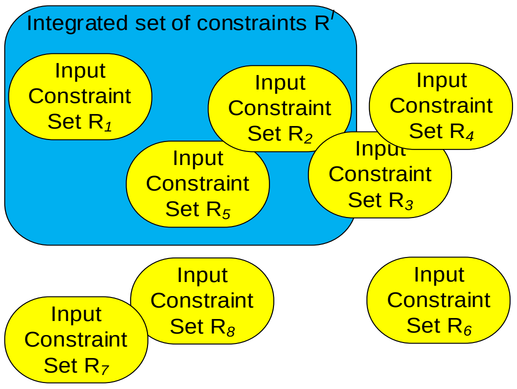

Figure 8: Diagram of the ratio of input and integrated sets of constraints

3. The fulfilment of this condition means that each of the constraints of the input data set takes place in the integrated set. But for the final determination of the possibility of applying restrictions to integrated data, the following factors are additionally checked [1-2, 15-18]:

- the presence among the data types of an integrated set of types for which restrictions are defined, i.e. for each of the restrictions $r^j(R_i) \in R_i$ there is $t^j(T_i) \in T^I$, where $t^j(T_i)$ is the data type to which the restriction is applied, T^I is multiple types of integrated dataset;
- the presence among the set of values of an integrated data set of values for which restrictions are defined, that is, for each of the restrictions $r(R_i) \in R_i$, $D^j_i \in D^I$ is performed, where D^j_i is a subset of values of the data set D_i to which the restriction is applied, D^I is an integrated data set;
- the fulfilment of these requirements ensures the possibility of applying the restriction to the data of the integrated set, and the lack of appropriate data types and/or values makes it impossible to apply this restriction to the integrated data set.

4. If among the set of constraints R_i of some input set D_i some are not constraints of the integrated data set D^I, i.e. $R_i \not\subset R^0$, the set of constraints is divided into subsets $R_i^1 = R_i \cap R^0$ and $R_i^2 = R_i \setminus R^0$.
5. The set of restrictions R_i^1 is consistent with the set R^0 and the process of its integration is performed as described above, but the set of restrictions R_i^2 has the following integration options [1-2, 15-18]:

- restrictions from the set R_i^2 are applied to values and/or data types that are not part of the integrated set;
- restrictions from the set R_i^2 are applied to the values and data types included in the integrated set.

In the first case, each of the restrictions that does not have an object of application can be removed without loss from the set of restrictions of the integrated set. In the second, the procedure for matching additional constraints R_i^2 and a set of constraints R^0 of the integrated data set is applied. The reconciliation of these sets of restrictions is achieved due to [1-2]:

- extraction of R_i^2 constraints from further application in the set of constraints of the integrated data set D^I;
- transformation of restrictions included in the set R_i^2 by replacing them with equivalent ones in content and application from the composition of the set R^0 according to the principle - each of the restrictions $r(R_i^2) \in R_i^2$ is matched with the restriction $r_1(R^0) \in R^0$, which is defined for types and values of the integrated data set;
- expansion of the set of constraints R^0 of the integrated data set by supplementing it with elements of the set of constraints R_i^2 to form a new version $R^1 = R^0 \cup R_i^2$.

6. The result of performing a sequence of actions on the integration of syntactic restrictions of input data sets is the formation of such a list of data presentation requirements that can be applied to determine additional properties of data values from the integrated set.

4.1.4. Procedure and requirements for syntactic data integration

By performing a sequence of actions on the integration of alphabets, data types and syntactic constraints according to the scheme described above, a complete and consistent set of elements of the integrated syntax of G^I data is formed, which is used as a method and means of displaying integrated data obtained as a result of the processes of data extraction, transformation and loading in DS, as well as with their dynamic integration in operational systems. At the same time, the problem of detecting and correcting the elimination of contradictions between the local syntax of the input data sets to be integrated is solved. The general order of syntactic integration describes such a sequence of steps [1-2].

Step 1. Constructing an integrated alphabet as a complete and consistent set of characters to represent data values in the original integrated set. Performing this step involves the implementation of the following procedures.

1. Detection and elimination of contradictions of input alphabets in the process of syntax integration. This is done according to the following rules.
   - Elimination of character polymorphism – the alphabet of the resulting integrated data set cannot contain different characters with the same interpretation. Such a rule for matching the representation of the symbols of the alphabets A_i and A_j is described by an expression of the form

$$Alph_1(A_i, A_j): \neg\exists\alpha\in A_i, \neg\exists\beta\in A_j \mid int_i(\alpha) = int_j(\beta), \quad (7)$$

where $Alph_1$ is the rule identifier, α, β are symbols of the input alphabets A_i and A_j; $int_i(\alpha)$, $int_j(\beta)$ are symbol interpretation functions.

   - Elimination of character polysemy - the output alphabet of the resulting integrated data set cannot contain the same characters with different interpretations. The rule for matching the interpretation of the symbols of the alphabets A_i and A_j is described by an expression of the form

$$Alph_2(A_i, A_j): \neg\exists\alpha\in A_i, A_j \mid int_i(\alpha) \neq int_j(\alpha), \quad (8)$$

where $Alph_2$ is the rule identifier; α is a symbol included simultaneously in the input alphabets A_i and A_j; $int_i(\alpha)$, $int_j(\alpha)$ are functions for interpreting symbols in the alphabets A_i and A_j.

2. Construction of an integrated A^I alphabet by combining agreed local input alphabets according to the rules defined in clause 1:

$$A^I = I^A(A_1, A_2, ..., A_N) = A_1\cup A_2\cup ...\cup A_N \mid \quad (9)$$
$$Alph_1(A_1, A_2, ..., A_N) = true \wedge Alph_2(A_1, A_2, ..., A_N) = true,$$

where I^A is the alphabet integration operator; A_1, A_2, ..., A_N is a set of input local alphabets; $Alph_1(A_1, A_2, ..., A_N)$, $Alph_2(A_1, A_2, ..., A_N)$ are the rules for matching alphabets defined in paragraph 1. a) and 1. b).

Step 2. Construction of a single, consistent list of data types that are used in the original integrated set. The process of integrating data types involves the following actions.

3. Identification and elimination of inconsistencies in data typing methods from input sets. The following rules apply to this.
   - Elimination of polymorphism of types - the data types used in the original integrated data set cannot contain different types that have the same interpretation. Such a matching rule for sets of data types T_i and T_j is described by an expression of the form

$$Type_1(T_i, T_j): \neg\exists t_1\in T_i, \neg\exists t_2\in T_j \mid int_i(t_1) = int_i(t_2), \quad (10)$$

where $Type_1$ is the rule identifier; t_1, and t_2 are data types of input resources, which are included in the sets of types T_i and T_j, respectively; $int_i(t_1)$, $int_j(t_1)$ are interpretations of types t_1 and t_2, respectively, in sets T_i and T_j.

   - Elimination of polysemy of types - in the composition of types used for data typing of the resulting integrated data set, there cannot be identically defined types with different interpretations. The rule for matching the interpretation of sets of types T_i and T_j is described by an expression of the form $Type_2(T_i, T_j): \neg\exists t\in T_i, T_j \mid int_i(t) \neq int_j(t)$, where $Type_2$ is the rule identifier; t is a type included simultaneously in the input local sets of types T_i and T_j; $int_i(t)$, $int_j(t)$ are type interpretation functions, respectively, in the sets T_i and T_j.

4. Construction of a set of types of the original integrated resource T^I by harmonizing and combining local input sets of types according to the rules defined in clause 1:

$$T^I = I^T(T_1, T_2, \ldots, T_N) = T_1 \cup T_2 \cup \ldots \cup T_N \,/\, Type_1(A_1, A_2, \ldots, A_N) = true \wedge Type_2(A_1, A_2, \ldots, A_N) = true, \quad (11)$$

where I^T is the integration operator of input sets of data types; $T_1, T_2, \ldots, T_N$ are sets of input local data types; $Type_1(T_1, T_2, \ldots, T_N)$, $Type_2(T_1, T_2, \ldots, T_N)$ are the rules for matching alphabets defined in clauses 1. a) and 1. b).

Step 3. Formation of a single consistent set of syntactic constraints by merging and matching local constraint sets of input datasets.

- Detection and elimination of contradictions in the syntactic constraint sets of input datasets. The following rules apply to this.
  a. Elimination of polymorphism of constraints - in the composition of the set of syntactic constraints, which are applied in the resulting integrated data set, there cannot be different constraints that have the same interpretation. Such a rule for matching the sets of constraints R_i and R_j is described by the expression

$$Restrict_1(R_i, R_j){:}\neg\exists r_1 \in R_i, \neg\exists t_2 \in R_j \,/\, int_i(t_1) = int_i(t_2), \quad (12)$$

where $Restrict_1$ is the rule identifier; r_1, r_2 are syntactic restrictions applied to input resources, which are included in the sets of restrictions R_i and R_j, respectively; $int_i(r_1)$, $int_j(r_1)$ are an interpretation of syntactic constraints r_1 and r_2 in sets R_i and R_j.

  b. Elimination of polysemy of constraints – in the set of syntactic constraints, which are applied to the data of the resulting integrated set, there cannot be identically defined constraints that have different interpretations. The rule for matching the interpretation of the sets of constraints R_i and R_j is described

$$Restrict_2(R_i, R_j){:}\ \neg\exists r \in R_i, R_j \,/\, int_i(r) \neq int_j(r), \quad (13)$$

where $Restrict_2$ is the rule identifier; r is a syntactic restriction that is simultaneously included in the input local sets of restrictions R_i and R_j; $int_i(r)$, $int_j(r)$ are functions for interpreting constraints in the sets R_i and R_j.

- Construction of a set of syntactic restrictions of the original integrated R^I resource by harmonizing and combining local input sets of application types defined in clause 1, rules:

$$R^I = I^R(R_1, R_2, \ldots, R_N) = R_1 \cup R_2 \cup \ldots \cup R_N \,/\, Restrict_1(R_1, R_2, \ldots, R_N) = true \wedge Restrict_2(R_1, R_2, \ldots, R_N) = true, \quad (14)$$

where I^R is the syntactic constraint integration operator; $R_1, R_2, \ldots, R_N$ are sets of input local syntactic constraints; $Restrict_2(A_1, A_2, \ldots, A_N)$, $Restrict_2(A_1, A_2, \ldots, A_N)$ are syntactic restriction matching rules defined, respectively, in clauses 1. a) and 1. b).

Step 4. Constructing an output syntax for representing data in an integrated set. The output integrated data syntax G^I is formed based on the integrated consistent alphabet A^I, the integrated consistent set of data types T^I and the integrated output set of syntactic constraints R^I

$$G^I = \langle A^I, T^I, \mathrm{R}^I \rangle. \quad (15)$$

The syntax formed in this way provides a correct, consistent and unambiguous representation of the data values in the data set, which is created as a result of their integration.

4.2. Structural data integration

4.2.1. General principles of integration of data structures

The problems of creating and maintaining heterogeneous structures of integrated information resources, in general, go beyond the functional capabilities of traditional data storage environments implemented by DB servers and DBMS [15-19]. Today, many other abstractions and management methods are known, which either confirm their suitability or are removed from the management environment of integrated heterogeneous content [1-2]. A comprehensive solution to the problems of management and application of integrated content, which includes both structured (relational data) and loosely structured data, provides tools that implement technologies for the joint processing of such resources as structured data, text, spatial, temporal, visual, multimedia data, procedural data, triggers, streams and data queues, imprecise and fuzzy data [20-23]. The heterogeneity of input data to be integrated extends to the diversity of their structures (Fig. 9). Modern IS uses data of various levels and forms of structuring. Along with the structured data stored in the DB, the information resources of open IS contain so-called non-relational data, in particular, weakly structured (semi-structured) data, data without a prior description of the structure (self-structured), stream data, procedural data, etc. [23]. Building a single agreed description of the structure of disparate data is one of the tasks performed in the process of their integration. A general description of the structure of the integrated data set is given as [1-2, 15-18]:

$$C^I=<R, NR_1, NR_2, \ldots, NR_k, J_R, J_N, J_{RN}>, \quad (16)$$

where C^I is a description of the structure of the integrated information resource; R is description of the structure of the relational component, which is formed by structured data presented in the form of database tables; NR_1, NR_2, ..., NR_k are description of non-relational components of various types; J_R is a set of connections between relational elements; J_N is a set of connections between non-relational elements; J_{RN} is a set of relations between relational and non-relational elements.

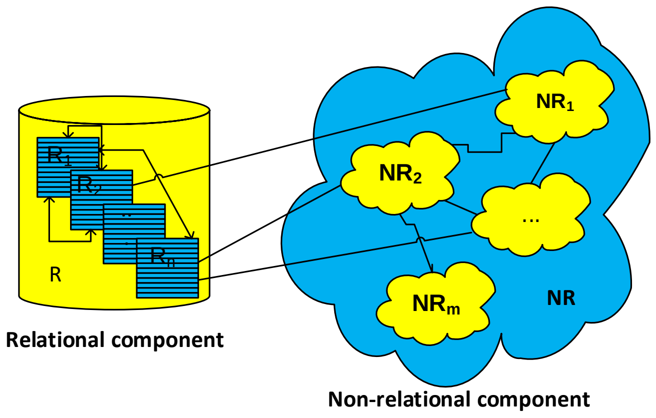

Figure 9: The general structure of the information resource of open IS [15-18]

The integration of data structures (structural integration) of data is defined as the process of a coordinated combination of structured (relational) data stored in databases and non-relational data stored in formats other than DB. The relational component in this sense is the central element of integration since modern DBs and DBMSs [15-18] provide a sufficiently wide range of opportunities for joint and coordinated processing of not only structured data, but also information resources specified by other data methods [1-2].

4.2.2. Models of structural integration

The main problems of DB integration with other types of data and directions and principles of their solution are defined in [21-23]. Typical approaches to the integration of structured relational, weakly structured/self-structured data are described by models [1-2, 15-18].

1. Integration of structured data with loosely structured (documentary, textual, spatial, temporal, visual and multimedia) data. Modern database management systems largely provide solutions to such tasks through the use of special data types (temporal types of symbolic and binary objects, generated types, etc.) and the "XML document" data type (Fig. 10). Values of these types are integrated into tables and supplement the list of elementary values in descriptions of entities and facts. The structure of relational database tables, in which loosely structured data is stored together with relational data, is described as [1-2, 15-18]: $R(A_1, A_2, \ldots, A_k, X_1, X_2, \ldots, X_m)$, where $A_1, A_2, \ldots, A_k$ are table columns that represent

scalar values of traditional and special types; $X_1, X_2, \ldots, X_m$ are columns that depict weakly structured values [15-18].

2. Integration of DB and procedural data. Such a model involves the integration of actual data stored in databases and a set of object data types together with methods that encapsulate them. In this case, each column of the table (Fig. 11a) can be represented by a pair of the form (A, M), where A is a column of the table, and M is a set of methods associated with this column. The structure of such a table is described by an expression of the form [1-2, 15-18]: $R((A_1, M_1),(A_2, M_2),\ldots,(A_k, M_k))$.

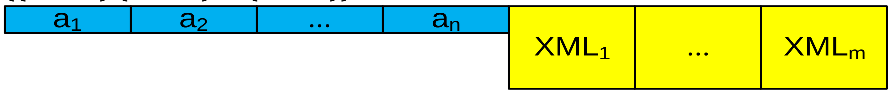

Figure 10: Structure of integrated records with XML components [15-18]

3. Integration into databases of triggers and data processing procedures. The use of such elements ensures the implementation of the concept of an active database. Such a DB, together with the values, stores a description of certain rules and actions that are performed when the state of the database changes. Tables, which include traditional data and active elements, are described by a structure model of the following type $R(A_1, A_2, \ldots, A_k, T_1, T_2, \ldots, T_m)$, where $A_1, A_2, \ldots, A_k$ are table columns that represent ordinary typed values, $T_1, T_2, \ldots, T_m$ are a set of triggers that describe the actions associated with changing the state of the table (Fig. 11b).

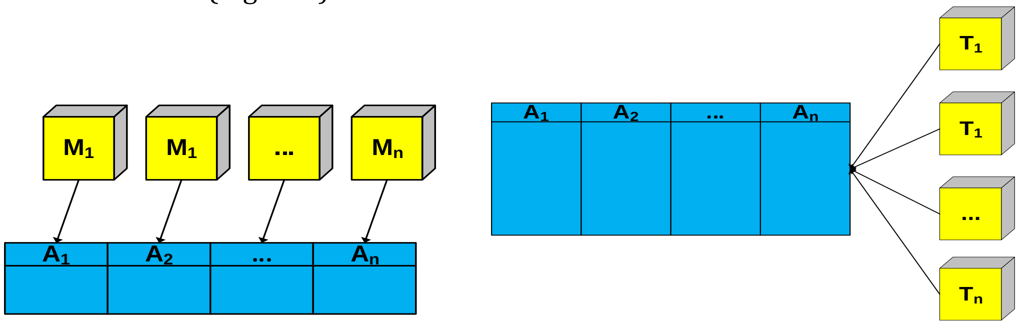

Figure 11: With procedural data and with triggers [15-18]

4. Integration of static data with streams and data queues. Stream data is a set of values that is not stored on the system media, but exists only at the time of application of this data. An example of streaming data is monitoring, stock exchange information, broadcast news in a standardized format, etc. The data flow is formed as a result of the execution of requests, forwarding or selection of data. A queue is a special type of data stream in which each unit is given an ordinal character. The structure of the data stream S at a certain time t is described by an expression of the form [1-2, 15-18]:

$$S_t=<S(R_t), S(X_t), S(S_t`), S(W_t)>, \tag{17}$$

where $S(R_t)$ is the structure of a set of values obtained as a result of selection operations from relational database tables; $S(X_t)$ is the structure of a set of data obtained as a result of selection from sets of weakly structured data; $S(S_t`)$ is the structure of a set of data obtained as a result of selection from other data streams; $S(W_t)$ is the structure of a set of data obtained from web resources. The result of the integration of static and flow components is a semi-dynamic structure (Fig. 12), which combines data stored in databases and data formed in the form of a time-varying flow [1-2].

The structure of the integrated set, which combines relational and stream data C^I, is described as $C^I=<R_t, S_t, J_{Rt}, J_{St}, J_{RSt}>$, where R_t is the structure of the relational component determined at time t; S_t is the structure of the flow component determined at time t; J_{Rt} is the scheme of connections between the elements of the relational component at the moment of time t; J_{St} is a diagram of connections between elements of the flow component at time t; J_{RSt} is the scheme of connections between the elements of the relational and flow component at the moment of time t [15-18].

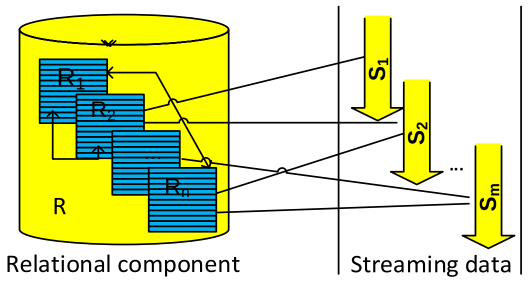

Figure 12: General scheme of integration of static data with data streams

1. Integration of information resources in DS structures and data spaces (Fig. 13) [15-18]. The significant volumes and variety of data used in the activities of large businesses and other formations require the use of special approaches to their organization and processing, which ensure their availability and efficiency. Today, DS and data space technologies are such ways. These information resources provide the ability to perceive and apply a large number of values obtained from tens or hundreds of operational DBs, due to their aggregation and fusion. Modern DSs support two methods of integration: asynchronous - "extraction-transformation-loading" (ETL) and operational (OnLine) integration of sets and streams and other units of data received from different sources. The first provides for the formation of a stable and time-invariant DS for long-term, repeated use, and the second - is the formation of an integrated resource that reflects the state of the data at a certain point in time and provides one-time access to the IR of the corporate system. DS, as a means of integrated presentation and processing of data, provides for the formation of some global data structure, which reflects the structures of local input resources. Depending on the method of integration, such a scheme can be static (in the case of using "extraction-transformation-loading" procedures) or created dynamically (for DS based on On-Line integration) [15-18].

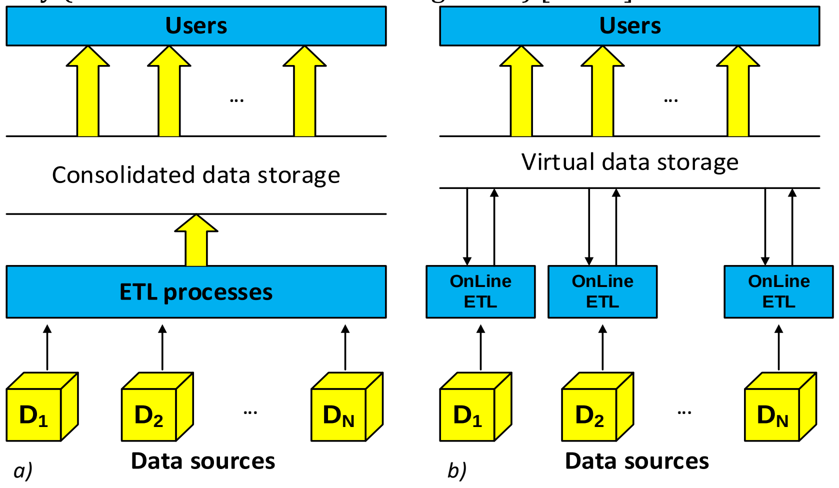

Figure 13: The general scheme of data integration in DS a) based on extraction-transformation-loading and b) based on *On*-Line integration

In both cases, the global C^G structure is presented as a union of the mappings of n local structures onto the global one, built according to a predefined procedure [1-2, 15-18]:

$$C^G = Str_1(C^G) \cup Str_2(C^G) \cup \ldots \cup Str_n(C^G), \qquad (18)$$

where C^G is the global structure of DS; $Str_i(C^G)$ is the mapping of the local structure of the ith input resource to the global structure (i=1,2, ..., n). The structure formed in this way is the result of the integration of a set of input resources and provides the possibility of their joint application, management and access.

2. Integration of sensor data and sensor networks. This relatively new direction in data integration makes it possible to realize the possibilities and advantages of cloud (network) computing [15-18]. A sensor network is a network of distributed devices, each of which is

a source of certain data [23]. Examples of such networks are a monitoring sensors network, a network of points of goods or services sale, terminals for customer service, meteorological or geo-informational distributed measuring complexes, etc. From the point of view of integration, the sensor network is a distributed database that can function both independently and is compatible with other databases as part of integrated data processing systems (Fig. 14) [1-2, 15-18].

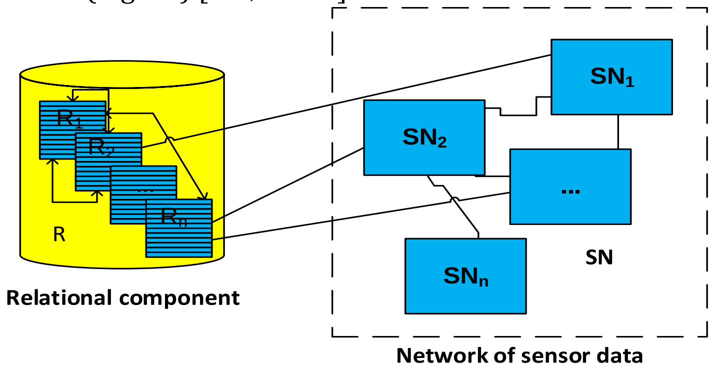

Figure 14: General scheme of integration of relational and sensory data [15-18]

A network of sensor data can include both structured objects and loosely structured data sets, or data sets without a predefined structure. In addition, sensor network data can be both static and dynamic, that is, it can be presented both in the form of sets and streams of data. In this case, the sensor network is divided into two parts - static and dynamic, between which connections and the order of interaction are defined [15-18]. The general structure of the data obtained as a result of the integration of relational and sensor network data: $C^I=<R, SN, SN_t, J_R, J_{SN}, J_{SNt}, J_{RSNt}>$, where R is the structure of the relational component; SN is the structure of the static component of the sensor network; SN_t is the structure of the dynamic component of the sensor network at time t; J_R is the scheme of connections between elements of the relational component; J_{SN} is a scheme of static connections between sensor network elements; J_{SNt} is the scheme of dynamic connections between elements of the sensor network at time t; J_{RSNt} is a scheme of connections between the elements of the relational and static and dynamic components of the sensor network at the moment of time t [15-18]. The integrated data structure formed in this way provides access and management of an integrated information resource, which includes relational structured data and sensor network data [1-2].

4.2.3. The general order of structural integration

The models of structural integration, which are described above, reflect the peculiarities of the formation, processing and application of heterogeneous content of open information systems of various directions. This approach allows to formalize and generalize methods of structuring and maintaining access to integrated information resources of open systems regardless of their content, purpose and means of implementation. The general order of integration of heterogeneous data structures is based on the following scheme (Fig. 15) [1-2, 15-18].

Step 1. Formation of a single agreed description of structured (relational) data. The structure of the relational component is part of the original structure and describes a set of tables/images from the DB of the input local resources, which form a subset of the data participating in the formation of the original integrated resource. It is described by the expression of the species

$$R^I=(DB_1.(R_1, R_2, ..., R_{k1}), DB_2.(R_1, R_2, ..., R_{k2}), ..., DB_N.(R_1, R_2, ..., R_{kN}), J^I_R), \qquad (19)$$

where R^I is the scheme of the relational component of the original integrated data set; DB_1, DB_2, ..., DB_N are databases of local input resources; N is the number of local input resources; $DB_i.(R_1, R_2, ..., R_{ki})$ is relational data structure in the database of the i^{th} input structured resource (i=1,2, ..., N), k is the number of tables or images of the input resource; J^I_R is a diagram of relationships between relational components in an integrated source dataset.

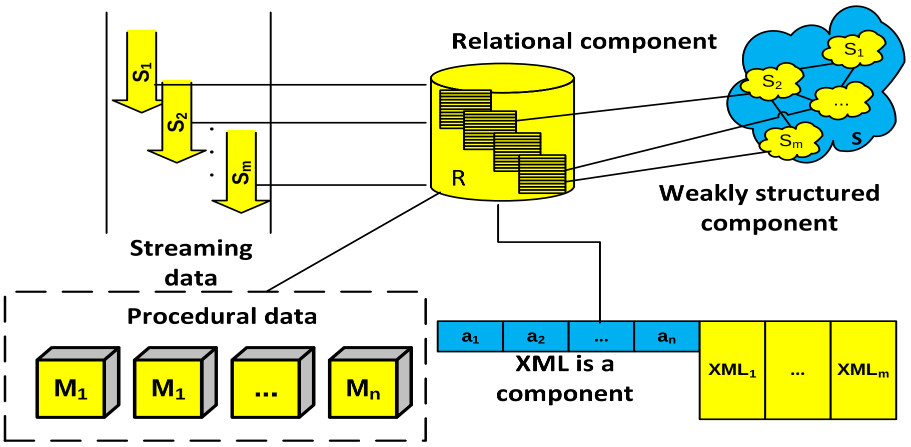

Figure 15: General order of integration of disparate data structures

Step 2. Formation of an agreed description of non-relational resources. Performing this step involves the following steps.

1. Specification of composition, properties of units of weakly structured resources and connections between them. As a result, they form a general description of resources in XML, HTML, formatted documents, etc. formats.
2. Specification of the composition and properties of resource units without a predetermined structure and the relationships between them. The result is a list and description of the structural features of text, graphic, and multimedia data sets.
3. Formation of an agreed description of procedural and active data and their connections.
4. Building a general description of the stream data structure.
5. Determination of relationships between various components of the non-relational component of the original integrated data set.

Step 3. Formation of a general description of the elements of the relational and non-relational components of the original integrated data set.

Step 4. Determination of the scheme of mutual relations between the elements of the relational/non-relational component of the original integrated data set.

Step 5. Construction of a description of a single global C^I structure as a description of the composition and properties of the original integrated data set, formed by integrating the structures of input local data sets of different natures [1-2, 15-18]: C^I=<R^I, X^I, N^I, M^I, S^I, J^I>, where C^I is a description of the structure of integrated relational resources; X^I is the description of integrated weakly structured resources; N^I is the description of integrated resources without prior description of the structure; M^I is the description of integrated active resources; S^I is description of integrated streaming resources; J^I is a description of the relationships between the elements of the integrated source data set. In this way, a single global output structure is formed, which combines the description of structured (relational) data stored in databases with the description of non-relational (weakly structured, self-structured, procedural, streaming and other) data. A formalized description of the global structure is a tool based on access, management and processing of a common integrated information resource [1-2, 15-18].

4.3. Development of formal means of specification of integrated data

The purpose of this section is to develop the general architecture, principles and order of functioning of data integration tools in open information systems based on a service-oriented approach. The main tasks arising from the goal are the following [1-2]:

- specification of general provisions and principles of service-oriented architecture of data integration tools: determination of the principles of data integration as a service of open information systems, formulation of basic concepts and definitions of the data integration service, development of the general architecture of the data integration service;
- development of the draft specification of the data integration service protocol:

a. description of the protocol status of the data integration service,
b. development of the protocol structure of the data integration service,
c. development of language tools for describing the syntax of integrated data,
d. development of linguistic means of describing the structure of integrated data,
e. development of language tools for describing the semantics of integrated data,
f. development of language means of describing additional properties of integrated data,
g. development and testing of data integration service tools.

The result of performing the assigned tasks is the specification of protocol and language tools for creating metadata that describe the syntax, semantics, and structure of data in integration processes, as an interoperable service of an intermediate level [1-2]. The fundamental difference between data integration tools and others is that they are implemented at the protocol level, not at the tool level, as is customary in many integration platforms. The chapter is written based on scientific publications and studies [1-2, 24-31].

4.4. Service-oriented data integration architecture

4.4.1. Data integration as a service of open intelligent systems

Some of today's popular approaches, principles and concepts [1-2, 32-45] are the basis for the construction of tools that implement the method of multi-level data integration.

1. The concept of Information-on-Demand. This concept is based on the data understanding as a certain asset of the business structure, which is a product of its activity and a subject of consumption by the client. The essence of this approach boils down to the following. In the age of advanced information and communication technologies, the consumer of an information product does not need to take part in its formation, storage and maintenance. As the need arises, the user turns to the relevant business structures, which form a set of information products or services by the user's request. In this way, the information asset acquires consumer qualities similar to the consumer qualities of material resources produced by the manufacturer. The implementation of such a concept involves the application of appropriate business architectural and technological solutions. One of the constituent parts of the "information on demand" dissemination process is the obtaining, matching, transforming and combining data process that forms the user requests execution result. The general task of such processes can be reduced to the integrating data task obtained from various sources and submitted according to user requirements.
2. Data-as-a-service (DaaS). The principle of presenting data as some service that the information system provides to the user is the extension of the service-oriented approach to various components of open systems. The essence of this principle lies in the possibility for the user to receive a set of data of the appropriate composition and content upon a request formulated according to predefined rules. This approach is justified in cases where creating your data repository is impossible, unavailable or unprofitable for the user. The factors that stimulate the development of data services are, first of all, the additional costs borne by the data consumer during the independent formation of IR, developed network infrastructure and a significant selection of public resources available within the limits of standardized technologies. This allows you to minimize the costs of technical means of data storage and transmission, costs of software and technological means of processing data repositories, costs of replenishment, updating and updating of data, etc. Obtaining data as a service provided by open resources for public use allows the user to significantly reduce his participation in solving the problems of information support of his activity. An important component of the complex problems that are solved within the framework of the implementation of the data service is the problems associated with obtaining data from disparate sources and combining them in the final product that the consumer receives.
3. Content Management Interoperable Service (CMIS). The CMIS concept provides for the possibility of free user access to content located in various data repositories. For this, a

special service interface is used, which is independent of the platform and which is configured according to the needs of a specific user. The data model and services of CMIS are independent of data access protocols and allow to implementation of the concept of a virtual information environment that reflects the user's views on SA. CMIS uses separate principles of data integration to form the final result of data search in different repositories, their selection and the combination of data obtained from disparate sources. However, using the CMIS approach only provides the user with access to the data and receives it on an "as is" (as is) basis and does not provide procedures for transforming the selection results to create a coherent, consistent data set. Therefore, CMIS services are, to a greater extent, a means of data virtualization than their integration.

4. Information resource description technology - Resource Description Framework (RDF), developed by the W3 consortium. The RDF toolkit is one of the promising tools for solving problems related to the description of semantics and pragmatics of data. This technology is based on a special RDF data model, which represents a set of facts and semantic relationships between them, presented in the RDF document format. The notation used to build RDF documents is based on XML principles, which makes them interoperable concerning platforms and environments and available for processing using various technologies. The concept of FDF assumes that a data model is a summary of information and knowledge about a certain information resource, and an RDF document, which is built according to a defined notation, is a means of presenting this information. The basic block of the RDF data model is a statement, which defines the subject, predicate, and object. A subject is a specific information resource, a predicate is a named property of this resource, and an object is the value of this property.
5. The XMI (XML Metadata Interchange) metadata exchange standard is a normative document, the specification of which was approved by the W3 consortium in 1999. According to its concept, the XML format is used for the formation, movement and exchange of metadata in IS environments. This makes it possible to apply a single, unified way of presenting metadata for various environments and applications, such as DB, DS, web resources, UML metamodels, data repositories, etc., which is important for implementation in open systems environments.

The main principles of XMI application are [1-2, 46-51]:

- metadata, regardless of nature, content and subject, are submitted in XML format, for which XMI specification defines the procedure for typing the corresponding XML documents;
- the standard defines methods of displaying metadata generated according to other standards in XML format, which enables the generation of correct descriptions created based on meta-models of various formats;
- the XMI standard provides for the possibility of exchanging metadata both in the form of data streams and the form of standardized format files (XML documents);
- IT, which is XMI standard basis, is supported by leading manufacturers of application and data management tools, including Oracle, IBM, Rational, and Sybase.

All the approaches described above, to a certain extent, have an impact on data integration processes. Therefore, in the integration service concept formulation, the features of each of them were used [52-63].

4.4.2. Architecture of the data integration service

The data integration service implements a common unified data access model through a special layer of standardized reusable data abstractions [1-2]. This, in turn, provides opportunities to obtain complete, reliable and timely generalized information without the use of complex mechanisms and additional resources or financial costs. The general architecture of service-oriented data integration (Fig. 16) combines heterogeneous information resources through data integration services with applications and different categories of users [1-2]. At the same time, it is taken into account that the general information resource contains data sets formed by various methods and technologies. To create the possibility of sharing the values concentrated in such sets as part of the information system, an additional level is created - the level of data integration services. At this level, a unified description of the syntax, structure and semantics of each set in the form of metadata, as well as means of manipulating them, is concentrated. Users and applications refer not directly to the data, but to the metadata that describes it. The result of metadata processing and interpretation is a generalized query, composed, in turn, of several detailed queries that formulate the order of access to the data sets located in the relevant repositories, and the order of creation of the resulting set of values that contain the results of the query. In this way, the use of data by the end-user of the information system becomes independent of the form and order of their presentation in the storage environment.

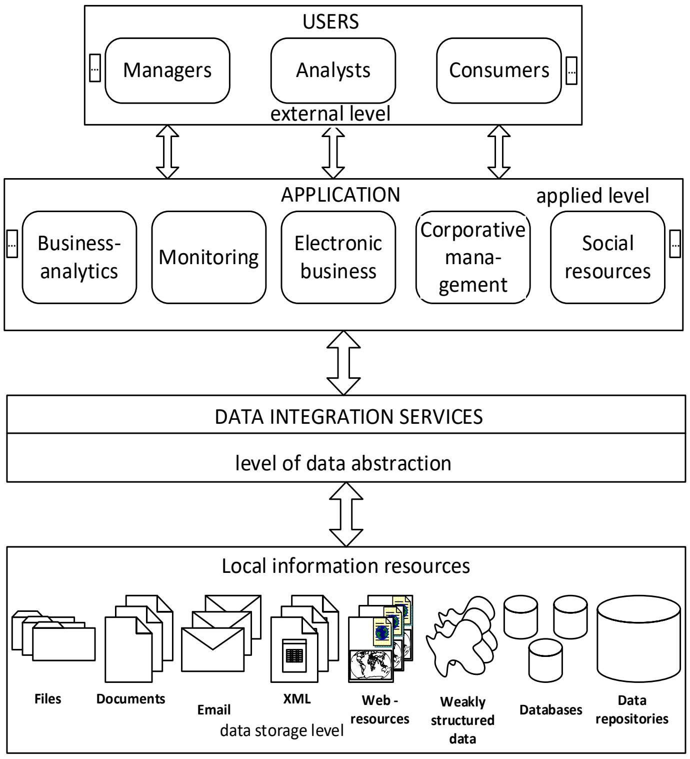

Figure 16: Architecture of service-oriented data integration

The functioning of data integration services as part of the information system is built according to the following scheme (Fig. 17) [1-2]. The basis of the data integration service is the data integration server, which is an intermediate link between application tools and data repositories. Such a server is created according to the principles of intercomponent support (middleware) as part of a set of active server units and means of their maintenance and a global meta schema of the information resource. Requests for the required set of values are sent from user applications to the data integration server in the environment-specific format. The request received by the server goes through the stages of analysis and interpretation, the task of which is to generate an integrated meta schema of the data set of the request results. An integrated meta schema combines a unified description of all data units that should be part of the resulting set and is an input to the data repository server. The repository server, based on the meta schema received at the entrance, performs access to the necessary components of the data storage environment based on the corresponding services, the selection of specified values and data units, and the formation of a single set of integrated data. The results of the actions of the repository server are transferred to the integration server, combined with their meta schema and transferred to the application from which the request came. The basis of such a service is a set of specialized tools that make up the data integration service protocol. The advantages of using protocol means of data integration over instrumental ones arise, first of all, due to such characteristic properties [1-2].

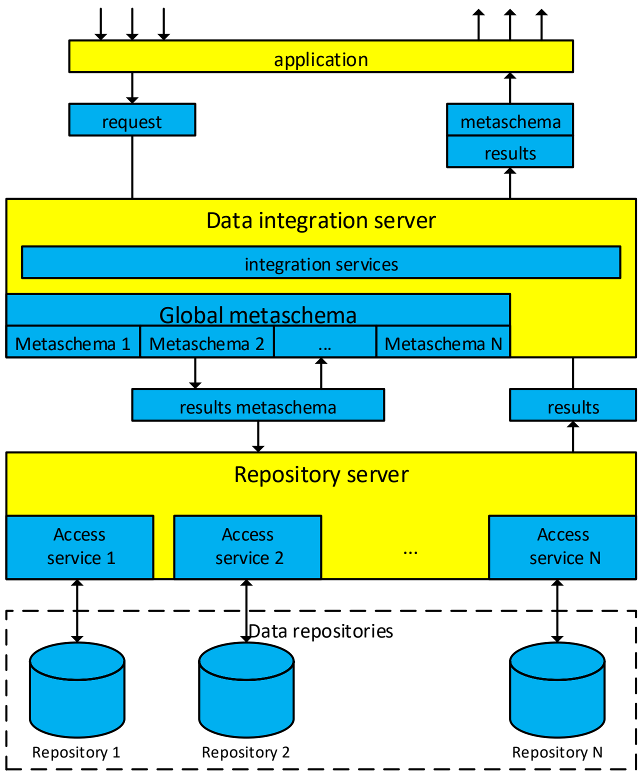

Figure 17: Scheme of data integration service operation

1. Unification - the use of protocol-level tools makes it possible to reduce the processes of data integration to the manipulation of typical concepts, objects, properties and procedures for their processing.
2. Interoperability - provides the possibility of joint use of means of maintaining the application protocol of data integration with any other means of data processing.
3. Mobility - the implementation of protocol means of data integration at the application level makes them independent of the specifics of the platforms and the implementation environment, which ensures the possibility of their free movement.
4. Processing formats and procedures standardization - XML use as the basis of the language means of describing objects and processing metadata makes it possible to process the necessary resources by standard means and according to standardized procedures.
5. Compliance with the principles of SOA - provides easy access and use of data integration protocol tools for a wide range of information system users.
6. Implementation ease – such means use of integration does not require restructuring business processes, platforms or other means of maintaining the open IS environment, as it involves the additional, relatively autonomous intermediate layer formation of data abstraction.
7. Insignificant cost - the absence of the need to rebuild information systems, and develop, purchase and implement complex and expensive data integration tools and platforms makes projects based on protocol tools relatively inexpensive.

4.5. Data integration service protocol

The Data Integration Service Protocol (DISP) is an application-level protocol that defines the methods, order and means of implementing the data integration service at the request of the user of the open information system. The main purpose of using the protocol is the organization of the service for the user of the open information system, which forms the data set he needs by integrating the data located in a set of disparate local information resources. The protocol does not depend on protocols of lower (presentation and session) levels and provides for user interaction with open information system services using standardized interfaces. The goals achieved by the protocol are as follows [1-2]:

- creation of a platform for building a data integration service in environments of open information systems with heterogeneous distributed information resources;
- maintenance of interaction with standard application tools of the user of the open information system;
- perception and interpretation of requests of the user of the open information system to receive the appropriate information product or service;
- ensuring user access to disparate local information resources set through integration;
- formation of a global data set based on local information resources through integration;
- transfer of the results of the request to the user of the open information system.

The goals of the protocol are not:

- determination of ways, methods and means and formation of local information resources;
- determination of the composition and characteristics of technological means of data processing in local repositories;
- administration of repositories in which local information resources are concentrated;
- implementation of data protection and security measures in the IS environment.

5. Conclusions

Based on the performed analysis and generalizations, the main methods of building information resources of business analytics systems are determined - homogenization, distribution, and integration. The approach based on data integration is the most appropriate to the features of business intelligence systems. Together with the other resources integration of business intelligence systems, data integration forms a single coordinated set of actions for the design, construction and support of business intelligence systems.

Based on the conducted analysis, it is possible to conclude that there are several relevant today's interrelated problems in the e-commerce field.

1. The problem of forming a high-quality information resource of business analytics systems that is relevant to the system goals, adequate to its tasks and appropriate to the needs and requirements of users.
2. The problem of heterogeneous form, content and properties integration of input information resources into a single agreed common resource of business analytics systems.
3. The problem of developing and substantiating a single generalized theoretical concept of data integration for the unified effective methods, tools and modern technologies creation to solve the problem of forming information resources of business analytics systems in various industries and fields of application.

The development of methods of multi-level data integration in business analytics systems allows us to draw the following conclusions:

- It is advisable to base the method on a multi-level data model that takes into account various aspects of this data (syntax, semantics, meaning);
- The key point of the integration process is its implementation based on data meta-schemas, which made it possible to reduce access to the actual data and reduce the time required for integration;
- The general process of integration is carried out as a set of sub-processes of integration of values, syntax and semantics of data;
- The use of ontologies in the processes of semantic data integration made it possible to carry out integration at the level of data content, to achieve the same interpretation of data by both people and machines.

In the process of developing knowledge presentation methods for intelligent business analytics systems, the algebraic theory of types was laid as its basis. An algebraic system has been built in which there is an unambiguous mapping between ontology entities and abstract data types. The development of presentation methods and the architecture of intelligent networks of business processes allows us to draw the following conclusions:

- IS is based on the ontology of business processes.
- Given the changing nature of SA and the incomplete knowledge of the ontology developer, it is necessary to use ontologies that can adapt to changes.
- Intelligent networks of business processes should be presented as a set of interacting executable ontological models.

As a result of the work, a specification of language tools was developed for creating metadata that describes the syntax, semantics, and structure of data in integration processes, as an interoperable service of an intermediate level. The advantages of linguistic means of data integration over instrumental ones are due to the following factors.

1. Unification - the use of protocol-level language tools makes it possible to reduce the processes of data integration to the manipulation of typical concepts, objects, properties and procedures for their processing.
2. Interoperability – provides the possibility of joint use of means of maintaining the application protocol of data integration with any other means of data processing.
3. Mobility – the implementation of protocol means of data integration at the application level makes them independent of the specifics of the platforms and the implementation environment, which ensures the possibility of their free movement.
4. Standardization of processing formats and procedures – the use of XML as the basis of the language means of describing objects and processing metadata makes it possible to process the necessary resources by standard means and according to standardized procedures.
5. Compliance with the principles of SOA – provides easy access and use of data integration protocol tools for a wide range of information system users.
6. Ease of implementation – such means use of integration does not require restructuring of BP, platforms or other means of maintaining the open information system environment, as it involves the formation of an additional, relatively autonomous intermediate layer of data abstraction.
7. Insignificant cost – the need absence to rebuild information systems, and develop, purchase and implement complex and expensive data integration tools and platforms makes projects based on protocol tools relatively inexpensive.

References

[1] A. Berko, I. Pelekh, L. Chyrun, M. Bublyk, I. Bobyk, Y. Matseliukh, L. Chyrun, Application of ontologies and meta-models for dynamic integration of weakly structured data, in: IEEE 3rd International Conference on Data Stream Mining & Processing, 2020, August, pp. 432-437.

[2] V. Vysotska, S. Holoshchuk, R. Holoshchuk, A Comparative Analysis for English and Ukrainian Texts Processing Based on Semantics and Syntax Approach, CEUR Workshop Proceedings 2870 (2021) 311-356.

[3] I. Kushniretska, O. Kushniretska, A. Berko, Designing of structural ontological data systems model for mash-up integration process, Applied Computer Science 11(1) (2015) 39-50.

[4] A. Berko, et al., Models and Methods for E-Commerce Systems Designing in the Global Economy Development Conditions Based on Mealy and Moore Machines, CEUR Workshop Proceedings 2870 (2021) 1574-1593.

[5] X. L. Dong, F. Naumann, Data fusion: resolving data conflicts for integration, Proceedings of the VLDB Endowment 2(2) (2009) 1654-1655.

[6] X. L. Dong, L. Berti-Equille, D. Srivastava, Integrating conflicting data: the role of source dependence, Proceedings of the VLDB Endowment 2(1) (2009) 550-561.

[7] J. P. Dittrich, M. A. V. Salles, iDM: A unified and versatile data model for personal dataspace management, in: Proceedings of the 32nd international conference on Very large data bases, 2006, September, pp. 367-378.

[8] F. K. Bruder, R. Hagen, T. Rölle, M. S. Weiser, T. Fäcke, From the surface to volume: concepts for the next generation of optical–holographic data-storage materials, Angewandte Chemie International Edition 50(20) (2011) 4552-4573.
[9] M. L. Brodie, Data integration at scale: From relational data integration to information ecosystems, in: 24th IEEE International Conference on Advanced Information Networking and Applications, 2010, April, pp. 2-3.
[10] M. Stonebraker, U. Çetintemel, S. Zdonik, The 8 requirements of real-time stream processing, ACM Sigmod Record 34(4) (2005) 42-47.
[11] IBM Rational Unified Process, Rational Process Library, URL: https://www.ibm.com/support/pages/rational-unified-process-rup-plug-ins-rational-method-composer-751.
[12] IEEE Guide for Developing System Requirements Specifications: IEEE Std 1233a-1998. Institute of Electrical and Electronics Engineers, Inc., 1998.
[13] IEEE Standard Glossary of Software Engineering Terminology: IEEE STD 610.12-1990. Institute of Electrical and Electronics Engineers, Inc., 1990.
[14] P. Zdebskyi, A. Berko, V. Vysotska, Investigation of Transitivity Relation in Natural Language Inference, CEUR Workshop Proceedings 3396 (2023) 334-345.
[15] P., Zdebskyi, et al., Intelligent System for Semantically Similar Sentences Identification and Generation Based on Machine Learning Methods, CEUR Workshop Proceedings 2604 (2020) 317-346.
[16] V. Lytvyn, S. Kubinska, A. Berko, T. Shestakevych, L. Demkiv, Y. Shcherbyna, Peculiarities of Generation of Semantics of Natural Language Speech by Helping Unlimited and Context-Dependent Grammar, CEUR Workshop Proceedings 2604 (2020) 536-551.
[17] V. Lytvyn, et al., Methods and Models of Intellectual Processing of Texts for Building Ontologies of Software for Medical Terms Identification in Content Classification, CEUR Workshop Proceedings 2488 (2019). 354-368.
[18] M. Fedorov, et al., Decision Support System for Formation and Implementing Orders Based on Cross Programming and Cloud Computing, CEUR Workshop Proceedings 2917 (2021) 714-748.
[19] E. E. Schadt, M. D. Linderman, J. Sorenson, L. Lee, G. P. Nolan, Computational solutions to large-scale data management and analysis, Nature reviews genetics 11(9) (2010) 647-657.
[20] J. Hamilton, A Conversation with Pat Selinger: Leading the way to manage the world's information, Queue 3(3) (2005) 18-28.
[21] P. Bernstein, et al., The asilomar report on database research, ACM Sigmod record 27(4) (1998) 74-80.
[22] R. Agrawal, et al., The claremont report on database research, ACM Sigmod Record 37(3) (2008) 9-19.
[23] S. Abiteboul, et al., The Lowell database research self-assessment, Communications of the ACM 48(5) (2005) 111-118.
[24] Y. Burov, V. Vysotska, V. Lytvyn, L. Chyrun, Software Based on Ontological Tasks Models, in; International Scientific Conference on Intellectual Systems of Decision Making and Problem of Computational Intelligence, 2022, May, pp. 608-638. Cham: Springer Int. Publishing.
[25] Y. Burov, Knowledge Based Situation Awareness Process Based on Ontologies, CEUR Workshop Proceedings 2870 (2021) 413-423.
[26] I. Pelekh, A. Berko, V. Andrunyk, L. Chyrun, I. Dyyak, Design of a system for dynamic integration of weakly structured data based on mash-up technology, in: IEEE Third International Conference on Data Stream Mining & Processing, 2020, August, pp. 420-425.
[27] A. Berko, I. Pelekh, L. Chyrun, I. Dyyak, Information resources analysis system of dynamic integration semi-structured data in a web environment, in: IEEE Third International Conference on Data Stream Mining & Processing (DSMP), 2020, August, pp. 414-419.
[28] A. Berko, Y. Matseliukh, Y. Ivaniv, L. Chyrun, V. Schuchmann, The text classification based on Big Data analysis for keyword definition using stemming, in: IEEE 16th International Conference on Computer Sciences and Information Technologies (CSIT), Vol. 1, 2021, September, pp. 184-188.

[29] O. Bisikalo, O. Kovtun, V. Kovtun, Neural Network Concept of Ukrainian-Language Text Embedding, in: 13th International Conference on Advanced Computer Information Technologies (ACIT), 2023, September, pp. 566-569. IEEE.

[30] V. Vysotska, P. Pukach, V. Lytvyn, D. Uhryn, Y. Ushenko, Z. Hu, Intelligent Analysis of Ukrainian-language Tweets for Public Opinion Research based on NLP Methods and Machine Learning Technology, International Journal of Modern Education and Computer Science(IJMECS) 15(3) (2023) 70-93,. DOI:10.5815/ijmecs.2023.03.06.

[31] V. Vysotska, A. Berko, V. Lytvyn, P. Kravets, L. Dzyubyk, Y. Bardachov, S. Vyshemyrska, Information Resource Management Technology Based on Fuzzy Logic, in: International Scientific Conference on Intellectual Systems of Decision Making and Problem of Computational Intelligence, 2020, May, pp. 164-182. Cham: Springer Int. Publishing.

[32] Z. Liu, Z. Li, A blockchain-based framework of cross-border e-commerce supply chain, International journal of information management 52 (2020) 102059.

[33] J. P. Cabrera-Sánchez, I. Ramos-de-Luna, E. Carvajal-Trujillo, Á. F. Villarejo-Ramos, Online recommendation systems: Factors influencing use in e-commerce, Sustainability 12(21) (2020) 8888.

[34] M. Li, S. Shao, Q. Ye, G. Xu, G. Q. Huang, Blockchain-enabled logistics finance execution platform for capital-constrained E-commerce retail, Robotics and Computer-Integrated Manufacturing 65 (2020) 101962.

[35] L. Li, J. Zhang, Research and analysis of an enterprise E-commerce marketing system under the big data environment, Journal of Organizational and End User Computing (JOEUC) 33(6) (2021) 1-19.

[36] D. Zhang, L. G. Pee, L. Cui, Artificial intelligence in E-commerce fulfillment: A case study of resource orchestration at Alibaba's Smart Warehouse, International Journal of Information Management 57 (2021) 102304.

[37] F. Schiavone, D. Mancini, D. Leone, D. Lavorato, Digital business models and ridesharing for value co-creation in healthcare: A multi-stakeholder ecosystem analysis, Technological Forecasting and Social Change 166 (2021) 120647.

[38] H. Guo, Y. Liu, X. Shi, K. Z. Chen, The role of e-commerce in the urban food system under COVID-19: Lessons from China, China Agricultural Economic Review 13(2) (2021) 436-455.

[39] R. E. Bawack, S. F. Wamba, K. D. A. Carillo, S. Akter, Artificial intelligence in E-Commerce: a bibliometric study and literature review, Electronic markets 32(1) (2022) 297-338.

[40] R. V. Karthik, S. Ganapathy, A fuzzy recommendation system for predicting the customers interests using sentiment analysis and ontology in e-commerce, Applied Soft Computing 108 (2021) 107396.

[41] A. Rosário, R. Raimundo, Consumer marketing strategy and E-commerce in the last decade: a literature review, Journal of theoretical and applied electronic commerce research 16(7) (2021) 3003-3024.

[42] S. Tyshko, O. Lavrut, V. Vysotska, O. Markiv, O. Zabula, Y. Chernichenko, T. Lavrut, Compensatory Method for Measuring Phase Shift Using Signals Bisemiperiodic Conversion in Diagnostic Intelligence Systems, CEUR Workshop Proceedings 3387 (2022) 144-154.

[43] H. Khudov, S. Yarosh, O. Droban, O. Lavrut, Y. Hulak, I. Porokhnia, S. Yarovyi, A. Rogulia, I. Yuzova, R. Khudov., Development of a direct penetrating signal compensator in a distributed reception channel of a surveillance radar, Eastern-European Journal of Enterprise Technologies 2(9 (110), 16–26. DOI: https://doi.org/10.15587/1729-4061.2021.

[44] Z. Zhu, Y. Bai, W., Dai, D. Liu, Y. Hu, Quality of e-commerce agricultural products and the safety of the ecological environment of the origin based on 5G Internet of Things technology, Environmental Technology & Innovation 22 (2021) 101462.

[45] C. Xie, J. Yu, S. S. Huang, J. Zhang, Tourism e-commerce live streaming: Identifying and testing a value-based marketing framework from the live streamer perspective, Tourism management 91 (2022) 104513.

[46] P. Kryndach, V. Vysotska, S. Chyrun, L. Chyrun, S. Goloshchuk, R. Holoshchuk, Analysis of Semantic Relationships in Ukrainian Text Content Based on Word2Vec and Machine Learning, in: Proceedings of IEEE 18th International conference on Computer science and

information technologies, CSIT-2023, Lviv, 19-21 October 2023 p. DOI: 10.1109/CSIT61576.2023.10324074.

[47] V. Lytvyn, V. Danylyk, M. Bublyk, L. Chyrun, V. Panasyuk, O. Korolenko, The lexical innovations identification in English-languagee eurointegration discourse for the goods analysis by comments in e-commerce resources, in: Proceedings of IEEE 16th International conference on Computer science and information technologies, Lviv, 2021, pp.85–97.

[48] D. Uhryn, V. Andrunyk, L. Chyrun, N. Antonyuk, I. Dyyak, O. Naum, Service-Oriented Architecture Development as an Integrating Platform in the Tourist Area, CEUR Workshop Proceedings 2631 (2020) 221-236.

[49] O. Duda, V. Pasichnyk, H. Lypak, N. Veretennikova, N. Kunanets, O. Matsiuk, V. Mudrokha, Formation of Integrated Repositories of Social and Communication Data by Consolidating the Resources of Museums, Libraries and Archives in Smart Cities Projects, CEUR Workshop Proceedings 2870 (2021) 1420-1430.

[50] I. Galushka, S. Shcherbak, Devising a mathematical model for pattern-based enterprise data integration, Eastern-European Journal of Enterprise Technologies 2(9) (2015) 59-64.

[51] V. Hryhorovych, Calculation of the Semantic Distance between Ontology Concepts: Taking into Account Critical Nodes, CEUR Workshop Proceedings 3396 (2023) 206-216.

[52] S. Albota, Creating a Model of War and Pandemic Apprehension: Textual Semantic Analysis, CEUR Workshop Proceedings 3396 (2023) 228-243.

[53] A.Taran, Corpus Analysis of Word Semantics, CEUR Workshop Proceedings 3396 (2023) 373-384.

[54] S. Olizarenko, V. Radchenko, Method for determining the semantic similarity of arbitrary length texts using the transformers models. Advanced Information Systems 5(2) (2021) 126–130. https://doi.org/10.20998/2522-9052.2021.2.18.

[55] T. Basyuk, A. Vasilyuk, V. Lytvyn, Mathematical model of semantic search and search optimization, CEUR Workshop Proceedings 2362 (2019) 96–105.

[56] S. Orekhov, Advanced Method of Synthesis of Semantic Kernel of E-content, CEUR Workshop Proceedings 3312 (2022) 87-97.

[57] N. Sharonova, I. Kyrychenko, I. Gruzdo, G. Tereshchenko, Generalized Semantic Analysis Algorithm of Natural Language Texts for Various Functional Style Types, CEUR Workshop Proceedings 3171 (2022) 16-26.

[58] S. Orekhov, H. Malyhon, Method for Synthesizing the Semantic Kernel of Web Content, CEUR Workshop Proceedings 3171 (2022) 127-137.

[59] L. Savytska, M. T. Sübay, N. Vnukova, I. Bezugla, V. Pyvovarov, Word2Vec Model Analysis for Semantic and Morphologic Similarities in Turkish Words, CEUR Workshop Proceedings 3171 (2022) 161-176.

[60] S. Albota, Modelling the Impact of the Pandemic on Online Communication: Textual Semantic Analysis, CEUR Workshop Proceedings 3171 (2022) 471-486.

[61] N. Romanyshyn, Stylistic and Conceptual Function of Epithet in Poetic Discourse: an Experience of "Tropes" Text Semantics Analysis Program Application, CEUR Workshop Proceedings 3171 (2022) 514-535.

[62] V. Shynkarenko, L. Zhuchyi, Semantic Checking of Different Type Information Sources About Permitted Speeds in Railway Transport, CEUR Workshop Proceedings 3171 (2022) 711-723.

[63] V. Hryhorovych, Analysis of Scientific Texts by Semantic Inverse-Additive Metrics for Ontology Concepts, CEUR Workshop Proceedings 3171 (2022) 801-816.