Modelling Information Systems by Document Flow Description

Vladimir Ivančević, Marko Knežević, Ivan Luković
University of Novi Sad, Faculty of Technical Sciences,
Trg Dositeja Obradovića 6, 21000 Novi Sad, Serbia
Email: {dragoman, marko.knezevic, ivan}@uns.ac.rs

Verislav Djukić
Djukic — Software Solutions,
Gärtnerstr. 17, 90408 Nürnberg, Germany
Email: info@djukic-soft.com

***Abstract*—In this position paper, we argue in favour of three points related to document-centric modelling of information systems: (i) information systems of some organizations may be understood in terms of documents, actions, actors, and document flow; (ii) modelling document flow may be a central step in modelling an information system; and (iii) as an approach to information system modelling, document flow modelling may be coupled with the form-based approach to information system development that is featured in IIS*Case, a model-driven development tool, because the form concept is semantically close to the document concept. With respect to these claims, we formulate a document-centric and model-driven approach to information system development, explain its particularities, and present a plan of activities that should lead to its implementation. By relying on domain-specific languages, the proposed approach should allow generation of information systems supporting document manipulation within a document flow and process mining.**

I. Introduction

DOCUMENTS are central to information systems (ISs). They may represent a plan of future actions or evidence of past activities that are relevant to the functioning of an information system. Therefore, in this position paper, we argue in favour of three points regarding information system modelling with respect to documents. First, we argue that some information systems may be described by specifying instances of concepts belonging to one of the following categories: documents, actions regarding document manipulation, actors, and document flow. Each piece of information that is needed for the operation of these information systems belongs to an instance from one of the aforementioned categories. Second, we argue that modelling document flow may be considered a central part of information system modelling since it integrates instances from all of the four categories. Furthermore, specifying document flow is generally platform independent and detached from the technology specific terminology, which makes it accessible to business analysts and allows their involvement with the IS modelling process. Third, we argue that document flow modelling may be integrated with the form-based approach to IS development, which is utilized in the IIS*Case software development tool [1], owing to the semantic similarity between forms and documents. The integration of the two approaches would provide the means to develop process-aware information systems (PAISs), which in turn would also support the use of process mining in generated instances of ISs. Therefore, in this paper, we present the rationale behind our claims, as well as sources that are related to the aforementioned issues. We also elaborate on the proposed document-centric approach to IS development and give a plan of activities that should lead to the implementation of document flow modelling capabilities and process mining by relying partially on the features of IIS*Case.

The research presented in this paper was supported by the Ministry of Education, Science and Technological Development of the Republic of Serbia, Grant III-44010

The proposed IS development approach, which in addition to being document-centric is also model-driven, would allow rapid automatic generation of prototype ISs. However, its principal advantage over traditional approaches is its suitability for the development of ISs in organizations whose activities are tightly related to constant manipulation of high volumes of official documents. Generated ISs would have built-in features supporting document tracking within the flow, document flow analysis, and use of attributes that are typical of documents (such as references between documents and validity dates). Furthermore, domain-specific languages (DSLs) have a key role in the proposed approach, because each step in the modelling process requires creation of a different specification or model. For that reason, there is a strong need to have separate DSLs for specifying document structure, modelling document flow, and defining form types. It is also necessary to define transformations between these DSLs since specifications of document structure and flow models need to be integrated first and then translated to specifications of form types, which are further used in the generation of IS prototypes. As the proposed approach combines domain-specific modelling languages (DSMLs), transformation engines, and generators, it represents a fine example of Model-Driven Engineering [2].

In the following sections, we argue in favour of the three claims that are stated herein. In Section II, we illustrate how capturing document flow may help uncover activities that are relevant for the functioning of the information system in an organization. Section III points out how document flow modelling may serve as a basis for information system modelling and implementation. In Section IV, ideas from the two aforementioned sections are evaluated with respect to the IIS*Case tool, which allows form-based modelling and development of information systems.

II. Relevance of documents and document flow within organizations

According to Briet [3], whose work has significantly influenced modern perspective on documentation, document is defined as "any concrete or symbolic indication, preserved or recorded, for reconstructing or for proving a phenomenon, whether physical or mental" [4]. Documents, whether they are paper-based or digital, are essential in any organization. This relevance may be further observed in the context of information retrieval, where a document is characterized as "a unit of retrieval. It might be a paragraph, a section, a chapter, a Web page, an article, or a whole book." [5]. Therefore, by being central to information retrieval, documents are also central to information systems and, consequently, could serve as a basic unit in IS modelling.

Although there are records that are required by state regulations, record keeping is primarily needed for purely practical reasons because information within documents about past, current, or planned activities is vital to the accomplishment of the organization's mission. For instance, in addition to keeping usual information about employees or finances, functioning of a faculty requires storing information about its central activities, such as teaching and research. Such records contain data about student enrolment, instructor assignment, individual assessment results, final grades, and many other aspects of faculty operation. These records are typically organized in the form of documents that have a precisely defined structure and life cycle. Furthermore, there are also strict regulations about who may create, have access, modify, or destroy a document. We illustrate the stated viewpoint on the example of electing an instructor for a vacant position at a faculty. The following description is based on a segment of the actual election process at the Faculty of Technical Sciences in Novi Sad. The dean of the faculty may make an official decision about announcing a call for a vacant position. Such decision represents an official document that needs to be signed and archived at the faculty. In the similar manner, the elective council of the department to which the vacant position is associated needs to make a decision to appoint committee for applicant evaluation, which is yet another document. Once these two documents come into effect, a third document, the decision on the appointment of the committee members for the evaluation of applicants, may be created. Next, applicants are required to submit a set of prescribed documents. Once the application period is closed, the committee members write a report on the applicants (another document). After another decision of the department's elective council and consent from the faculty's educational-scientific council, the dean may make a decision to elect the candidate that is recommended by the elective committee. As illustrated, instructor election may be a complex process involving creation of numerous documents and intervention of various faculty members and non-faculty personnel. It includes storing several decisions of faculty bodies (faculty documents) and candidate applications (also documents). Each relevant step of the process is marked by an official document.

For each workflow at a faculty or some other organization, there may be a model that contains: instances of required documents, description of action flow between these instances, actors that are involved in the workflow, and specification of actions regarding document manipulation. From the perspective of data, such a document-centric model could capture all the relevant aspects of activities within the modelled system. Document content determines which data need to be stored in the information system of an organization, while action flow indicates how the state of a document may change. Actors correspond to actual user groups of the information system while actions define which document data may be manipulated by which user group. Therefore, the four principal elements of such model include instances of: documents, actions regarding documents, actors, and document flow between actions.

The use of documents and document flows within a system generally requires formalisms for document description. For instance, according to the Pentaformat model [6], five components may be extracted from a document: content, structure, presentation, behaviour, and metadata. For that reason, it is necessary to define languages that could be utilized to describe all of the aforementioned dimensions. These languages could be further used to model various aspects of organizational systems and, consequently, ISs, such as communication between actors and the system where documents are means of implementing this communication.

The document-centric approach may not be the most convenient solution for all types of organization, particularly not for those where document flows are unstable or where structured documents are not much used. Nonetheless, in organizations where documents are explicitly acknowledged, precisely defined and constantly manipulated, such approach should be preferred because its main concepts are semantically closer to actual reality and functioning of those organizations.

III. Developing information systems with respect to document flow

In IS modelling, there are many well-tried approaches. As evidenced by the current popularity of UML (Unified Modeling Language) [7], BPMN (Business Process Model and Notation) [8], WS-BPEL (Web Services Business Process Execution Language) [9], and YAWL (Yet Another Workflow Language) [10], process and workflow modelling are often used as means of capturing the essence of the system for which an IS is built. Modelling document flow is generally similar to modelling a workflow since the nature of flow varies little between these modelling scenarios. The main difference between the two is that, in document flow modelling, an activity generally results in document creation or modification, while in workflow modelling a result is not necessarily tied to any document. If the main focus of understanding an organization and its activities is set on understanding manipulated official documents, which in many organizations represent the only relevant records, then it may be possible to reduce workflow modelling as a basis of IS modelling to document flow modelling. An added benefit of this approach would be the possibility to employ best practices from workflow modelling, such as workflow patterns [11], [12], [13]. In the rest of this section, we elaborate on the proposed approach.

Documents represent the only version of truth. If a piece of information may not be found within the archived documents, then it is considered to be missing or unavailable. All of the official actions within an organization should be based on information that may be found within the documents. A system is not understood in terms of various modelled entities, which is typical for many traditional IS development approaches, but solely as an environment in which documents are circulating along the predefined paths. In many instances there is little practical difference between entity models and documents (document types). For example, at a faculty, information about individual students or staff members may be contained within personal dossiers. However, the key difference between the proposed and the traditional approaches is in the view of an organization since the proposed approach is based on a paradigm that is document-centred. Such paradigm implies that arbitrary entity types are substituted by documents and all of their particularities. Therefore, modelling document flow in terms of the four concepts mentioned in the previous section (document, action, actor, and flow) could prove to be a feasible approach for IS development. Nevertheless, in the context of both IS modelling and implementation, additional information about these four concepts needs to be specified.

For each document, there should be a complete specification of its structure and dependencies (relations to other documents). The document concept denotes all document instances that conform to a shared specification. A document may be divided into sections and these further decomposed into subsections and fields. A section should feature a name. For each field, there should be a name, data type, and indication whether it is mandatory to be filled in.

There are various constraints that are generally related to document instances. Temporal aspects of a document instance may be relevant in certain cases, e.g., employment period is typically specified in employment contracts for faculty staff members. Therefore, a document instance should feature optional information regarding its validity period: a date when it comes into effect and expiry date. There are some documents that should have only one active (valid) instance. Such cases should be explicitly specified for each document. In this manner, the satisfaction of the aforementioned constraints could be automatically checked. Moreover, the case when a document is expected to be signed (verified) by a group of authorized persons may also be explicitly indicated in the document specification.

Relations between documents (or their parts) should be explicitly defined for three principal reasons: (i) to avoid information redundancies; (ii) to support detection of inconsistencies in the model; and (iii) to support analysis of document flows and, consequently, of workflows in the organization.

These relations may belong to one of the two types: (i) referring relations, where a relation only marks other document (or its part) that is semantically relevant for the current document but whose contents are not included in the current document, similarly to the notion of a hyperlink in a web page; and (ii) inclusive relations, where a relation also indicates insertion of selected contents from other document into the current document.

In document flow, each step in the life cycle of a document represents a different document state. Each state is described by a set of actions that associated actors need to perform on a document instance or set of its parts so the instance could advance to the next state in the flow. A document flow is not restricted to a single document, as it should enclose all documents that are relevant to a well-defined procedure within an organization. Allowed actions include creation, reading, modification, signing, copying, and removal. An action may refer to a whole document or a set of its parts, with the exceptions of creation and removal, which may refer only to whole documents. Similarly to other flow modelling languages, there should be support for at least the following relations between states: sequences, parallelism, loops, exceptions, and cancelation. A single state may have multiple preceding or multiple succeeding states. Conditions for state transitions should include relational and logical operators, while satisfaction of these conditions would allow actions in one or more succeeding states depending on the actual relation between the current and the succeeding states. Each flow has at least one initial and at least one final state. Starting from an initial state in the flow, a final state may be reached by satisfying conditions for intermediary states that connect the initial and final state. Two flows may be associated by connecting a final state of the preceding flow to an initial state of the succeeding flow using one of the aforementioned state relations. In this manner, long flows may be decomposed and the complexity of flow models significantly reduced. Initial and final states are further classified as: (i) independent, which cannot be associated to any state from other flows; or (ii) dependent, which have to be related to a state in another flow.

As opposed to decomposition, it should be also possible to automatically form a complete model of document flows that features each individual flow, whether it is related to other flows or not. Such universal model could indicate all of the actions that initiate different workflows in an organization, as specified by independent initial states. Moreover, it could provide analysts with a valuable perspective on all document-related processes within an organization.

Documents and actors are also related through actor permissions regarding document manipulation. The permission concept is derived from the basic concepts, which are presented in Section II. Each actor permission refers to a selected document state, action and actor. An actor permission applies to all of the subsequent states of that document inside its flow. In the case of conflicting actor permissions for a single state, precedence is given to the permission whose assignment is closest to the state, starting from the state in conflict and going upwards in the document flow. As in the case of actions, actor permissions refer to document creation, reading, modification, signing, copying, and removal. Each type of permission may be granted or denied. If there is no explicit declaration of actor permission for a document in some state, then such permission is considered to be denied. Furthermore, permissions regarding reading, modification, signing, and copying, may also refer to individual document parts.

The proposed approach is model-driven because the information that is necessary for generation of IS prototypes may be found within the models of document flow and the transformed flow information could be used as an input to the process of form-based IS development in IIS*Case or some other tool. Moreover, the application of the approach should result in generation of a PAIS, i.e., "a software system that manages and executes operational processes involving people, applications, and/or information sources on the basis of process models" [14]. Generated ISs should allow their users to work in terms of document flow by featuring automatized notification for new work tasks that are related to manipulation of a single document. These tasks would correspond to states in flow models and, in a single document flow, an IS user should be able to perform only actions that are associated to the current state in the flow. However, another idea worth considering is the implementation of an option to bypass a part of a regular procedure because every model is a simplification of the real process and does not encompass all process instances that may occur in an organization. Automatic storing of information about all performed user actions (document manipulation according to some document state) would allow process mining, i.e., analysis of document flows in terms of time, actors, and actual vs. expected paths of documents in the organization (flow-to-model and model-to-flow conformance).

IV. Integrating document-centric IS development into IIS*Case

The document-centric view of a system, as expressed in the previous section, shares some of its main points with the approach to IS generation that is used in IIS*Case. In this section, we elaborate on how concepts such as document, action, actor, and document flow may be merged with the standard concepts of IIS*Case.

IIS*Case is a software tool for IS development that has been constantly extended and updated over the past 20 years [1]. It relies on the form concept as the basic unit in the development of database schemata [15], [16] and ISs [17]. Such form [18], [19] is actually a screen form in an application that is used to enter, view, modify, or remove data about an entity which may be recorded in an IS. It generally corresponds to a business document that is used in the organization supported by the IS. Form specification is fully detached from the technology that is used in IS implementation, which could facilitate integration of the two approaches. All of this indicates that the form concept may be employed in the proposed document-centric approach since form specification generally covers description of the corresponding document's structure. Furthermore, document flow, which may encompass several documents, could be mapped to the application system concept, which is primarily used to group form types in IIS*Case.

Using IIS*Case, it is possible to generate program code of an IS by specifying forms (form models) and then running a transformation process. The concepts that may be used to create these platform independent models (PIMs) of an IS are formally presented in a meta-model [20] which was specified using the EMF Ecore [21] version of MOF [22]. For form-based IS design, IIS*CDesLang [23], a textual DSL, was specified by an attribute grammar using the VisualLisa programming environment [24]. For the similar purpose, another textual DSL is being developed by directly using the meta-model of IS concepts to generate concrete syntax of the language. In the integration of the two approaches, these two DSLs could serve as target languages in the translation of document specifications. This would require construction of a new DSL for creating document descriptions that are in accordance with the guidelines from the previous section. In case the new document specification DSL is derived from a meta-model, one of the available QVT [25] implementations could be used to map the concepts of this new DSL to the IS concepts in IIS*Case. Another possibility that also deserves consideration is the direct use of the two IS design DSLs instead of a more document-centric specification language.

On the other hand, some concepts from the proposed document-centric approach do not have a match in IIS*Case. In the latest version of IIS*Case, there is no explicit notion of document flow. Therefore, a DSML should be devised for this purpose and combined with the document specification DSL. Since document flow is modelled by human experts, a graphical language would be the most convenient solution. The main reason for this decision may be found in the results of an empirical study that compared understanding of business process descriptions with respect to the used type of notation [26] because the authors concluded that business analysts benefited from reading a graphical model. Regarding the semantics, the new language could be formally based on Petri Nets [27] or a graphical variant of π-calculus [28], as both have been extensively used to model various types of processes. Whatever the formal foundation of the language, document flow modelling would become a central part of the IS development process in IIS*Case. It would shift focus from data entry and grant a more prominent role to processes within the organization, which has been a general trend in IS development in recent years.

Furthermore, there is a need to introduce actors (users) and actions/permissions regarding document use. Actors and their permissions would have to be explicitly specified, possibly within the new document flow language or the document specification language. These concepts would be tightly related to forms in the generated application: actors would correspond to application users, while actor permissions would correspond to data manipulation privileges within the application. As a result, both the meta-model and IIS*CDesLang would need to be updated with these new concepts.

In IS development, explicit acknowledgement of processes within an organization, has many far reaching implications. In the latest generation of process-aware ISs, process mining [29] allows advanced analysis of processes within an organization and, consequently, their improvement. In the proposed document-centric approach, similar benefits could be obtained by making ISs that are generated using IIS*Case process-aware. In order to support process mining, key events (actions) in the generated ISs need to be recorded in special logs. In this context, an event designates a transition of a document to some of the modelled states. For each event, there should be information about the corresponding document instance, reached document state, actor whose action triggered that transition, and timestamp. Such event logs could serve as input to various types of process-related analyses.

Support for analysing explicitly modelled document flows could be automatically added to ISs during the generation process in IIS*Case. One component of each generated IS should be focused on process mining and support viewing of document flow models and recorded events. Out of the typical process mining scenarios [30], application support should be added for the following two: (i) conformance, in which recorded document flows are compared to those expected according to flow models; and (ii) extension, in which flow models are enhanced with concrete information about flow path coverage, load distribution between actors, and temporal aspects of flow. The conformance check would help analysts identify how exactly a procedure deviates from the plan (model), i.e., which recorded events are not expected since the action sequence that they are part of is not supported by the corresponding flow model. A closer inspection of such events could uncover periods when these deviations most often occur and indicate which actors are responsible. On the other hand, the extension could provide an analyst with valuable information on which path segments require the longest time to complete or which actors are disproportionately burdened by workload. Such insight could help reduce delays in workflows or achieve better load balancing.

The form-based approach to IS development in IIS*Case may serve as foundation for the proposed document centric approach. However, as previously mentioned, there are several key activities that should be performed in the integration of these two approaches: (i) a DSL for document structure specification needs to be constructed, together with adding support for translation of document specifications to programs written in IIS*CDesLang or the DSL for IS design that is being developed from the meta-model of the IIS*Case PIM concepts; (ii) a graphical DSML for document flow modelling needs to be constructed and coupled with the document specification language; (iii) notions of actor and action/permission need to be introduced into IIS*Case; (iv) ISs that are generated by IIS*Case need to be also process-aware; and (v) ISs that are generated by IIS*Case need to feature a set of typical process mining capabilities related to conformance and extension. The details on the implementation of process mining support within IIS*Case also need to be specified in our future research. Once all of the aforementioned activities are completed, a detailed evaluation of the proposed approach could be made, together with a comparison to other approaches to IS development.

V. Related work

The importance of the data perspective in process modelling has increased recently, as evidenced by the growing number of data-oriented approaches to workflow and process modelling [31], which also include document-based approaches.

In [32], the authors propose an XML Document Centric Workflow Management System (XDoc-WFMS), which supports embedding document flows within organization workflows, document access control, and agents handling various tasks within the system. Although XDoC-WFMS and the approach proposed herein share similar goals and concepts, there are three key differences. First, XDoc-WFMS uses the XML format to store documents, whereas we do not restrict our approach to a single document storage format — documents are going to be stored in a database automatically designed in the system generation process within the IIS*Case tool, which may include different types of databases. Second, as opposed to our approach, XDoc-WFMS does not acknowledge the need for process mining, which may be understandable given the period when the approach was conceived. Third, the main objectives of the two approaches do not completely coincide. Whereas XDoc-WFMS should be considered only a part of a larger system in which document processing should be integrated with other business activities, our approach is aimed at the development of information systems for organizations where every activity has to be formally documented. XFlow [33] represents another example of a document-centric workflow framework that revolves around agents and XML documents. In addition to this, it heavily relies on XML-related technologies such as XSLT, XPath, XForms, and SVG.

In [34], the authors present a document-driven workflow system not focused on the control flow perspective. Unlike the previous two XML-based approaches, this one is implemented using a relational database management system (RDBMS) and database triggers, which are used in the enactment of the workflow system. Although such approach may be suitable for ad hoc workflows, which lie between the well-structured workflows and less-structured cooperation systems, processes cannot be visualised because there is no predefined control flow, which in turn narrows down the environments where this approach may be used. For these reasons, we may adopt only some of the ideas exhibited in that approach.

VI. Conclusion

In this position paper, we outline some of our ideas on IS modelling and development with respect to the role of documents in organizations. We argue that understanding documents, their flows, and performed actions of actors in an organization may serve as a basis for IS development. The proposed document-centric and model-driven approach, which includes explicit modelling of document flows and document structure specification, could be a step preceding IS generation. On the other hand, the approach to IS development in IIS*Case, which revolves around specifying forms for data manipulation and generating executable code using these specifications, could be used in the later phase of IS development. Based on the comparison of concepts from the two approaches, it appears that these approaches could be integrated owing to the similarity of the form concept to the document concept. Document specification and flow modelling would represent the first phase in IS development, while the form-based approach, after a series of intermediate transformations, would provide generation of an IS. Each step in this development process would include using one of the custom DSLs to create necessary specifications. In the context of integrating the two approaches, primary research activities are identified in this paper. Because the unified approach would acknowledge organization processes in the form of document flows, it could be also classified as process-based. The idea to make generated ISs process-aware and add "out-of-box" support for process mining is motivated by the latest trends in IS development that are gaining commercial recognition.

In the proposed approach, there are certain issues whose resolution is a part of future research efforts. These include: (i) finding a way to support changes in document structure or flow without the need to fully replace the previous version of IS (change patterns [35] could be a starting point); and (ii) extend the approach to support the idea of "a paperless office" where documents are (almost) exclusively stored and manipulated in their digital form. After adding new concepts and document flow modelling capabilities to IIS*Case, the unified approach could be more thoroughly evaluated and compared to other IS development approaches. This addition to IIS*Case could also serve as a basis for automatic generation of SOA (service-oriented architecture) system specifications and more thorough analysis of the modelled system. Moreover, the application of the proposed approach could improve understanding between the two key groups of people involved with IS development – business users who state requirements and actual system users on one hand, and analysts and IS designers on the other.

REFERENCES

[1] I. Luković, V. Ivančević, M. Čeliković, and S. Aleksić, "DSLs in action with model based approaches to information system development," in *Formal and Practical Aspects of Domain-specific Languages: Recent Developments*, ed., M. Mernik. IGI Global, Hershey, PA, 2012, pp. 502–532.

[2] D. C. Schmidt, "Model-driven Engineering," *IEEE Computer*, vol. 39, no. 2, 2006, pp. 25–31.

[3] S. Briet, *Qu'est-ce que la documentation?*, Éditions documentaires, industrielles et techniques, Paris, 1951.

[4] M. Buckland, "Information as Thing," *Journal of the American Society of Information Science*, vol. 42, no. 5, 1991, pp. 351–360.

[5] R. Baeza-Yates and B. Ribeiro-Neto, *Modern Information Retrieval*, Addison Wesley Longman Publishing, Boston, MA, 1999.

[6] A. Di Iorio, "Pattern-Based Segmentation of Digital Documents: Model and Implementation," Doctoral thesis, University of Bologna, Italy, 2007.

[7] Unified Modeling Language (UML), http://www.omg.org/spec/UML/

[8] Business Process Model and Notation (BPMN), http://www.omg.org/spec/BPMN/index.htm

[9] Web Services Business Process Execution Language Version 2.0, http://docs.oasis-open.org/wsbpel/2.0/OS/wsbpel-v2.0-OS.pdf

[10] YAWL: Yet Another Workflow Language, http://www.yawlfoundation.org/

[11] W. M. P. van der Aalst, A. P. Barros, A. H. M. ter Hofstede, and B. Kiepuszewski, "Advanced Workflow Patterns," In *Proceedings of the Fifth IFCIS International Conference on Cooperative Information Systems (CoopIS'2000)*, volume 1901 of Lecture Notes in Computer Science, Springer, Eilat, Israel, 2000, pp. 18–29.

[12] W. M. P. van der Aalst, A. H. M. ter Hofstede, B. Kiepuszewski, and A. P. Barros, "Workflow Patterns," *Distributed and Parallel Databases*, vol. 14, no. 1, 2003, pp. 5–51.

[13] N. Russell, A. H. M. ter Hofstede, W. M. van Der Aalst, and N. Mulyar, "Workflow Control-flow Patterns: A Revised View," 2006, http://www.workflowpatterns.com/documentation/documents/BPM-06-22.pdf

[14] M. Dumas, W. Van Der Aalst, and A. Ter Hofstede, *Process-aware Information Systems*, John Wiley & Sons, Hoboken, NJ, 2005.

[15] I. Luković, S. Ristić, P. Mogin, and J. Pavićević, "Database Schema Integration Process — A Methodology and Aspects of its Applying," *Novi Sad Journal of Mathematics*, vol. 36, no. 1, 2006, pp. 115–140.

[16] I. Luković, P. Mogin, J. Pavićević, and S. Ristić, "An Approach to Developing Complex Database Schemas using Form Types," *Software — Practice & Experience*, vol. 37, no. 15, 2007, pp. 1621–1656.

[17] S. Ristić, S. Aleksić, I. Luković, and J. Banović, "Form-driven Application Development," *Acta Electrotechnica et Informatica*, vol. 12, no. 1, 2012, pp. 9–16.

[18] P. Mogin, I. Luković, and Z. Karadžić, "Relational Database Schema Design and Application Generating using IIS*CASE tool," in *Proceedings of International Conference on Technical Informatics*, Timisoara, Romania, Vol. 5, 1994, pp. 49–58.

[19] P. Mogin and I. Luković, "A Prototyping CASE Tool," in *Proceedings for the Dedicated Conference on Rapid Prototyping in the Automotive Industries*, XXVIII International Symposium on Automotive Technology and Automation, Stuttgart, Germany, 1995, pp. 261–268.

[20] M. Čeliković, I. Luković, S. Aleksić, and V. Ivančević, "A MOF based Meta-model and a Concrete DSL Syntax of IIS*Case PIM Concepts," *ComSIS*, vol. 9, no. 3, 2012, pp. 1075–1104.

[21] Eclipse Modeling Framework Project (EMF), http://www.eclipse.org/modeling/emf/

[22] OMG's MetaObject Facility, http://www.omg.org/mof/

[23] I. Luković, M. J. V. Pereira, N. Oliveira, D. d. Cruz, and P. R. Henriques, "A DSL for PIM Specifications: Design and Attribute Grammar based Implementation," *ComSIS*, vol. 8, no. 2, 2011, pp. 379–403.

[24] N. Oliveira, M. J. Varanda Pereira, P. R. Henriques, D. Cruz, and B. Cramer, "VisualLISA: A Visual Environment to Develop Attribute Grammars," *ComSIS*, vol. 7,no. 2, 2010, pp. 265–289.

[25] Meta Object Facility (MOF) 2.0 Query/View/Transformation (QVT), http://www.omg.org/spec/QVT/

[26] A. Ottensooser, A. Fekete, H. A. Reijers, J. Mendling, and C. Menictas, "Making Sense of Business Process Descriptions: An Experimental Comparison of Graphical and Textual Notations," *Journal of Systems and Software*, vol. 85, 2012, pp. 596–606.

[27] C. A. Petri, "Kommunikation mit Automaten," Unpublished doctoral thesis, Fakultät für Mathematik und Physik, Technische Hochschule Darmstadt, Germany, 1962.

[28] R. Milner, *Communicating and Mobile Systems: The π-calculus*, Cambridge University Press, Cambridge, UK, 1999.

[29] W. M. van der Aalst, *Process Mining: Discovery, Conformance and Enhancement of Business Processes*, Springer Verlag, Berlin, 2011.

[30] W. van der Aalst, "Process-aware Information Systems: Lessons to be Learned from Process Mining," *Transactions on Petri Nets and Other Models of Concurrency II* (2009), pp. 1–26.

[31] H. Garcia, "Evaluation of Data-centric Process Modeling Approaches," Master thesis, Eindhoven University of Technology, the Netherlands, 2011.

[32] R. Krishnan, L. Munaga, and K. Karlapalem, "XDoC-WFMS: A framework for document centric workflow management system," in *in Conceptual Modeling for New Information Systems Technologies*, Springer Berlin Heidelberg, 2002, pp. 348–362.

[33] A. Marchetti, M. Tesconi, and S. Minutoli, "Xflow: An xml-based document-centric workflow," in *Web Information Systems Engineering — WISE 2005*, Springer Berlin Heidelberg, 2005, pp. 290–303.

[34] J. Wang and A. Kumar, "A framework for document-driven workflow systems," in *Business Process Management*, Springer Berlin Heidelberg, 2005, pp. 285–301.

[35] B. Weber, M. Reichert, and S. Rinderle-Ma, "Change Patterns and Change Support Features — Enhancing Flexibility in Process-aware Information Systems," *Data & knowledge engineering*, vol. 66, no. 3, 2008, pp. 438–466.