Oct 31st, 1:30 PM - 4:15 PM

Proposal of a legal framework through the development of new domain specific languages (DSL) in compliance with GDPR

Tea Tavanxhiu
University of Tirana, tea.tavanxhiu@unitir.edu.al

Andrea Alberici
Studio Synthesis, a.alberici@studiosynthesis.biz

Kozeta Sevrani
University of Tirana - Faculty of Economy, kozeta.sevrani@unitir.edu.al

Proposal of a legal framework through the development of new domain specific languages (DSL) in compliance with GDPR

Tea TAVANXHIU[1], Andrea ALBERICI[2], Kozeta SEVRANI[3]
[1,2,3] University of Tirana
tea.tavanxhiu@unitir.edu.al
a.alberici@studiosynthesis.biz
kozeta.sevrani@unitir.edu.al

Abstract. The adaptation of company processes to the EU Regulation represents a major opportunity to review, update and improve the internal processes and management tools used. The loss of data, in most cases, causes serious damage to the image and very often the total closure of the company. The legislation therefore represents an opportunity and a stimulus to verify the management methods applied, to define an organizational model and a code of conduct (policies, processes, rules / provisions and controls) capable of improving internal processes, defining and achieving desired objectives, ensure data and systems protection with proper risk management and assessment. This paper presents the principles of the LegalRuleML applied to the legal domain like General Data Protection Regulation (GDPR) and discusses reasons that LegalRuleML is convenient for modeling norms. We need to understand why it is important to develop a specific domain language that refers to internal GDPR privacy consulting and BPM mapping. LegalRuleML allows inconsistent renditions of a legal source to coexist in the same LegalRuleML document and provides functionality to identify and select interpretations.

Keywords: *LegalRuleML, framework, business, process, GDPR*

1. Introduction

This paper represents an in-depth analysis of the different specific problems and technological requirements that a domain specific language has. We have examined some aspects of the needs of a company and also realized the classification of the specific domain. Several analyses have been made, like the analysis of the methods applied for the DSL and the development of new programming languages for specific purposes.

After further verifying and studying the framework[3], it results that it serves to derivate DSPML from a generic BPML which sufficiently addresses the needs expressed in this literature. By using this artifact, the engineering of domain-specific process modeling languages can be methodologically grounded, which structures and systematizes the development process. This case leads to an increased adequacy and quality of resulting languages, which need to be designed towards increasingly complex requirements driven by domain, technology and end-user. However, a framework which integrates existing knowledge regarding the DSPML development by highlighting required building blocks has not yet been proposed as an idea which can serve/explain or give a solution to this issue.

BPM (Semantic Business Process Regulatory Compliance Checking using LegalRuleML) is the development of tools being able to design business processes graphically. However, nowadays BPM-s are also managed meaning that the process modeling tool has been integrated into some process management system which controls the process execution and integrates other parts of the information system. It is know that companies are subject to regulations. Non-compliance to such regulations would affect the added value of business processes and result to judiciary pursuits. When it comes to BPM, when checking the compliance of a business process by

considering relevant regulations, it means identifying if a process violates or not a set of norms.

To ensure business processes are compliant we need two components:

- A conceptually sound formal representation of a business process
- A conceptually sound formalism to model and reason with the norms derived by the regulations

The purpose of this paper is to build a framework starting form a specific problem analysis and requirements so could potentially fits the customer needs in the market. The paper is organized in 4 parts. Section 2 and 3 will give a brief overview of all related studies and literature reviews in the domain specific language focusing mainly in studies on LegalML Framework.

In this framework, is analysed an application of the semantic business process regulatory compliance checking the semantics of the LegalRuleML regarding the representation of the norms and its dynamics. Different ways to analyse and model the semantics of norms and their dynamics. There is also shown/explained how the semantic modelling phase can address, improve and also answer the needs for compliance checking of the companies.

Section 4 will address to all the techniques and technologies needed to design possible patterns and entity relation mappings taking in account GDPR as legal domain. Section 5 will discuss results and future work.

2. Terminology

The question what exactly is a domain-specific language is subject to debate. We propose the following definition:

"A domain-specific language (DSL) is a programming language or executable specification language that offers, through appropriate notations and abstractions, expressive power focused on, and usually restricted to, a particular problem domain."

The key characteristic of DSLs according to this definition is their focussed expressive power. Our definition inherits the vagueness of one of its defining terms: problem domain. Moreover, we refer to [9], which contains an interesting discussion contrasting a "domain as the real world" point of view as adopted in the artificial intelligence community, with a "domain as a set of systems" approach, as used in the systematic software reuse research community. DSLs are usually small, offering only a restricted suite of notations and abstractions. In the literature they are also called micro-languages and little languages [10].

Sometimes, however, they contain an entire general-purpose language (GPL) as a sublanguage, thus offering domain-specific expressive power in addition to the expressive power of the GPL. This situation occurs when DSLs are implemented as embedded languages (see Section 5). Domain-specific languages are usually declarative. Consequently, they can be viewed as specification languages, as well as programming languages.

A common term for DSLs geared towards building business data processing systems is 4th Generation Language (4GL). Related to domain-specific programming is end-user programming, which happens when end-users perform simple programming tasks using a macro or scripting language. A typical example is spreadsheet programming using the Excel macro-language.

3. Related Studies

We give some general evidence to the Business Process Management (BPM) and an introduction to the DSL literature. More about specific references is given at appropriate issues throughout this article rather than in this section.

Until recently, Domain-specific process modelling has gained increased attention, since traditional modelling languages struggle to meet the demands of highly specialized businesses and there are a few books on the subject. We mention paper [5], a framework

for creating domain-specific process modeling languages. In this article we have explored two frameworks for developing a domain specific language, a meta-model framework and a framework based on RuleML called LegalRuleML.

To study the first framework we referred in [6], a collection of articles on designing the digital transformation especially designing a framework for the development of domain-specific process modelling languages which indicate a framework that is a result of 23 requirements from relevant literature and contains essential building blocks that need to be considered during the development process.

Frank, U[7], focuses on the requirement analyses defining the scope and purpose of the DSPML, stakeholder and building blocks; and Barron[8], focused on defining guidelines to support a DSL developer to achieve better quality of the language design and a better acceptance among its users.

To study the LegalRuleML we referred on those articles that are focused in capturing alternative interpretations or renderings of a legal source we suggest to referred LegalRuleML framework. LegalRuleML gives the possibility of mutually incompatible renderings of a legal source to coexist in the same LegalRuleML document, and provides facilities to identify the interpretations and to select them. Legal documents are the source of norms, guidelines, and rules that often feed into different applications.

Guido Gorvenatori presents the principles of the OASIS LegalRuleML applied to the legal domain and discusses why, how, and when LegalRuleML is well-suited for modelling norms. Ho-Pun Lam shows how the semantic annotations can be used to empower a business process (regulatory) compliance system and discusses the challenges of adapting a semantic approach to legal domain.

4. DSL Design Methodology

The purpose of this paper is to analyse and study a computer system, dedicated to the management of internal GDPR privacy consulting with the use of relational DBs and specific domain languages (DSL). We need to understand why it is important to develop a specific domain language that refers to internal GDPR privacy consulting and BPM mapping and how to develop a domain specific language.

To develop a domain-specific language, we will focus on a legal framework such as the GDPR LegalRuleML which consists of capturing alternative interpretations or renderings of a legal source. LegalRuleML allows mutually incompatible interpretations of a legal source to coexist in the same LegalRuleML document and provides functionality to identify and select interpretations.

4.1. GDPR LegalRuleML Framework

Developing a domain specific language is not an easy process. It required both developing and analysing knowledge. The process of developing a specific domain language, pass through some essential steps like decision, analysis, design and implementation.

Decision: The decision is the first phase of DSL development. This phase is connected with "when" part of the DSL development. Deciding in favour of a new DSL is usually not easy. Investing in the development of DSL must pay for itself after more economical development and / or maintenance of the software.

Analysis: In the DSL development analysis phase, we will identify the main problem and gather the knowledge of the domain. In this project the main problem in the generation of GDPR documents and in the automation of internal processes. To do this it is necessary the contribution of experts of the domain or the availability of documents or code from which it is possible to obtain knowledge of the domain. We have selected domain analysis methodologies such as those described in the third phase, design phase.

Design: The design phase it will be followed by a model-based design, in which the design models are a distillation of common wisdom in the organization of the structural parts, of the grammar and of the constraints of a language and of the orthogonality criteria, in which the constructs of language are independent of each other, thus

allowing their systematic combination. Within GDPR LegalRuleML we introduce five design models.

- *container*, which is a structure of elements with independent existence (eg <Context> can include several <Association> sub-elements;
- *collection*, a sub-frame of the container that comes in the form of a list of elements of the same type (eg. <Roli> or a sequence of <Roli> elements);
- *recursive element* (for example, <Obligation> may include other <Obligation> elements);
- *marker*, an element that uses the @same attribute. Regarding the identification of a source, e.g. <lrml: LegalSource key = "" sameAs = "" />
- *composite* elements made up of different dependent parts (eg a <Rule> rule) consists of a previous <if> and a <then> conclusion).

Implementation: The implementation models we have identified are interpreter, application compilation / generation.
In the interpreter model, DSL constructs are recognized and interpreted using a standard fetch-decode-execute cycle. This approach is appropriate for languages that have a dynamic character or if speed of execution is not a problem.

The advantages of interpretation over compilation are greater control over the execution environment and a simpler extension. While, in compiling / generating applications, DSL constructs are translated into basic language constructs and library calls. On the DSL program or on the specifications, a complete static analysis can be performed. DSL compilers are often called application generators. The principles we are considering for building the GDPR LegalRuleML framework are the following.

Multiple semantic annotations: a legal rule can have multiple semantic annotations, where these annotations represent different legal interpretations. Each annotation appears in a separate annotation block as internal or external metadata. A set of parameters provides the interpretation regarding the origin, the applicable jurisdiction, the logical interpretation of the rule and others.

Linking rules and provisions: GDPR LegalRuleML includes a mechanism, based on IRI, which allows many (N: M) relationships between rules and textual provisions: more rules are incorporated in the same provision and different provisions contribute to the same rule. This mechanism can be managed in the metadata block, allowing an extensible management, avoiding the redundancy in the definition of IRI and avoiding errors in the associations.

5. Implementation

To begin with the design of the data management process, the first step is logical data modeling, which is a method of discovering the data, relationships and rules of a company, collectively defined as a business rule, and forms the basis of physical data modeling, which deals with aspects of the physical development of the database model.

The work for logical data modelling usually begins in the requirements analysis phase, directly when the project team studies the business requirements. Starting from the initial requirements and after subsequent detailed analyses, system analysts construct an initial data model for the representation of company data and processes. Extraordinary, on the transition between the systems analysis phase and the systems design phase, the data model is improved and obtains other details. Finally, in the systems design phase, the data model is established in a final version and the changes must be confirmed both by the customer and by the project team.

Changing the data model in the later stages of development or testing is not a good thing at all, especially when it comes to relational databases. Therefore, the logical data model should be defined from the beginning of the development of the system and it is not necessary to change it later. Relational databases, as indicated above, offer many advantages in terms of data integrity, consistency, transaction management, which are vital points in this project. Most modern applications must be able to recover data as quickly as possible. And that's when you can consider de-normalizing a relational database. In this project we have denormalized the focal points of the databases in order to optimize performance and improve data recovery.

The normalization process brings the data together in an organized way to eliminate redundancies, in other words, the denormalization process can be considered as the process of putting a fact in numerous places. This can have the effect of speeding up the data recovery process, generally at the expense of modifying the data. Instead of trying to de-normalize the entire database, in these automation projects we focused on particular parts in order to speed up the document generation process. However, developers should use this tool only for particular purposes.
We have used denormalization in the cases necessary to perform a calculation repeatedly during the queries, it is advisable to archive the results in the main table. Even in cases where a normalized database requires the merging of many tables to retrieve queries, we have added redundancy to the databases by copying the values between parent and child tables. This situation occurs when DSLs are implemented as embedded languages as this Domain-specific language. Consequently, they can be viewed as specification languages, as well as programming languages.

Mapping: The GDPR automation process is designed based on the legal framework GDPR LegalRuleML and the best practices that the regulation recommends to respect. The main purpose of GDPR automation is to help our corporate customers better align with the new EU GDPR legislation.

The first step in designing the data management process is to discover the data, relationships and rules of business logic. As the backbone of this system will serve the organizational module. This form will store data starting from basic information such as their name, site address, website, contact person and other detailed information related to the GDPR Regulation. For more information, the GDPR LegalRuleML requires organizations to legally keep records of processing activities under their responsibility and to make it available to the competent control authority upon request. For this purpose, a module, processing activity is required.

The form processing activities will represent the register of all databases processed by the organization. The set of data that they process and store in their organization will only be a record that will contain all the databases or data processed.

All data processed by the organization are analysed and classified into categories such as human resource data, research and development data, customer / supplier data or other particular data. The form of personal data categories will contain all the categories of data that a company processes. In other words, all databases are divided into n categories of personal data, which is a one to many relationship. A class of personal data categories will contain the databases that are processed, for example the human resource data categories will contain information on employees, health, CV candidates, etc. To speed up the data recovery process, the database register is de-normalized according to the automatic template system generation and also used in other modules that will be presented in the following section.

The denormalization of data in this case will improve the recovery of data from different related modules and through them we will be able to create expressions and workflows to more easily merge data from related modules. In terms of business logic, it is important to define correctly because it provides a framework not only on the classes of responsibility, but also through them we define the control of data access for each employee included in one of these categories of personal data and finally all the data that the respective categories can view, modify, store, process or delete.

4. Conclusion and Future Work

This article is an original investigation undertaken in order to acquire new knowledge that consists of a new specific domain language directed mainly towards a specific objective, practical or objective as the automation of management processes to respond positively to the GDPR and allow to exploit these data from the commercial point of view. The DSL created in this article is based on the GDPR LegalRuleML framework.

GDPR LegalRuleML is a framework that can be mapped to RDF triples to reuse the linked data. The GDPR automation process is designed on the basis of this framework and of the best practices that the regulation recommends to respect, which will help our business customers to better align with the new EU GDPR legislation. What we concluded during this project will guide our future work, that includes also the evaluation of the internal system with larger processes, and applying this methodology

with other kinds of regulations in order to make the system more flexible to the needs of the companies adopting it.

References

[1] Karsai, G., Krahn, H., Pinkernell, C., Rumpe, B., Schindler, M., Völkel, S.: "*Design guidelines for domain specific languages"*. In: Proceedings of the 9th OOPSLA Workshop on Domain-Specific Modelling (2009)

[2] Fellmann, M., Bittmann, S., Karhof, A., Stolze, C., Thomas, O.: Do we need a standard for EPC modelling? The state of syntactic, semantic and pragmatic quality. Lecture Notes Informatics (LNI), vol. P-222, pp. 103–117. Gesellschaft fur Inform (2013)

[3] Jannaber, S., Dennis M. Riehle2, Delfmann, P., Thomas, O., Becker, J.: "*Designing a Framework for the Development of Domain-Specific Process Modelling Languages"* (2017)

[4] Athan, T., Governatori, G., Palmirani, M., Paschke, A., Wyner, A.: LegalRuleML: "*Design Principles and Foundations"* (2015)

[5] Heitkotter, H.: "*A Framework for Creating Domain-specific Process Modeling Languages"* (2012)

[6] Maedche, A., vom Brocke, V.: *"Designing the Digital Transformation"* (2017)

[7] Frank, U [2011, 2013]

[8] Barron [2000]

[9] A. Basu, M. Hayden, G. Morrisett, and T. von Eicken. A language-based approach to protocol construction. In Kamin [43], pages 1–15. Reports on the design and implementation of Promela++, a DSL for protocol construction and validation. Promela++ adds domain-specific constructs to restricted C, and supports validation and optimization on the domain-level.

[10] J. L. Bentley. Programming pearls: Little languages. Communications of the ACM, 29(8):711–721, August

1986. Demonstrates and advocates the use of "little languages".