Legal Knowledge and Information Systems
J. Savelka et al. (Eds.)

doi:10.3233/FAIA241231

Decision Support in Law: From Formalizing Rules to Reasoning with Justification

Jeremy BOUCHE-PILLON [a], Nathalie AUSSENAC-GILLES [a], Yannick CHEVALIER [a,c] and Pascale ZARATE [a,b]
[a] *Université de Toulouse - CNRS - IRIT, France*
[b] *Université Toulouse Capitole*
[c] *Université Toulouse 3 Paul Sabatier*
ORCiD ID: Jeremy BOUCHE-PILLON https://orcid.org/0000-0001-6923-9915, Nathalie AUSSENAC-GILLES https://orcid.org/0000-0003-3653-3223, Yannick CHEVALIER https://orcid.org/0000-0002-8617-4209, Pascale ZARATE https://orcid.org/0000-0002-5188-1616

Abstract. With the emergence of the digital transition, the need to control the processing of digital information has significantly increased. In the EU in particular, Law Enforcement Agencies (LEAs) are caused to exchange information. In recent years, many regulations have emerged to control data processing and exchange. Texts other than the GDPR, such as the "Law Enforcement Directive (LED)", appeared to regulate specifically their data processing. And although many new formalisms have emerged to represent legal norms and rules, few are provided with a reasoning mechanism. The explainability of the results of systems using these formalisms also remains a major issue when dealing with critical decision situations. This paper aims to propose a framework to operate formal rules from regulations and guide a user in its decision process in a situation of data processing by LEAs by focusing on both the operability of the rules through reasoning and the explainability of the results from the reasoning.

Keywords. decision support, formal rules, operability of rules, reasoning mechanism, explainability

1. Introduction

With the advent of the digital transition and the increase in data volumes in many domains, protecting the privacy of people's information and ensuring their lawful usage has become more and more complex. This has led to numerous new laws and norms at various levels, from European directives to company charters. Among them the GDPR [1] from 2016 is the reference text that covers *"the protection of natural persons with regard to the processing of personal data and on the free movement of such data"*. Although GDPR is relevant in most situations involving personal data, it does not apply to the processing of personal data by authorities responsible for proceedings relating to criminal offences. Yet Law Enforcement Agencies (LEAs) are often required to exchange and

process potentially sensitive information as part of cooperation between police forces. As a result, other regulations have emerged that regulate procedures involving the exchange and the processing of personal information as part of investigations by LEAs, thus vastly increasing the complexity of information protection in such contexts.

As a consequence of this complexity, it has become challenging for law experts to work with so many sources. To assist them in their work, the question of formalizing regulations has emerged. Regulations are thus formalized to achieve different purposes: to document retrospectively the arguments presented and decisions made in court cases and create synthetic analyses, or more generally to document and help understand the legal texts themselves, in order to make them operational and guide law experts when they need to make a decision. Although many formalisms have emerged in recent years to represent legal norms and rules, few are accompanied by a mechanism for reasoning from the rules written in these languages. Another major issue in decision support systems is their inability to give a satisfying explanation of their decision.

Based on these observations, this paper focuses on two aspects of the formalization of legal rules: (i) The operability of rules through reasoning; (ii) The explicability of the results of reasoning. We propose a framework that supports the implementation of a decision support system to check the conformance of a context to a set of rules taken from regulations. To illustrate the design and use of this framework, we selected a use case about checking the conformance of data sharing between LEAs to several European regulations.

2. Related Work

In recent years, numerous works have been done to formalize the Law through languages, models and standards to achieve a variety of purposes, from facilitating the archiving and search of legal cases to assisting legal experts in their decision-making processes. For instance, "Reified Input/Output Logic" [2] combined Input/Output Logic [3] used in normative reasoning with reification [4] that can be used to convey many linguistic aspects of natural language into a simple logical formalism. Later works focused on compliance checking in reified I/O logic [5] using Shapes Constraint Language (SHACL), a W3C standard for validating and reasoning with RDFs/OWL.

Similarly, other approaches proposed a formalism using semantic web languages for reasoning. Examples of it can be found through the development of a policy language and an event log vocabulary within the framework of the European SPECIAL project [6], or the ontological representation of normative requirements from Gandon *et al.* [7]. Exploiting their ontology Normative Requirements Vocabulary (NRV)[1], they used SPARQL[2] to formalize norms. An advantage of this approach is that SPARQL implementations are already widely used and accessible through triplestores, APIs and storage engines. The Semantic Web Rule Language (SWRL)[3] [8] is a W3C proposal for a rule interchange format combining ontologies in OWL, more specifically its Description Logic (DL) subset, with an XML format for rules in the Unary/Binary Datalog subset of RuleML. A review work from [9] mentions three implementation approaches of inference engines

[1] https://ns.inria.fr/nrv/v1/nrv_v1.html
[2] https://www.w3.org/TR/sparql11-query/
[3] https://www.w3.org/submissions/2004/03/

for SWRL: Hoolet[4], Bossam[5] and Pellet [10] but unfortunately it was noted that SWRL reasoners are too weak to be effectively used [9].

Work from [11,12] also directly relies on OWL reasoners like Pellet [10] and inference mechanisms to implement legal compliance checking at an ontology level. However this approach requires an expertise in ontology design and might prove hard to maintain when needing to add, delete or modify rules.

The Rule Interchange Format (RIF)[6] [13] was submitted to the W3C with the goal of developing an extensible rule interchange format for the Web. The Integrated Rule Inference System (IRIS) [14] is an open-source Datalog engine, extended with XML Schema data types, built-in predicates, function symbols and Well-founded default negation, that is notably compatible with RIF rules. However, according to [9], RIF suffers from similar limitations as SWRL.

These observations on SWRL and RIF led to the creation of The Legal Knowledge Interchange Format (LKIF) [9], based on XML. LKIF provides a formalized syntax while allowing to write natural language sentences inside its predicates, making its rules easily readable by humans. Unfortunately, tests with the inference engine CARNEADES[7] supposedly compatible with LKIF proved unsuccessful with any LKIF rule. LegalRuleML[8] [15,16] is based on the Rule Markup Language (RuleML)[9] [17]. It has been applied to the GDPR and the resulting repository, the knowledge base DAPRECO[10] [18]. Although LegalRuleML was used to represent legal knowledge, it lacked reasoning mechanisms that external works tried to create [19].

Although these state-of-the-art solutions help represent information about legislation and legal cases, the usefulness of most of these languages in decision-support systems appears limited either by the lack of fully developed, widely accessible dedicated reasoning engines or by their complexity to implement.

3. Decision Support Framework

Given the lack of explicable reasoning frameworks over formal legal rules, this article proposes a decision support framework based on Marakas' model [20], which will ensure both the operability of formal rules and the explicability of reasoning results.

3.1. Framework Architecture

The decision support system framework, illustrated in figure 1 is composed of the following elements:

- The input is a context description for which users want to check the conformance to legal rules. Users describe it through a form.

[4] http://owl.man.ac.uk/hoolet/
[5] https://bossam.wordpress.com/about-bossam/
[6] https://www.w3.org/TR/rif-overview/
[7] https://carneades.github.io/about-carneades/
[8] https://docs.oasis-open.org/legalruleml/legalruleml-core-spec/v1.0/legalruleml-core-spec-v1.0.html
[9] https://www.ruleml.org
[10] https://github.com/dapreco/daprecokb

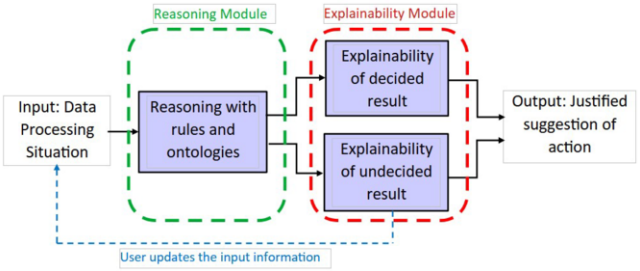


Figure 1. Schema of data processing situation framework

- A first module performs reasoning over the input context based on the formal rules extracted from the regulations applicable to this context.
- A second module is in charge of justifying the outputs from the reasoning module, with a distinction between two different cases: (i) when the reasoning module is able to decide on what action to take and (ii) when it is unable to do so. In the latter case, an interaction with the user will be engaged.

Figure 2 gives more details about the framework components and the module outputs: The reasoning module relies on an ontology (presented in [21] [11]) built from concepts and relations required to represent both the regulations and the input information. This module needs to be adapted to each use case, by formalizing the applicable rules taken from the reference regulation. The action described in the input situation description can be respectively obligatory, prohibited or only allowed. These cases are considered "decided". However, when no single formal rule from the rule base has been triggered the case is considered "undecided", and if several rules are respected but give contradictory answers, it is considered "contradictory" (yellow on the schema). The latter two cases are considered "undecided". When confronted with an "undecided" result, users are invited to indicate whether they wish to update their input, while receiving guidance on the modifications to be made to the context description that are most likely to lead to a "decided" result.

3.2. Use case: checking data sharing conformance to various European regulations

To illustrate how to use of this framework, we present a use case where the context is the sharing of datasets between various Law Enforcement Agencies (LEAs), and the regulation applicable to this context is a set of articles from three regulation documents that we selected in 2022 with the help of a doctoral student in digital Law. More specifically, the 15 most relevant articles from these texts were used as basis for this use case:

1. The *Directive (EU) 2016/680 of the European Parliament and of the Council of 27 April 2016 on the protection of natural persons with regard to the processing of personal data by competent authorities for the purposes of the prevention, investigation, detection or prosecution of criminal offences or the execution of criminal penalties, and on the free movement of such data, and repealing Council Framework Decision 2008/977/JHA* more simply called *Law Enforcement Directive (LED)* [22].

[11] available at: https://github.com/JeremyBOUCHEPILLON/legalDataProcessing/blob/main/ontology/legal_data_sharing_v2.owl

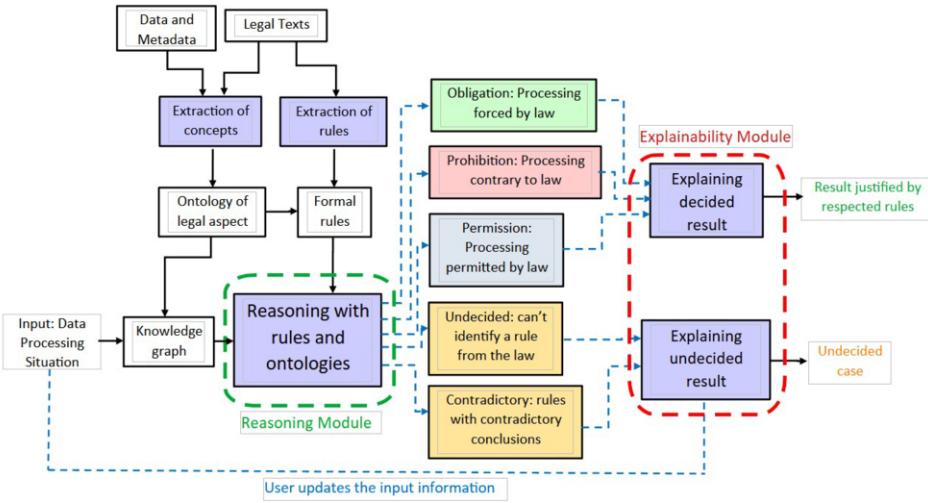


Figure 2. Detailed schema of the framework

2. The *Directive 2014/41/EU of the European Parliament and of the Council of 3 April 2014 regarding the European Investigation Order in criminal matters* [23]. This text regulates the "European Investigation Order" (EIO) procedure in which LEAs can issue or answer to an investigation order that can involve several investigation measures such as the acquisition of evidence data.
3. The *Regulation (EU) 2023/1543 of the European Parliament and of the Council of 12 July 2023 on European Production Orders and European Preservation Orders for electronic evidence in criminal proceedings and for the execution of custodial sentences following criminal proceedings* [24]. This text regulates the "European Production Order Certificate" (EPOC) and the "European Preservation Order Certificate" (EPOC-PR) procedures that allow LEAs to request or retain data originally stored in another organization.

Other regulations came into effect in 2023 and will later be included in the rule base.

3.3. Regulation Formalization

Regulation formalization is required to adapt the framework to the usecase. Users have to select the appropriate applicable regulation according to the context to be checked, and then to identify a set of rules that need to be formalized in the framework. Both the rules and the input context are represented using the concepts and properties of the ontology presented in [21]. This ontology provides a foundation to represent legal information and the context of cases.

It has been decided to use SPARQL to express the formal rules. However, for clarity reasons, in this paper, the formalization will be presented in a First-Order Logic (FOL) form following a $Conditions \rightarrow Effect$ syntax. The SPARQL equivalents of the FOL expressions are given on Github [12]. Moreover, the effects of rules in We adapted the

[12] https://github.com/JeremyBOUCHEPILLON/legalDataProcessing/blob/main/rules/

formalization approach of Gandon [7]. These rules do not indicate whether or not an action is permitted, prohibited or mandatory, but rather each rule is characterized as permission, obligation or prohibition. Reasoning aims at assessing the compliance of the input situation to each rule. Finally, for optimization purposes, it has been considered splitting the rules in two: a first part to determine if a legal rule is applicable to a context situation and a second part to determine if a situation is compliant with an applicable rule.

Making the rules operable requires a reasoning engine to check which rules are respected in a given situation. The rules being expressed in SPARQL, a SPARQL endpoint suffices as reasoning engine. In the current version of the framework, GraphDB is used, with the OWL-max ruleset for inference on the knowledge graph.

3.4. Input data : data processing situation description

The input form in the framework has to be adapted to the use-case and the applicable regulation, using concepts and properties from the ontology proposed in [21]. Since the format and vocabulary are the same as the one used to formalize rules, the input data is RDF-OWL to populate the ontology and generate a knowledge base. We use the TriG format to be able to manage named graphs.

An example of data processing situation is given listing 1 and its visualization in a graph in figure 3. In the case of data sharing between LEAs, the form requires to provide information regarding the data and actors involved in the procedures as well as the context in which the procedures occurs, for example if it is an urgent situation.

Listing 1: Input of Framework: description of a data processing situation in a LEA

```
{:Situation02 a lrmlmm:FactualStatement .}

GRAPH :Situation02 {
    :storage_change_02
        a :DataStorageChange;
        :involvesData :dataset_02;
        :hasIssueAuthorityAction :authority_1;
        :hasExecutionAuthorityAction :exec_auth_1;
        :isNecessary "true"^^xsd:boolean;
        :isAuthorizedLaw "true"^^xsd:boolean;
        :protectsVitalInterests "false"^^xsd:boolean.
    :dataset_02
        a :DataSet;
        :containsData :data_02;
        :hasOriginData :stor_entity_1;
        :hasDestinationData :authority_1.
    :data_02
        a :SensitivePersonalData;
        a :PrivateData.}
```

Each input is composed of two parts: (i) A single triple to be added in the default graph to characterize the situation described as a "Factual Statement"; (ii) The declaration of a named graph that will contain all the information related to the case situation.

Each named graphs encapsulates the information of each situation in different sub-parts of the knowledge graph, which can be used to limit access to information about only some of the situations.

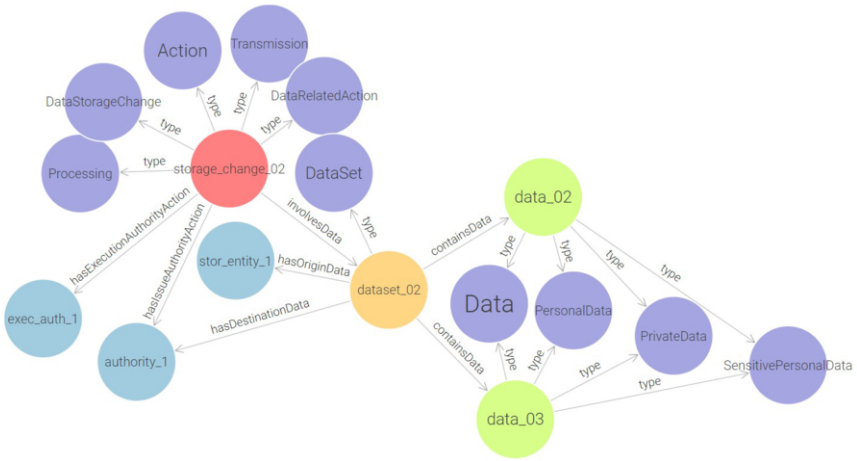


Figure 3. Knowledge graph containing information regarding a data processing situation

The graph view (cf. figure 3 allows to easily identify the information types and their relationships in a data processing situation. The red node is the root of the graph from which the rest of the graph has been developed. Purple nodes are classes to indicate the information types. Blue nodes are final nodes without type. Finally yellow and green nodes are intermediary nodes at different depths of the graph.

4. Handling the outputs of the reasoning mechanism

The reasoning mechanism can generate 5 different output valuess that derive from 3 types of results. First, cases where at least one rule is satisfied by the context situation, and all the satisfied rules give a coherent decision. Secondly, cases where no rule is satisfied by the context situation. Thirdly, cases where at least 2 rules are respected but their decisions are inconsistent with each other. For now, this part of the framework is mainly conceptual and has not been implemented yet.

4.1. Satisfied rules are consistent

Let's consider the case where at least one rule has been recognized as satisfied through the reasoning mechanism, and where the results of all satisfied rules are consistent. Satisfied rules are consistent if they give the same deontic answer regarding a given situation. Justifying the result of the system then consists of returning to the user the list of satisfied rules using their original legal text. For example: *"According to article 10 from the Law Enforcement Directive, the action you want to perform is permitted."* In cases where an action is forbidden, it can be relevant to initiate an interaction to the user, informing him of the specific reasons causing the refusal. Given these information, the user could re-specify some input data in order to get a positive answer.

4.2. No rule is satisfied

If no rule has been triggered, the system finds itself in an indecisive situation, unable to give the user an appropriate suggestion. An indication based on the rules closest to being triggered is then given to the user in an attempt to adjust the input information and to increase the chances of obtaining a definitive decision. The following procedure is used to determine the indication to be given to the user: (i) Only consider the subset of rules that are "applicable" to the input situation. (ii) Sort the rules of this subset in ascending order of the number of conditions not met in each of these rules. (iii) Keep the three first rules after sorting and modify the input form of the framework to highlight and comment the fields in it that need to either be completed or have a different value to become compliant with the rule. (iv) Return these forms to the user so he can consider whether or not he is able to provide the required information. (v) If the user is able to complete the input information, the system can return a definitive decision like in the first case, otherwise the situation stays undecided.

4.3. Satisfied rules are inconsistent

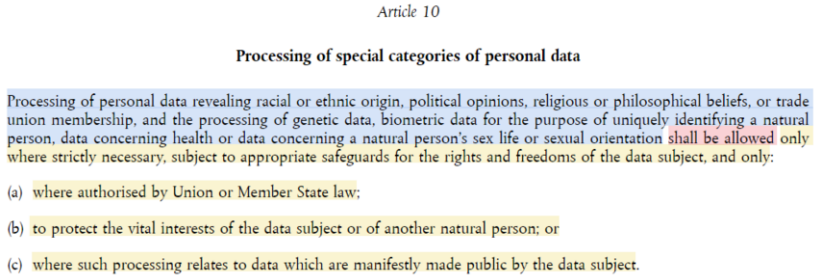
The third and final case is where several satisfied rules provide inconsistent results, i.e. they give contradictory deontic answers regarding the given situation. For instance, one rule may state that an action is "prohibited" while another one concludes that it is "permitted". This situation may occur for several reasons. It can reveal an error in a rule formalization, but also the primacy [13] of one rule over the other that has not been taken into account, like the *lex specialis* or *lex posterior* principles for example. In this case, the problem can be tackled through supplementary rules solving primacy issues, in a similar way as in LKIF [25]. When faced with this situation, extracts from the conflicting legal texts are presented to the user who is asked if all input information is correct.

5. Running example and results

We illustrate the various components of the framework through a detailed running example. This simple example will focus on confronting data processing situations to a single rule: Article 10 from the Law Enforcement Directive [22] (Figure 4). The test set consists of about twenty manually created data processing situations similar to the one in section 3.4 to test each condition of the rule.

The first step is to formalize this article, from which the following elements can be extracted: (i) The deontic class of this article is "Authorization", as indicate the terms *"shall be allowed"*. (ii) The object of the rule relates to the processing of what we could call *"sensitive personal data"*. (iii) The conditions of this rule are *"the strict necessity"* of the processing, the *"safeguard of rights and freedom of the data subject"* and a disjunction of conditions: *"allowed by Union or Member State"* **OR** *"protection of vital interests"* **OR** *"the data is public"*. A first attempt to translate this article using ontology classes and properties gives the following FOL expression. The SPARQL equivalent of this rule can be found on Github as *SPARQL_LED-10_v1.txt*.

[13] https://eur-lex.europa.eu/EN/legal-content/glossary/primacy-of-eu-law-precedence-supremacy.html



Article 10

Processing of special categories of personal data

Processing of personal data revealing racial or ethnic origin, political opinions, religious or philosophical beliefs, or trade union membership, and the processing of genetic data, biometric data for the purpose of uniquely identifying a natural person, data concerning health or data concerning a natural person's sex life or sexual orientation shall be allowed only where strictly necessary, subject to appropriate safeguards for the rights and freedoms of the data subject, and only:

(a) where authorised by Union or Member State law;

(b) to protect the vital interests of the data subject or of another natural person; or

(c) where such processing relates to data which are manifestly made public by the data subject.

Figure 4. Example of law article: article 10 from 2016/680/UE directive (LED)

$$Processing(action) \wedge InvolvesData(action, dataset) \wedge ContainsData(dataset, data) \wedge SensitivePersonalData(data) \wedge Necessary(action) \wedge SafeguardRights(action) \wedge (AuthorizedLaw(action) \vee ProtectsVitalInterest(action) \vee PublicData(data)) \rightarrow HasCompliance(LED10, situation)$$

When confronting the test set to this rule, the results were not as expected, notably some situations in which several data are involved, only some of which being sensitive, and only some of these sensitive data being public while the others are private, ended up marked as compliant with the rule, when they should not be. This observation led to analyze the formalization of the rule and highlighted a crucial element subject to interpretation that was missing in the formal form: the last part of the rule, *"where such processing relates to data which are manifestly made public by the data subject"* had been formalized as a simple predicate *PublicData(data)*. This predicate is evaluated as "true" as long as at least one of the data involved in the processing is public. This does not correspond to the desired rule behavior, as we instinctively understand that the purpose of this condition is to ensure that all sensitive data involved are public. This example highlights one of the major challenges when formalizing legal rules: the implicit quantifiers that are identified only when confronting the raw written text with our common sense. Indeed, the correct version of the rule is:

$$Processing(action) \wedge InvolvesData(action, dataset) \wedge ContainsData(dataset, data) \wedge SensitivePersonalData(data) \wedge Necessary(action) \wedge SafeguardRights(action) \wedge (AuthorizedLaw(action) \vee ProtectsVitalInterest(action) \vee (\forall data, SensitivePersonalData(data) \Rightarrow PublicData(data)) \rightarrow HasCompliance(LED10, situation)$$

Translating this new formal rule in SPARQL could not be done in a straightforward way, since there is no universal quantification in SPARQL. Instead, two nested negated existential quantifiers were used. Thus instead of literally representing "all data that is sensitive is made public", what is expressed is "No sensitive data is made public". The SPARQL equivalent of the corrected rule is on Github as *SPARQL_LED-10.txt* Confronted to the test set of data processing situations, this corrected rule performed as expected and all the correct situations were evaluated compliant with the rule after reasoning.

To prepare the explanation part of the framework, this rule was split in two parts to reason separately on the applicability and the compliance aspects. This results in the two

following FOL rules, the SPARQL translations of which can be found on the GitHub as *SPARQL_LED-10_applicable.txt* and *SPARQL_LED-10_compliance.txt*

$$Processing(action) \wedge InvolvesData(action, dataset) \wedge ContainsData(dataset, data) \wedge SensitivePersonalData(data) \rightarrow isApplicable(LED10, situation)$$

$$isApplicable(LED10, situation) \wedge Action(action) \wedge InvolvesData(action, dataset) \wedge ContainsData(dataset, data) \wedge Necessary(action) \wedge SafeguardRights(action) \wedge (AuthorizedLaw(action) \vee ProtectsVitalInterest(action) \vee (\forall data, SensitivePersonalData(data) \Rightarrow PublicData(data)) \rightarrow HasCompliance(rule, situation)$$

It can be noted that some conditions appear redundant in both rules, because keeping the predicates that link variables to each other is necessary in all rules.

6. Conclusion and Future Works

This paper presented a framework of a decision support system while focusing on two aspects: First, the need to use a language to formalize rules for which operability solutions are available to easily enable rule reasoning. Second, in a critical domain like the application of the Law, ensure that the decision-support system provides a satisfactory explanation of its result to properly help the user making an informed decision.

After reviewing some formalisms as well as the inference and reasoning engines compatible with them, the different components of the framework were presented, with the formal rules expressed in SPARQL. Finally, the principles and results of the framework were illustrated through a running example.

Future works involve implementing the explainability part of the framework to test the completed version. It also involves extending the number of formal rules in the framework. Indeed, new directives have come into force in 2023 regarding the data processing by LEAs [26]. Although current rules have been manually extracted from the regulations, adding new formal rules to the system would be an opportunity to automate rule extraction by using state-of-the-art Natural Language Processing methods [27,28,29]. Finally, while the formal language used in this study is SPARQL, other formal standards could be used to test and compare the results, like LegalRuleML [15,16] for example.

Acknowledgments

This work is partially funded by the H2020 project STARLIGHT[14] ("Sustainable Autonomy and Resilience for LEAs using AI against High priority Threats") that received funding from the European Union's Horizon 2020 research and innovation program under grant agreement No 101021797. We would also like to thank Ronan PONS, PhD student in Law, who assisted this work by providing his insight as legal expert.

[14] https://www.starlight-h2020.eu/

References

[1] Lex E. Regulation - 2016/679 - EN - gdpr - EUR-Lex.europa.eu. EU; 2016. Last changed: 04/05/2016, Last Accessed: 04/04/2024.

[2] Robaldo L, Sun X. Reified Input/Output logic: Combining Input/Output logic and Reification to represent norms coming from existing legislation. Journal of Logic and Computation. 2017 04;27(8):2471-503. Available from: https://doi.org/10.1093/logcom/exx009.

[3] Makinson D, van der Torre L. Input/Output Logics. Journal of Philosophical Logic. 2000;29(4):383-408.

[4] Davidson D. The Logical Form of Action Sentences. In: Rescher N, editor. The Logic of Decision and Action. University of Pittsburgh Press; 1967. p. 81-95.

[5] Robaldo L. Towards compliance checking in reified I/O logic via SHACL. In: Proceedings of the Eighteenth International Conference on Artificial Intelligence and Law. ICAIL '21. New York, NY, USA: Association for Computing Machinery; 2021. p. 215–219. Available from: https://doi.org/10.1145/3462757.3466065.

[6] Kirrane S, Fernández JD, Bonatti P, Milosevic U, Polleres A, Wenning R. The SPECIAL-K Personal Data Processing Transparency and Compliance Platform. ArXiv. 2020 1. Available from: https://arxiv.org/abs/2001.09461v3.

[7] Gandon F, Governatori G, Villata S. Normative requirements as linked data. In: JURIX 2017-The 30th international conference on Legal Knowledge and Information Systems; 2017. p. 1-10.

[8] Horrocks I, F Patel-Schneider P, Boley H, Tabet S, Grossof B, Dean M. SWRL: A Semantic Web rule language combining OWL and RuleML. W3C Subm. 2004 01;21.

[9] Gordon TF, Governatori G, Rotolo A. Rules and norms: Requirements for rule interchange languages in the legal domain. In: International Workshop on Rules and Rule Markup Languages for the Semantic Web. Springer; 2009. p. 282-96.

[10] Sirin E, Parsia B, Grau BC, Kalyanpur A, Katz Y. Pellet: A practical OWL-DL reasoner. Journal of Web Semantics. 2007;5(2):51-3. Software Engineering and the Semantic Web. Available from: https://www.sciencedirect.com/science/article/pii/S1570826807000169.

[11] Francesconi E, Governatori G. Legal Compliance in a Linked Open Data Framework. Frontiers in Artificial Intelligence and Applications. 2019 12;322:175-80. Available from: https://ebooks.iospress.nl/doi/10.3233/FAIA190321.

[12] Francesconi E, Governatori G. Patterns for legal compliance checking in a decidable framework of linked open data. Artificial Intelligence and Law. 2023 9;31:445-64. Available from: https://link.springer.com/article/10.1007/s10506-022-09317-8.

[13] Kifer M. Rule Interchange Format: The Framework. In: Calvanese D, Lausen G, editors. Web Reasoning and Rule Systems. Berlin, Heidelberg: Springer Berlin Heidelberg; 2008. p. 1-11.

[14] Bishop B, Fischer F. IRIS - Integrated rule inference system. CEUR WS Proceedings. 2008 01;350.

[15] Palmirani M, Governatori G, Rotolo A, Tabet S, Boley H, Paschke A. LegalRuleML: XML-based rules and norms. In: Rule-Based Modeling and Computing on the Semantic Web: 5th International Symposium, RuleML 2011–America, Ft. Lauderdale, FL, Florida, USA, November 3-5, 2011. Proceedings. Springer; 2011. p. 298-312.

[16] Palmirani M, Governatori G, Rotolo A, Tabet S, Boley H, Paschke A. LegalRuleML: XML-Based Rules and Norms. In: Olken F, Palmirani M, Sottara D, editors. Rule-Based Modeling and Computing on the Semantic Web. Berlin, Heidelberg: Springer Berlin Heidelberg; 2011. p. 298-312.

[17] Boley H, Paschke A, Shafiq O. RuleML 1.0: The Overarching Specification of Web Rules. In: Dean M, Hall J, Rotolo A, Tabet S, editors. Semantic Web Rules. Berlin, Heidelberg: Springer Berlin Heidelberg; 2010. p. 162-78.

[18] Robaldo L, Bartolini C, Lenzini G. The DAPRECO Knowledge Base: Representing the GDPR in LegalRuleML. In: Proceedings of the Twelfth Language Resources and Evaluation Conference. Marseille, France: European Language Resources Association; 2020. p. 5688-97. Available from: https://aclanthology.org/2020.lrec-1.698.

[19] Lam HP, Hashmi M. Enabling Reasoning with LegalRuleML. Theory and Practice of Logic Programming. 2018 09;19:1-26.

[20] Marakas G. Decision Support Systems in the 21st Century. Prentice Hall; 1999.

[21] Bouché-Pillon J, Aussenac-Gilles N, Chevallier Y, Zaraté P. An ontology for legal reasoning on data sharing and processing between law enforcement agencies. In: 3rd international workshop KM4LAW – Knowledge Management and Process Mining for Law (2024). IAOA; 2024. p. TBP.

[22] EUR-Lex. 2016/680 - EN - Law Enforcement Directive; LED - EUR-Lex. EU; 2016. Last changed: 04/05/2016, Last Accessed: 04/04/2024.
[23] EUR-Lex. Directive - 2014/41 - EN - EUR-Lex. EU; 2014. Last change: 13/03/2022, Last Accessed: 04/04/2024.
[24] EUR-Lex. Regulation - 2023/1543 - EN - EUR-Lex. EU; 2023. Last Accessed: 19/06/2024.
[25] Gordon TF. Constructing Legal Arguments with Rules in the Legal Knowledge Interchange Format (LKIF). In: Casanovas P, Sartor G, Casellas N, Rubino R, editors. Computable Models of the Law. Berlin, Heidelberg: Springer Berlin Heidelberg; 2008. p. 162-84.
[26] EUR-Lex. Directive - 2023/977 - EN - EUR-Lex. EU; 2023. Last Accessed: 19/06/2024.
[27] Fawei B. NLP-Based Rule Learning from Legal Text for Question Answering. Asian Journal of Research in Computer Science. 2024 Jun;17(7):31–40. Available from: https://journalajrcos.com/index.php/AJRCOS/article/view/475.
[28] Recski G, Lellmann B, Kovacs A, Hanbury A. Explainable Rule Extraction via Semantic Graphs. In: ASAIL/LegalAIIA@ ICAIL; 2021. p. 24-35.
[29] Ferraro G, Lam HP, Tosatto SC, Olivieri F, Islam MB, van Beest N, et al. Automatic extraction of legal norms: Evaluation of natural language processing tools. In: New Frontiers in Artificial Intelligence: JSAI-isAI International Workshops, JURISIN, AI-Biz, LENLS, Kansei-AI, Yokohama, Japan, November 10–12, 2019, Revised Selected Papers 10. Springer; 2020. p. 64-81.