Linked Open Data technologies for authoring and publishing text documents

Cherkashin E.A.1,2,3, Shigarov A.O.1,3, Orlova I.V. 2, Mikhailov I.S.4

1V.M. Matrosov's institute of system dynamics and control theory SB RAS   
 Lermontov str. 134, Irkutsk, 664033, Russia  
2National research Irkutsk state technical university, Lermontov str. 83,  
 Irkutsk, 664033, Russia  
3Иркутский scientific center SB RAS, Lermontov str. 134, Irkutsk, 664033, Russia  
4Limnological Institute SB RAS, Ulan-Bator str. 3, Irkutsk, 664033, Russia

[eugeneai@icc.ru, shigarov@icc.ru](mailto:eugeneai@icc.ru)

**Abstract.** The problems of automating the creation of text documents with the active use of declarative tools and technologies of Linked Open Data is discussed. Software tools to compose HTML5 documents from various sources and present the document in a specified layout, such as a template or form, is being developed. The sources are web pages and the results of text processing on the server or client side (web browser) processing results. Several examples of the application of the developed software are considered.

Keywords: Linked Open Data, automated composing a document, semantic markup of document

1. Introduction

Linked Open Data (LOD) [1] technology has been suggested by W3C consortium to represent the semantic information in the data published in a way that provides not only the possibility of its processing with software agents (Semantic Web), but also to link all available information into a single semantic graph using relations and global universal identifiers (URIs) of theory resources. The descriptive capabilities of semantic web technologies, HTML5 document publishing tools and LOD technologies form an infrastructural basis of authoring and publishing documents. The document is constructed out of individual parts (text and images) that are loaded from other servers with links to relevant resources. The resources represent both the static content and the results of a data conversion algorithm execution generating text content. The LOD provide a logical markup for the information presented in a document, informative basis for the different variants of visual representation and interpretation, logical connections with other documents, export information into other documents, procedural processing, etc. An important advantage of LOD usage in creation of information environments is a weakening of the requirements to published information warehouses: the document itself is the formalized data warehouse. In some extent, this allows redirecting time spent on designing the database structure for a storage of partially formalized documents to the process of solving a substantive problem: the user (developer) markups the text data with semantic meaning.

The aim of the study is to design an approach to the development of software performing the functions of the authoring and typesetting of documents marked up according to LOD principles. To achieve the goal, a set of server functions and client (web-browser) JavaScript routines has been developed that allows the user to create documents with the web browser. The functions support uploading of text fragments from other documents and their generation with server and client scripts. The resulting document is placed in the data warehouse, printed or downloaded to the user workstation as an HTML file. JavaScript implements various document views, e.g., as input forms, a conversion of parts of the text, e.g., to lean nouns and substitute pronouns, and other content generation.

The created editing tools are the basis of developed digital archives of documents intended for solving problems of creating documentation of university courses, representing research results of DNA/RNA sequencing. The use of LOD data formats, HTML processing tools of web browser and developed technologies allows one to solve a wide class of problems of formation of documentation, ranging from the formation of a meaningful part of the texts through the presentation of stylistic characteristics of the texts and the integration of logical markup into a global data access services.

The development of the tools for document representation on the basis of LOD aimed at the formation of the prototype of a global environment that supports automation of data transfer between documents of several companies, and create, ideally, a distributed document processing with the principles of social networks. The task is relevant, especially in some small business environments with no well-established information flows and relationships between documents. This is mainly due to the dynamism and diversity of the small business. Use of large-scale workflow automation software in the environments of such kind requires large financial expenses for the purchase and/or continuous improvement of the software product. Moreover, in companies with a deployed document flow automation, there is almost always a wide class of documents that are not integrated in the main document flow (e.g., memos, statements), documents with difficult-to-formalize information (e.g., job descriptions) and the documents received by the organizations from the outside having no contain logical markup (e.g., industry standards).

This work continues the research outlined in [2] in implementing client services operating on the platform of the web browser.

1. Technology of document authoring based on the LOD

The developed software for typesetting of text documents are based on technologies, which implement the HTML5 standard available in modern web browsers. The document is a web page that is composed out of different parts by means of the JavaScript routines functioning within the web browser. To make changes in the content of the page, built-in browser editing tools of web pages (contentEditable attribute) are used. The tools are managed by the special libraries (Medium Editor, https://yabwe.github.io/medium-editor/), which extend the basic functions of the built-in editor with document structure conversion and LOD markup.

The text of the document is being marked up with RDFa attributes. The semantic markup is the logical structure of the document, the domain relations between the entities presented in the document, for example, the distinct [parties] are described in an agreement, subject of agreement; formal data; the “part-whole” relationships sectioning the text document. In addition to the semantic markup, command structures are embedded in the text of a document. Such structures are used for algorithm activation of content conversion, for example, some structures activate the inclusion of parts of other document in the current one, other instruction organize grammatical transformations of sentences. The command structures are recognized more easily than the structures of semantic markup. The command structures interpretation mainly aimed at algorithmic text processing.

As the basis for the semantic markup of the documents, the results of the open project Dokieli [3] is used. The project focuses on implementation of HTML5 text-editing tools for authoring a scientific publication by several co-authors. Dokieli allow to markup text with tags RDFa, to copy a document in its own repository and make an annotation of the text with readers' and authors' comments. In the markup of a document, a number of standardized ontologies are used.

**Open Annotation (**oa**).** The ontology standard is accepted in 2017, its main purpose is to describe the content (annotation) describing other content. Browser bookmarks are examples of such content. An oa resource (an bookmark) is in two major relations to other resources: oa:hasTarget, oa:hasBody. The first relation refers to the annotated content with a corresponding resource URIs, and the second one, which can be multiple, refers to an annotation resource, for example, a text description of the annotated resource or an associated comment. The edited document is an annotation to the target document, i.e., at the beginning the documents annotates itself, and after the completion of the final version of the document, the annotation describes the generated immutable copy, e.g., the PDF file of the document.

**Friend-of-a-friend (**foaf**)** describes information about agents: physical and legal entities, together with software agents. This ontology is widely used to represent relations between entities (agents) in a social network.

**Provenence (**prov**)** provides a vocabulary of terms and relations describing the origin of the information. This ontology is a convenient way to refer the document's source parts originated from other documents. The prov ontology is a basis of informational flows descriptions in documents and their mutual relationships.

**Dublin Core (**dc**)** is used to describe the elements of authored annotations, for example, section headings, document title, document types, various descriptions contained in the document. In contrast to oa this ontology describes the documents mainly at metainformation level.

**DBPedia resource (**dbr**)** is the namespace of the objects (resources) of Wikipedia. Ontology dbr is a convenient way to refer to specific instances of objects and concrete classes, for example, the passport of a person (dbr:Passport), a city, etc.

**Schema.org (schema)** represents the objects that are recognized by the site scanning agents of Google, Yandex, Yahoo etc. This ontology is used in the case if none of the above specialized ontological have a relevant term or a relationship denotation. This ontology contains useful relationships like “part-whole” (schema:hasPart), “result of a creative activity” (schema:CreativeWork). The relations are used to describe the sectioning of the document, i.e., its hierarchical structure.

In addition to the above-mentioned ontologies, the standard ontologies (RDF, RDFs, RDFa, XSD) and ontologies from the NEPOMUK project (https://userbase.kde.org/Nepomuk) describing objects stored in the index of full-text digital archives are used as well. They allow one to describe meta-information about the creative work of various kind in details, e.g., file, image, document, document section; and specify the type of document, text content, identification of works, etc. In our research, there was a labor-intensive stage devoted to the search ontologies to represent documents by subject areas. The resource Linked Open Vocabularies (http://lov.okfn.org/dataset/lov/) happened to be very useful reference book as well as support of the community of researchers, which develop this resource.

The architecture of the system is presented in Fig. 1 (all relations shown in the figure are bi-directional). The page is downloaded from a server where it is composed from a content stored in a database or a file system, results of an algorithmic data conversion of other sources, and elements of a template that includes the user view interface elements and interpretation modules on the displayed page. In the formation of the final form of the document, interpretation modules request the required data and resources on the source server or other Internet servers.

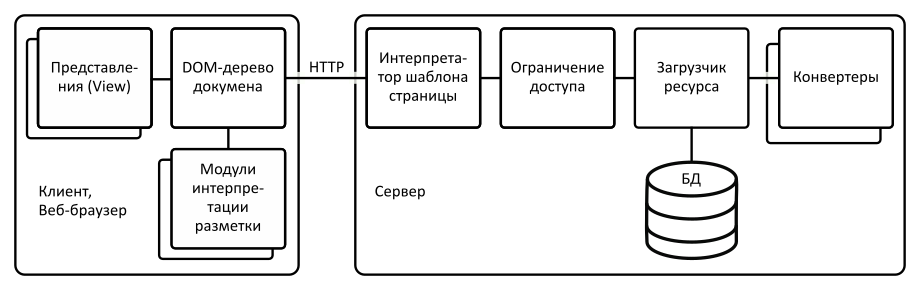


Fig. 1. The general architecture of the document publishing system

Interpretation modules of the semantic markup are implemented with client-side JavaScript. They are executed as soon as the main content is loaded by the browser. The modules scan the tree structure of the document, recognizing the conditions for their activation. If the scan was successful, the body of the module is executed, changing of the document content (tree structure). The document composition finalizes as soon as all conditions were met and all command activated.

A database of the system is a set of tools for storing and access providing to multi-format data. The body of the document loaded in the first place is stored in XML files in the file system on the server. The file system is wrapped with a document versions tracking level to monitor the development of the document in time and create integrated backup copies. This level is quite easy to be implemented using modern version control systems like GIT or Subversion. The XML format allows to represent the logical structure of the main document body, and to store it in the document.

In order to organize standard SPARQL data source access and full-text search, storage components for the logical layer and related data as a graph of triples have been developed. ClioPatria [4] and Jena are the component implementations. The first system is interesting because it is implemented entirely with the programming languages Prolog and C. It supports several formats of compact storage of data, and tight integration of triple data with the runtime and language environment of Prolog. A regulated access to the triples is supported using so called Pengines protocol [5]. There are a number of implementation libraries for the protocols for popular programming languages, including client-side JavaScript, Java, Python. If for any reason an implementation of a service for storing triples by means ClioPatria is not convenient, the Java-based Jena library can be used for the service realization.

A full-text index is provided by a Elasticsearch server. The implementation of this service is simple enough because any RDF graph is representable as JSON (JSON-LD). JSON is the main format of storage of indexed information in ElasticSearch. For the representation of the resource to the user on the search result page some tripes must be marked as representative of the result. Elasticsearch has the means of fuzzy comparison of terms that allows construction of relevant information search. The main aspects of module functioning are discussed later in the applications section.

1. Analogs of the software system under development

\_ The text of the document is being marked up with RDFa attributes. Semantic annotations allow development of search engines for Wiki pages that takes into account the semantics of the marked text [6].

Unlike Semantic MediaWiki, OntoWiki project similar results were obtained starting from different ideally. OntoWiki is based on the primary usage of the logical description of information in a semantic network. This logical structure is edited by means of the software generated user forms for the known terms in the vocabulary. The user can change only one text property of LOD lod:content, which, in General, contains HTML texts. The HTML markup is not tied to the logical structure of the displayed object (the subject). The text id edited with OntoWiki built-in WYSIWYG editor. The OntoWiki project is aimed at support of social networking technologies based on Linked Data [6].

As was mentioned above, the used the format of the markup of the documents is a development of the project Dokieli (https://github.com/linkeddata/dokieli) [2]. The project is a WYSIWYG-editor of LOD marked up HTML pages, display style sheet (CSS) and built-in subgraph (its content is not visible on the page) presented in different formats (TTL, N3, JSON-LD, TriG). The user is able to add a new RDFa markup with editor's tools, in this case, the function of our interest is implemented in the most general form. In addition, user authentication functions are implemented according to the WebID standard with client-side JavaScript. The system extensively supports commenting (text markup) with Open Annotation (oa). Annotation engine is integrated with gitter.im service, which shows readers' comments in the form of a dialogue. All the information is stored either on servers that support the Protocol Solid (https://solid.mit.edu/), or locally in the database of a browser. The edited document be moved (copied) from one Solid-store to another one easily. The resulting texts can be downloaded to the user workstation. Thus, dokieli is a system supporting distributed text content editing, which fully implements the interaction with other LOD resources, and all these functions are implemented exclusively by means of the client-side browser JavaScript.

There is also a huge class of document editors focused on the creation of scientific publications, see, e.g., http://substance.io/. In this study, the development of tools is aimed at automating the markup of a document based on the analysis of document changes proposed in [2], and specifically in this work, at implementing one of the infrastructural problems connected with a creation of the documents on the platform web browser controlled with LOD.

1. Applications of the technologies

The tools developed for typesetting documents are used in solution of applied problems.

* 1. The texts of the curricula of the universities

The Ministry of education and science of the Russian Federation after a series of experiments started the large-scale deployment of the Bologna process in the educational environment of the Russian Federation. One of the goals to be solved in the introduction is the transition to competence-oriented representation of the requirements to the pedagogical process. The reform affects all aspects of the process, including the system of classification of professions, introduction of "programs" and "directions", specialization by skill level (bachelor, master, etc.), introduction of applied bachelor degree, list of courses, goals and objectives of the courses agreed with the competencies specified in the Federal state educational standard, forms of conducting classes, e.g., the introduction of interactive forms, the distribution of lectures and practical sessions, etc. The existing documentation of the courses is supplemented with new forms of mandatory documents: the basis of assessment means (BAM), annotations of the courses. In addition, the university management with the aim of improving the quality of educational services introduced its own additions to the form, content and requirements to the documentation typesetting, in particular, to the quality of the conversion to HTML for publication on the website of the university.

For the minimum fulfillment of the requirements of the university management for each course the instructor is required to design at least three documents: the working program of the course, the summary and the BAM. This set of documents is prepared for each following possible combination: the university, department, specialty, direction (profile), program, qualification level (bachelor, specialist, undergraduate, graduate), academic or applied version of the qualification, form of study (full-time, part-time, evening, part-time, etc.). Each combination is presented in the curriculum of the university from which data is annually or twice a year should be synchronized in the work programs. The last five years showed the level of forethought of decisions taken by the managers of the ministry in terms of requirements to the presentation of a course – four generations of the standards (1,2,3 and 3+) have been developed – for each generation the documents in the appropriate "new" form should be prepared. The task of developing new handbook now seems to be more simple than before on the background of the document set preparation complexity. The raw statistic shows that most teachers are not able to cope with high-quality paperwork in the provided time constraints, resulting in the necessity to departments to hire a secretary, whose function is to bring documentation to the required quality level.

The solution to this problem is supposed to be in developing a software system that allows collecting the texts of the work programs, abstracts and BAMs from separate parts: a list of competencies and curriculum; the subject and module contents, whose text parts shared between different versions of documents. The title pages are generated from data of curriculum and designed templates. Microsoft Word and Excel are commonly used for this task with a built-in VBA, but in this case it is clearly not enough.

Consider the scheme of representation of a marked up work program in the proposed system.

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"  
 "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html lang="en" xmlns="http://www.w3.org/1999/xhtml"

**xmlns:taa="http://irnok.net/engine/rdfa-manipulation"**

xml:lang="en" metal:define-macro="page">

<head> <!-- Connecting stylesheets and modules -->

</head>

<body prefix="rdf: http://www.w3.org/1999/02/22-rdf-syntax-ns# ... foaf: <http://xmlns.com/foaf/0.1/> imei: imei.html#   
course: https://irnok.net/college/plan/01.03.02-16-1234-2461\_1%D0%BA\_PB-SM.plm.xml.xlsx-%D0%911.%D0%92.%D0%94%D0%92.3.1.html#"

resource="#post" typeof="schema:CreativeWork sioc:Post prov:Entity">

<!-- The application control panel -->

<main lang="en" resource="#annotation" typeof="oa:Annotation"

id="main-document-container">

<div **property="oa:hasTarget"   
 resource="#course-work-program"**></div>

<article **property="oa:hasBody"**   
 typeof="**schema:Article foaf:Document curr:WorkingProgram**"  
 resource="**#course-work-program**" id="main-document">

<div taa:content="imei:title-page"></div> <!--The title page..

<div taa:content="imei:neg-UMK"></div> <!--The approval page..

<section id="contents" class="break-after">

<h2 class="nocount c">Table of Contents</h2>

<div id="tableOfContents"></div>

</section>

<section id="course-description" resource="#description"

**property="schema:hasPart" typeof="schema:CreativeWork"**>

<div property="schema:hasPart" resource="#purpose"  
 **typeof="dc:Text cnt:ContentAsText"** >

<div **property="cnt:chars" datatype="xsd:string"**>

<h2 property="dc:title" datatype="xsd:string">The goals and the objectives of the discipline (module)</h2>

<p>

The purpose of the discipline "Programming Technologies" studying is the development of ...</p>

</div>

</div>

. . . . . . . .

<div property="schema:hasPart" typeof="dc:Text  
 cnt:ContentAsText" resource="#volume">

<div property="cnt:chars" datatype="xsd:string">

<h2 property="dc:title" datatype="xsd:string"> The аmount of discipline (module) content and training activities (divided by form of training)</h2>

<div **taa:content="course:time-differ"**></div>

</div>

</div>

. . . . . . . .

In the presented example, the key structures are highlighted in bold. Let us comment these structures.

1. The page displayed to the user is the annotation of the document "#annotation", with both the annotated content and the annotation text being the same at the stage of formation of the document - resource "#course-work-program". In LOD all the resources are global. In this case, this is achieved by substitution of the default namespace - the full URI of the current page - on the left hand side of the name of the resource.
2. The title page and the approval page are inserted from the page templates prepared in “imei.html” (information about the Institute of mathematics, Economics and Informatics of Irkutsk state University). The course data and the name of the profession is inserted into the template from the context of the document. All the key static templates of courses can be placed on one template page.
3. The text is divided into sections, and wrapped by tags <div> and <span> with the relevant RDFa structures. Analysis of the experience of the developers of LOD resources has shown that for the formation of the relationship is sufficient to use RDFa tags property, typeof and datatype. The usage of rel and about should be avoided. This makes the structure of semantic markup to be stricter by reducing the number of entities.
4. The command taa:content adds text from another page to the current document. The address of the included text is formed by the interpretation of the parameter of the command, for example, taa:content="course:time-differ" adds the text of the table of distribution of time spent between the types of classes (lectures, practice, laboratory works, etc.). To generate such tables, a server of web pages of these courses has been developed. Each page represents a course/module presented in a curriculum. The page content is defined by the structure of its URL and rendered at the time of first treatment. The text templates are identified by the id attribute.

In edit mode of the work program (document) all tags having the combination of attributes property="cnt:chars" and datatype="xsd:string" are converted to editable text. For clarity the document is displayed to the user without taa:-inclusions, and these inclusions are indicated with special tags and styles. At the finishing time moment, the edited text stored on the server file system. After that, the user can commit the changes and synchronize the text of the saved document with a repository.

Project source code is available in modules the Python programming language at the following address: https://github.com/isu-enterprise. Main module is isu.college.

* 1. Designing of legal documents

Students of the Institute of mathematics, economics and Informatics of Irkutsk state university developed an application for creating legal documents, non-standard form of judgments of judicial officers and notarial documents. The markup documents in this project is similar to the one used in the previous example, a number of ontologies for representation of significant terms in the legal domain were added: fibo (The Financial Industry Business Ontology), and formal specifications of identity documents acrt (A Certification Ontology).

Fibo ontologies allow specifying the role of entities in the document, e.g., for a general power of attorney, who is the principal (truster) and who is trusted:

<p>Я, <span   
 **property="fibol:designatesSignatory bibo:owner"**   
 typeof="**fibol:Signatory** foaf:Person **dbr:Principal**"   
 **resource="#principal"**><span **property="foaf:name" id="signatory-name" datatype="xsd:string"** **class="edit"**>Иванова Елена Викторовна</span>,<span **property="adoc:hasPassport" resource="#signatory-passport"** **typeof="acrt:Certification"**> <span **property="acrt:qualification" resource="dbr:Passport"**>паспорт</span>…

This example expresses the fact that the principal (fibol:designatesSignatory) is being certified (adoc:hasPassport) with a passport (acrt:qualification dbr:Passport).

The application has a mode for the document-editing in a view of form to be filled in. The JavaScript module searches for the tags containing a set of attributes datatype="xsd:string", class="edit" and converts the tag text into an input field, the main text is not editable. The basic grammatical form of sentences in accordance with the format of the RDFa, one can describe in the content attribute. It allows, e.g., storing peoples' names in the nominative case, and the displayed text to be in a suitable declination.

Harmonization of grammar is implemented using the attributes id, data- and class. The value of the class attributes equal to “disp” indicates that an edited string identified by id attribute is substitued as the text in this tag. The attributes data-m (masculine) and data-f (feminine) specify a variant of the word corresponding to the genus of the main noun in the text. The attribute data-case specifies the declination (the case and number) of the words of the text to be substituted. The grammar transformation algorithms are implemented on the server. Here is an example of text view of the power of attorney with grammatical structures.

...проживающ<span **class="disp" id="signatory-name" data-m="ий" data-f="ая"**>\_\_\_</span> по адресу:...Подпись <span **class="disp string" id="signatory-name" data-case="gent"**>\_\_\_</span> удостоверяю...

Copy functions and document data substitution from one document to another has been implemented. Further improvement of the software is a construction of a workflow graph based on analysis of data transfer between documents. The address of the application source code development page is https://github.com/isu-enterprise/isu.aquarium.

1. Further development of the technologies

At present, a large amount of information is published in tabular form in unstructured sources, e.g., government statistics, financial and business reports, sets of open scientific data, social media. The popular data representation formats of the sources are PDF, raster scans, DJVU and Word and Excel formats. These sources data is a valuable resource in the document editing. For example, the majority of regulations and education standards of Ministry of education and science of the Russian Federation are published in PDF format in scanned form. To be able to use data from these sources, it is necessary to extract the data and transform it into marked-up text or in tables of a relational database. The authors develop technologies for conversion, analysis, interpretation, cleaning and tracking the provenance of the tabular data from unstructured sources, the conceptualization of their natural-language content to provide a solution for this important task. We created an organization Cells Research Group at Github (https://github.com/cellsrg), where two projects have been implemented: a system of detection and recognition of tabular structures in PDF documents (https://github.com/cellsrg/TabbyPDF); a rule based semantic structure analysis of Excel spreadsheets (https://github.com/cellsrg/cells-ssdc). The recognition result of the tables is the data in a relational table [7].

LOD markup for the tabular data taken from the unstructured sources is described using the ontology qb (Qube ontology), which allows presenting the data as a hypercube. For each table cell, in a general case, a coordinate of the cell in the cube and the properties of the stored values, for example, the data type and units have been specified. The qb markup allows direct algorithmic processing for the table, i.e., without additional analysis of the structure.

The results are expected to be applied for automation of data publication of research results in the field of high-throughput sequencing of DNA/RNA carried out at the Limnological Institute SB RAS. In order to prepare the data and paper publication, the text and tables are to be additionally RDFa marked up with formats supported by biological LOD data servers, e.g., BIO2RDF (http://bio2rdf.org/). The implementation of this service for LIN SB RAS, the researchers will integrate data on the microbiome of lake Baikal in the global research environment of species of plants, animals and micro-organisms similar to [8].

Another important task of analysis of unstructured information is to automate the layout and the structure recognition of similar documents according to a template. The task is relevant for universities, where a large amount of material in the form of training programs and handbooks are accumulated. The material could be converted into various forms of preparation, such as training programs on the modern standard FOS-3+, e-courses of Moodle, etc. In addition, the universities of Russian Federation are developing intelligent systems for training automation where the material the student is given depending on a predefined general scenario and the results of the assessment of the current knowledge. Marked up methodological and documentary material could be used for partial automation of the process of populating the module databases of e-learning systems.

1. Conclusion

The report discusses the utilization of the Linked Open Data (LOD) technologies for solving problems of creating text documents with the editing and programming means provided by modern web browsers. Thanks to the LOD notation, a data integration of documents and web applications into a single resulting text document is being implemented.

The editing and typesetting of the documents is carried on using algorithms that interpret the relation between the elements of document's HTML tree. The relations are described by means of semantic web and LOD, i.e., with RDFa structures and commands defined in the attributes of HTML tags. The layout algorithms are activated in the presence of corresponding conditions in the nodes of the document tree. Each algorithm changes the tree structure, constructing the final content and layout of the document. The edited documents are stored on servers that support HTTP Protocol (GET and POST requests). This technology is very scalable even for shared hosting environment supporting only of PHP programming. Test storages are implemented for different programming languages.

A few examples of the use of the developed tools has been presented in an educational environment, preparation of legal documents etc. Further development of the project is carried out in several directions: a) improvement of the means of typesetting documents, b) the automation of the document structure recognition based on the analysis of its changes [2], c) the implementation of domain applications, d) the collection and analysis of user activity and requirements, d) development of a regulated access to documents. The development is aimed at the implementation of a software environment for a global electronic document workflow, allowing individuals and organizations to share document data, logical structure which is difficult to formalize within the framework of traditional approaches to workflow automation.

1. Acknowledgments

The results obtained with the partial support of the following projects:

* Irkutsk scientific center of SB RAS No 4.1.2;
* The Council for grants of the President of Russian Federation, state support of leading scientific schools of the Russian Federation (NSH-8081.2016.9).

The results obtained with the use of the network infrastructure of Telecommunication center of collective use "Integrated information-computational network of Irkutsk scientific-educational complex" (CCU, IIWS of IRNAK) (http://net.icc.ru). The authors are grateful to the community of Linked Open Vocabularies (http://lov.okfn.org/dataset/lov/) resource for assistance in the search for formalizations of subject areas (ontologies).

References

1. Bizer, Ch., Heath, T., Berners-Lee, T. Linked Data – The Story So Far. *International Journal on Semantic Web and Information Systems.* 5 (3): (2009) 1–22. doi:10.4018/jswis.2009081901. ISSN 1552-6283.
2. Cherkashin E., K.Paskal, I.Bychkov et al An Ontology Polysystem Driven Technique for Semantic Markup for Edited Documents Procs. of International conference on Applied Internet and Information Technologies. University of Novi Sad, Technical Faculty ”Mihajlo Pupin”, Zrenjanin, Zrenjanin, Serbia, October 24, 2014. pp. 201-206.
3. Capadisli, S., Guy, A., Verborgh, R., Lange, C., Auer, S., Berners-Lee, T.: Decentralised Authoring, Annotations and Notifications for a Read-Write Web with dokieli, Procs of ICWE international conference, 5-8 June, 2017, Rome, Italy. (to appear) Preprint URL:http://csarven.ca/dokieli-rww
4. Wielemaker, J., Beek, W., Hildebrand, M., Ossenbruggen, J. ClioPatria: A SWI-Prolog infrastructure for the Semantic Web.  [*Semantic Web*](https://www.researchgate.net/journal/1570-0844_Semantic_Web) 7(5):529-541, 2016, DOI: 10.3233/SW-150191
5. Lager, T., Wielemaker, J. Pengines: Web Logic Programming Made Easy. *Theory and Practice of Logic Programming* 14(4-5), 2014, DOI: 10.1017/S1471068414000192
6. N.Heino, S.Tramp, N.Heino, S.Auer. Managing Web Content using Linked Data Principles – Combining semantic structure with dynamic content syndication. Computer Software and Applications Conference (COMPSAC), 2011 IEEE 35th Annual. pp. 245 - 250. URL:http://svn.aksw.org/papers/2011/COMPSAC\_lod2.eu/public.pdf (access date: 30.05.2013).
7. Shigarov A. Rule-Based Table Analysis and Interpretation // Communications in Computer and Information Science. Springer. 2015. Vol. 538. pp. 175-186. URL:http://link.springer.com/chapter/10.1007%2F978-3-319-24770-0\_16.
8. [Dalamagas, T., Bikakis, N., Papastefanatos, G., Stavrakas, Y., Hatzigeorgiou A. Publishing Life Science Data as Linked Open Data: the Case Study of miRBase Proceedings of the first International Workshop On Open Data, WOD-2012.](https://arxiv.org/find/cs/1/au:+Dalamagas_T/0/1/0/all/0/1)   
   URL: <https://arxiv.org/abs/1205.2320>