Assessment of Rules as Code methods concerning the goal of assisting citizens personally with regulations.

Digicampus for GovTech4All, Bas Kaptijn & Jet Klaver

Definitive version, April 2024

Abstract

Rules as Code[1] (RaC) is a trend in which official law and regulations are interpreted or defined and published in a machine-consumable form, for the purpose of improving aspects like compliance and consistency in services provided to citizens. When assisting people with these services it is important that they are explainable and that all content is consistent and traceable towards how society agreed to these officially in a fine-grained way. In this paper an assessment framework is proposed to assess a range of current RaC solutions for the purpose of the so-called personal regulations assistant (PRA) application in mind as an example building block that utilizes RaC models. None of the evaluated solutions in this paper seem to be viable for such a PRA on their own, but solutions are more complementary than full alternatives of each other. Which indicates possibilities for new or improved standards to combine these solutions so that they are usable in combination. However, it is important to note, that the evaluation in this paper is limited. This paper is intended to help start a discussion on what is needed and how RaC can be assessed accordingly and introduce and share some innovative ideas around RaC within the context of the EU GovTech4All trajectory in which Digicampus is involved.

[1] As a primer on this topic we refer to the paper published by the OECD-OPSI : Mohun, J. and A. Roberts (2020), "Cracking the code: Rulemaking for humans and machines", OECD Working Papers on Public Governance, No. 42, OECD Publishing, Paris, https://doi.org/10.1787/3afe6ba5-en.

Introduction

The Netherlands is currently looking into the development of a personal advisor for citizens, a so-called Personal Regulations Assistant (PRA)2 because citizens frequently fail to apply for entitled benefits and services even though they are eligible. This is particularly crucial for individuals living in poverty, as receiving these benefits could have a substantial impact.

The assistant will be able to help people identify their needs through finding, assessing, and utilizing the regulations and the products and services that can help citizens engage with their needs. The PRA is capable of assisting them, because one of the main aspects within the PRA is compliance by design. Utilizing compliance by design ensure accuracy and makes it possible for anyone to track in a detailed way the correctness of the information, because it provides by pointing out what specific statements in official laws and regulations it is based on. Additionally, this aligns with the Dutch government's aspiration to transition towards a more Open Government.

Moreover, in its proposed design, the assistant will be able to simplify assessing someone's personal situation in regard to a regulation. With the use of Self Sovereign Identity (SSI) and personal data wallets,34 citizens can enter their personal data, without having to bother with gathering and/or entering a lot of information by hand. However, even though they are essential to the PRA, wallets and SSI are out of scope for this paper. This paper's focus is Compliance By Design of explanatory textual information and machine consumable rules as code, both in relation to the PRA.

Compliance by the Design within the PRA

The PRA will need machine consumable models that are in strong relation to official law and regulations and contain everything necessary to give citizens proper advice, which amongst other things should be complete, clear, consistent, compliant, manageable, and respectful.

Whilst the creation of these models probably can very well be assisted by AI techniques, there is no good standard or set of good enough examples as to what should be contained in these models and therefore a lack of good training data. Within the PRA, these rules are only to be used to advise people and never for automated decision-making without also giving people the means to evaluate these rules for themselves first and have them explained adequately. Moreover, there should be an option to ask for and receive timely and effective human support in case there is a conflict. In this context, a conflict arises when someone feels or thinks the outcome of the advice or the explanation of law and regulation is not right or fair. Whilst RaC cannot really be called AI (it

[2] At time of writing this innovation project is about to scaled up under a new name. https://www.digitaleoverheid.nl/overzicht-van-alle-onderwerpen/dienstverlening-aan-burgers-en-ondernemers/persoonlijke-regelingen-assistent/

[3] Personal data wallets can continuously gather and handle information from multiple sources in bulk by making use of the GDPR-right on data portability as best practice, even for governments where the GDPR exempts them from the obligation to abide by this. See https://ec.europa.eu/newsroom/article29/items/611233 *"Article 20(3) and Recital 68 provide that data portability does not apply when the data processing is necessary for the performance of a task carried out in the public interest or in the exercise of official authority vested in the data controller, or when a data controller is exercising its public duties or complying with a legal obligation. Therefore, there is no obligation for data controllers to provide for portability in these cases. However, it is a good practice to develop processes to automatically answer portability requests, by following the principles governing the right to data portability. An example of this would be a government service providing easy downloading of past personal income tax filings. For data portability as a good practice in case of processing based on the legal ground of necessity for a legitimate interest and for existing voluntary schemes, see pages 47 & 48 of WP29 Opinion 6/2014 on legitimate interests (WP217)."*

[4] Note that current personal data / SSI wallet projects are mostly only capable of doing actions on single or few credentials at a time and therefore still will not result in a minimal viable user experience, so innovation in this domain is still necessary even now a first reference implementation of the eID wallet has been published.

Co-funded by the European Union

depends on your definition of AI), similar problems can arise with RaC as with AI, even when only advising people. Fairness cannot be automated5. Also not being able to find a solution for one's needs is already to be seen as a conflict. In case of such conflict, a citizen should never get into a situation where he cannot fulfill his basic needs or sustain his way of life as before due to this conflict resulting from automated decision making. A citizen should be able to experience a government that takes these conflicts seriously, and engages in compassionate manner with them until a solution or understanding amongst all parties involved is reached, not enforced.

For the part of the functionality of the PRA in which calculations (including the use of decision trees) are made, Rules as Code methods and techniques come into play. And there are a vast number of them in quite varying forms6. Note that, besides making calculations an important part of the advisor will be providing a clear and compliant explanation of regulations. This explanation should be understandable for both stakeholders developing the models and also more user centered: the citizens and their care-takers. While this is often not the main focus of RaC methods, especially not with this biggest group of users in mind, we do take this into account as it seems essential to the goal of assisting citizens with regulations and gaining their trust while doing so.

This paper examines some existing RaC methods known within the Dutch and French governments, to assess them in relation to realizing a qualitative PRA within the scope of RaC. By evaluating them using a PRA assessment framework as proposed in this paper we aim to determine their suitability for the PRA. Decision can be made based on the proposed aspects and how they differ, overlap, or compliment. Finally, this paper discusses the findings and proposes future work.

This limited research has been conducted as part of Work Package 2, Pilot B of the GovTech4All7 trajectory in which Digicampus is involved in. As the PRA project8 is chosen as a use case, it also is input for the development of the PRA solution. More information about the current status of the PRA project can be obtained by contacting Digicampus.9 After a first final concept version, some feedback and additional information and improvements has been processed resulting in this definitive version of this paper.

Disclaimer and limitations

The solutions selected for assessment are those collected on https://regels.overheid.nl in September 2023, which might be an arbitrary list. We have added three solutions listed within the French government on https://code.gouv.fr/fr/explicabilite/ to this list by hand. Also afterwards
 we added Datalex.org because of its characteristics. If a solution is not included in this selection, it does not mean anything.

We assessed selected solutions mostly through desk research, not by applying the solutions to a given problem yet. We only have some experience with this with some of the solutions, namely CALCULEMUS/FLINT, ALEF, Wetsanalyse, RuleManagement, RuleSpeak and, OpenFisca. Which is why these six are analyzed more in depth. With this research, we want to start the discussion on what is needed and how RaC can be assessed accordingly and introduce some innovative ideas around this. Moreover, as this is a conversation starter, we did not yet

[5] Sandra Wachter, Brent Mittelstadt, Chris Russell, Why fairness cannot be automated: Bridging the gap between EU non-discrimination law and AI, Computer Law & Security Review, Volume 41, 2021, 105567, ISSN 0267-3649, https://doi.org/10.1016/j.clsr.2021.105567.
[6] https://publications.cohubicol.com/typology/
[7] https://joinup.ec.europa.eu/collection/govtechconnect/govtech4all , https://joinup.ec.europa.eu/collection/govtechconnect/news/decoding-govtech4all-partnerships-goals-and-digital-landscape
[8] https://www.digitaleoverheid.nl/overzicht-van-alle-onderwerpen/dienstverlening-aan-burgers-en-ondernemers/persoonlijke-regelingen-assistent/ (Dutch)
[9] info@digicampus.tech

Co-funded by the European Union

discuss the assessment in this paper with most of the organizations behind the assessed solutions. Except for CALCULEMUS/FLINT and ALEF, they did not get a notice or the opportunity to give us feedback in depth or have us correct potential errors we made assessing the solutions just through desk research. This should be considered when interpreting the results in this paper.

PRA RaC Assessment Framework

First, we define which scope the solutions have. This can be one or a combination of the following uses for the RAC solution. These uses are: interpretation, analysis, interpretation explanation, user-centered and/or coding. Secondly, we identified eight distinct aspects on which the solutions could deliver. These aspects are: traceability, explainability, compliance, traceable Identifiability, transparency, easy to integrate, affordability and maturity. With this framework we think these solutions can be assessed and discussed side by side.

Assessments of solutions are a set of scores on the aspects above which can only have three values, except for the first aspect (Scope). A solution either delivers on an aspect (☑), might deliver on it while it is clear it is not intended to do that and possibly requires some adjustments (½), or it just does not deliver on the aspect, and it is likely it will not (be feasible), to be indicated by a blank value. For the Scope, a score is indicated by the types of actions the solution supports by enumerating the applicable actions in the form of the letters A, I, E, U, and C as stated in the description of this aspect.

An ideal RaC solution for a PRA type of service would support all types of actions and deliver on all other aspects. We deem a minimally viable solution to support all types of actions and to deliver on all other aspects except for Traceable Identifiability, as a solution could be made usable without it, and it has a dependency on other technological trends like Self Sovereign Identity. Note however that more advanced and highly desired functionality, like automated and easy to use test and proactive notification functionality based on RaC models and locally gathered personal data in verifiable credential wallets will require Traceable Identifiability.

Scope

When crafting new RaC models, whether based on new or existing rules in natural language, we distinguish several types of tasks that take place, whether this is done consecutively, iteratively or in parallel, deliberately, or implicitly in an ad hoc way. There a few solutions mentioned as a RaC solution that focus on only some of these tasks. This means RaC solutions can sometimes be used in combination with each other. For instance, a lot of solutions only involve the interpretation of rules expressed in natural language or some intermediate form that is easier to codify into rules in the form of machine executable code. We denote these as having a scope on **Interpretation (I)** when these solutions in some way describe how to do this.

However, instead of a purely rule-based approach focusing on just the parts that clearly are computable, some RaC methods primarily focus on a more higher level model-based approach to describe the normative system described in law as a conceptual model in a machine consumable form. In that case, a well defined analysis works as a first step in order to continue with the interpretation of only smaller parts within the model like in a rule-based approach.

There are also solutions that only focus on this type of analysis in order to model a normative system. These solutions have identified conceptual metamodels with which any piece of rules as natural language can be modeled just by denoting parts of the text of laws and regulations as being what concept it denotes within the conceptual model linking them together and often these solutions developed a methodical way to do this kind of modeling/analysis. Sometimes with accompanying tools and standards. This task of analysis does not yet involve interpreting the model of linked concepts and coding them in a formal language that is calculable except for only simple Boolean logic. A lot of RaC solutions do not specify how to do this type of analysis or even interpretation at all. This is left entirely for the expert using the RaC solution.

It is important to note that when directly codifying rules from natural language, a normative system gets modeled anyways whether this is done consciously and or methodically or not. Often human experts can or are to be hired as a service to use RaC solutions, but we see that as the solution itself (apart from human experts) does not have a clearly defined view on this kind of analysis, even though they might hold this kind of knowledge for themselves. We denote solutions that openly prescribe a way to do this kind of analysis in detail as having a scope on **Analysis (A)** though without much evaluation of how usable this analysis is.

Apart from this, we distinguish an action of explaining (identified concepts within) rules in another natural language description in which it is explained how the original expression of rules are interpreted (if not extremely obvious), though often this type of explanation is only for and by the legislative and technical experts involved in the modeling process and only logged as comments. We denote solutions that promote logging these kinds of explanations of interpretation as having a scope on **Interpretation explanation (E)**.

Solutions that also focus on making these explanations suitable and part of the model for the purpose of presenting them to all users (including citizens, for instance) in relation to original rules and or code are denoted as **User-centered (U)**. To be inclusive as possible this should involve having multiple different forms of explanation for distinct types of end users or on multiple different levels of detailedness, also always in detailed reference to parts of the original rules in natural language. Together with codified rules, this could also involve the task of defining interactive content, for instance, a calculation of eligibility for a benefit based on information in a verifiable credential wallet together with how to provide this and explain this to the user.

Lastly, we denote solutions that prescribe how to codify rules (without specifying methodically how to do analysis and interpretation) as having a focus on **Coding (C)**. In that sense, all programming languages could be seen as RaC solutions having this scope, though these programming languages are not created, optimized, and scoped with this purpose in mind and therefore not to be seen as RaC solutions. Optimally a solution, or a combination thereof, would have a scope on all of these tasks and denoted with all the corresponding letters: A, I, E, U, C.

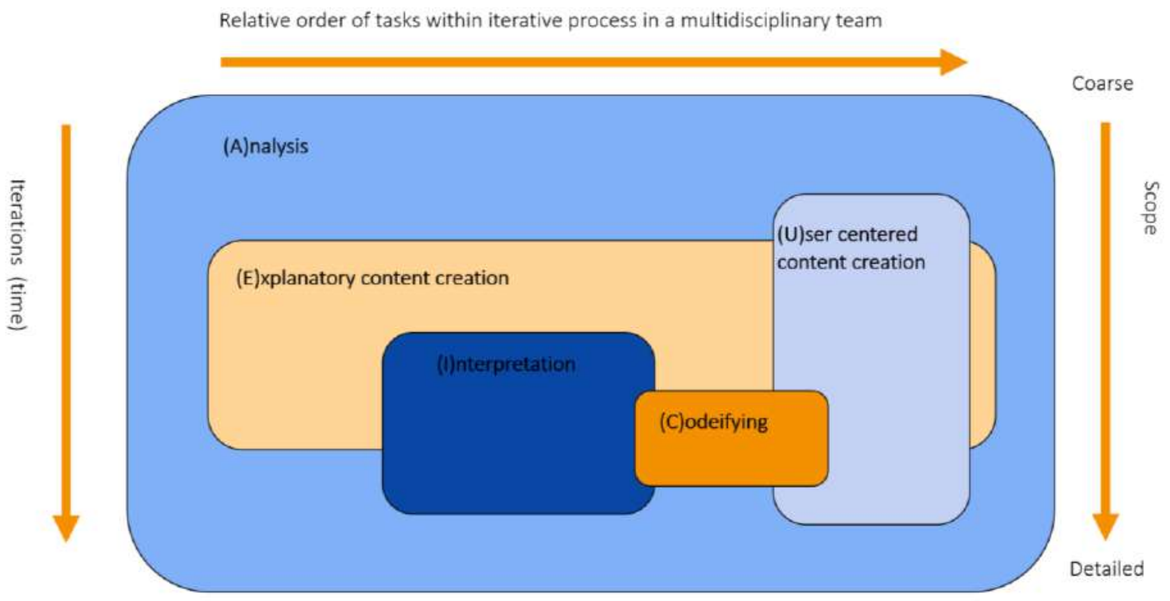
Relative order of tasks within iterative process in a multidisciplinary team
Coarse
Scope
Detailed
Iterations (time)
(A)nalysis
(E)xplanatory content creation
(U)ser centered content creation
(I)nterpretation
(C)odeifying

Figure 1, relation between different tasks RaC solutions are scoped on in terms of sequentially and possible detailedness

Co-funded by the European Union

Figure 1 illustrates how these different tasks relate in terms of relative sequentially within the process of creating content based on a normative system described in natural text and also in relation to the scope, from coarse to more detailed. Creating traceable explanatory text for all kinds of users can be done both in a very summarized way and involving parts that otherwise might not even get codified. Whereas codifying rules into machine consumable form is often done in an all-or-nothing detailed way which might make it less affordable to do. Note that Analysis towards a model could also be done in the form of codifying them in terms of the concepts of the conceptual model with which a normative system can be described in a model-based approach. RaC solutions that are designed this way, and define this model-based approach in this manner, are also denoted with a scope on Analysis when they can be applied to any text in the form of natural language describing a normative system in a generic way. The possibility to extend a RaC solution to also include this does not count as it has a focus on Analysis (yet). Out of scope of this assessment is that while it is obvious to start with doing the tasks of analysis, it seems best to practice performing all tasks concurrently, in a multidisciplinary team to be most effective. Also, where there are existing interpretations (or code) already used in production but which are not traceable to law, one might also choose to reverse engineer this traceability and fix errors and misinterpretations where they are encountered.

Aspects

1. Traceability: Using the solution results in models that can also include machine-consumable information on the traceability of rules and/or explanatory text or even interactive content towards all the specific statements within the official law and regulations in natural language they are based on.

2. Explainability: The solution includes a method and guide for constructing compliant natural language besides code, for instance, an explanation of regulation for citizens (including lawmakers and developers) in multiple levels of detail including the explanation of their traceability. Moreover, it should help construct explanations that are complete, clear, consistent, compliant, manageable, and respectful. Often RaC solutions only focus on explainability for stakeholders included in the construction of models, not for end users. In such cases, the solution does have some but not optimal explainability. This is then indicated in the Scope by having a scope on Explanatory content creation but not User-centered content creation.

3. Compliance: The solution aims to include a well-defined and proven method and accompanying guidance for analyzing existing law and regulations in official natural language. As is commonly used within law within at least the EU for the construction of rules in the form of code (or explanatory content).

4. Traceable Identifiability: The method includes the identification of regulations and information types (definitions) and can identify them uniquely by using some sort of universal unique identifier that can be officially recognized. Hence they can be referred to in other models and as such can be related to other regulations or information types in other models (of other regulations). Think of having verifiable credentials in SSI Wallets with information being the combination of data as value for a data field that refers to its definition within regulations in a broad sense (which can include specifications). Additionally, think of rules that specify relations between different variants of the same information, for instance, in Dutch law, there are numerous slightly different definitions of what is seen as having a "partner" dependent on context. RaC methods having this aspect will make it much easier to give

Co-funded by
the European Union

personalized advice, match provided verifiable information to what is mentioned in rules and prevent the need for creating and documenting new definitions in systems architecture as something different from the rules evaluated in context of these systems and which are already defined in law and regulations.

5. **Transparency**: The methods result in models that can be published as open source and are usable for anyone without getting locked into a dependency to proprietary solutions to evaluate and maintain them.

6. **Easy to integrate**: The methods result in models that can be easily reused within common frameworks for end services that assist citizens and others (think of lawmakers and developers) including those at the edge (like smartphones).

7. **Affordability:** The solution is affordable in terms of time and cost necessary to start making use of it. We deem it affordable if it is possible to use it to model the fifty most used (Dutch) regulations within 2 years. It is out of the scope of this paper to assess this thoroughly by doing and measuring. What should be noted for now is that none detailed machine-consumable rules of a single regulation can be crafted gradually such that it will be usable soon enough even though it is not finished. At the same time, this is possible for explanatory natural texts that are compliant by design. Most time-consuming is a methodical analysis and interpretation of official laws and regulations (or getting to know what to create as official law and accompanying code directly) in all its details. It is possible to start doing this analysis and interpretation on a high granular level and gradually work towards more detailed granularity. We deem these high granular level compliant by design explanatory texts as already desirable and therefore usable. If a solution to be assessed supports this kind of gradual development, we deem it affordable. Note that there are regulations that are so big, that organizations back down on starting to create RaC models for them because they cannot be made usable enough in an affordable time.

8. **Maturity**: The methods have at least been proven in proof of concept projects, and are well-defined enough, documented, and in some way supported with tools and support.

Results

Evaluated solutions

We evaluate all the RaC methods, software, domain-specific languages, and platforms that have been mentioned at http://regels.overheid.nl/methoden and https://code.gouv.fr/fr/explicabilite/:

1. RuleSpeak®
2. Wetsanalyse (Law Analysis)
3. ALEF (RegelSpraak)
4. Calculemus/FLINT
5. OpenFisca
6. Catala
7. Publi.Codes

Co-funded by the European Union

8. Datalex.org (yscript)
9. Concordia Legal
10. Avola
11. Semantic HTML-vocabulaire
12. USoft
13. RuleManagement Method
14. Blueriq
15. Sparkwise

Note that we omitted De LegitiMaat and CircuLaw. They have a different focus than on RaC itself, which does not mean they are not interesting solutions that could help in modeling, but they are just not of interest in the context of this paper. De LegitiMaat is a framework to thoroughly assess existing laws. CircuLaw focuses on policymakers to give them a good overview of the law regarding a circular economy. Also a new latest entry in the list at regels.overheid.nl called DEMO[10] is not included in this paper, yet it harbors very interesting scientifically grounded and mature knowledge for the RaC domain too. It is focused on identifying so-called universal translation patterns from natural language descriptions of systems. By doing so it avoids unnecessary complexity often encountered when trying to model a system through business process modeling. Another interesting method that could add to this is the USM method[11] which identifies only five universal processes around services in general, simplifying over all kinds of other well-known enterprise architecture techniques in which the problem the RaC domain tries to solve is only a facet. All of this is out of scope of this paper however.

PRA RaC solutions assessment matrix

Aspect> Solution	Scope	Traceability	Explainability	Compliance	Traceable Identifiability	Transparency	Easy to integrate	Affordability	Maturity
1. RuleSpeak®	I		½			✓		✓	✓
2. Wetsanalyse	A, I, E	✓	½	✓		✓			✓
3. ALEF (RegelSpraak)	I, E, C	✓	½	✓	½	½	½	½	✓
4. Calculemus/FLINT	A, I, E	✓	½	✓	½	✓	½	½	½
5. OpenFisca	E, C	✓	½		½	✓	½		✓

[10] https://ee-institute.org/demo/, at the time of writing the final version of this paper has been added to the list on regels.overheid.nl
[11] https://usm-portal.com/what-is-usm/?lang=en

6. Catala	I, C					✓	½		½
7. Publi.Codes	I, E, U, C	½	✓			✓	½	✓	✓
8. Datalex.org (yscript)	I, E, U, C	✓	½	✓		✓	½	✓	✓
9. Concordia Legal	A, I, C	½	½	✓	½		½		✓
10. Avola	I, E, C		½				½		✓
11. Semantic HTML-vocabulary	E, U	½	½		½	✓	½		
12. USoft (using SBVR)	C	½					½		✓
13. RuleManagement Method (using RuleSpeak® and SBVR)	I, E, C	✓	½	½		½	½		✓
14. Blueriq	C, E	½	½				½		✓
15. Sparkwise	C	✓	½				½		✓
2b. Wetsanalyse + when with annotation tool and combined with solution focused on end-user explanations*	A, I, E, U	✓	✓	✓		✓		✓	✓
3b. Open Sourced ALEF (using Wetsanalyse) + when with annotation tool and combined with solution focused on end-user explanations and extra export functions*	A, I, E, U, C	✓	✓	✓	½	✓	✓	✓	✓
4b. Calculemus/FLINT + with improved annotation tool, choice of fact function language*, export functions and combined with solution focused on end-user explanations	A, I, E, U, C	✓	✓	✓	½	✓	✓	✓	✓
5b. OpenFisca + when made suitable to be combined with Wetsanalyse or Calculemus/FLINT with Python as fact function language and combined with solution focused on end-user explanations and extra export functions*	A, I, E, U, C	✓	✓	✓	½	✓	½	✓	✓

Table 1: PRA RaC solutions assessment matrix
**) see the Discussion section for an elaboration of this option*

In Table 1 you can find the assessment using the PRA RaC assessment framework of all solutions evaluated in a single matrix of required aspects on one axis and solutions on the other. The bottom four rows display the expected scores when solutions are combined and or altered. As mentioned before, the solutions are not of the same type and may be combined and or refined to get to a more suitable solution, as none of the existing solutions are minimally viable. More combinations might be possible. In the Discussion section, we elaborate on these combinations. Below, we summarized how we came to this assessment for each solution (except for the extra combinations, which are discussed in the next section).

RuleSpeak

Within this list, RuleSpeak is a relatively mature solution that has been developed since 1996 and has been translated into multiple languages. It is not a formal programming language, but more a set of strict guidelines on how to formulate rules in natural text such that there will be no or less ambiguity when evaluating the rules expressed in RuleSpeak whilst being readable for non programmers. It is freely available from https://www.rulespeak.com/en/ in the form of practitioner kits.

RuleSpeak does not specify how to analyze existing laws and seems to be focused on business process modeling. It might help interpretation when translating existing law into RuleSpeak, and also it seems possible to generate code from sets of rules that follow RuleSpeak guidelines. Furthermore, it does not include tools, and it will let you publish models using RuleSpeak in public. It could easily be paired and published together with explanatory content. In that sense, it is affordable. But without tooling that guides you into its application, it would be harder to integrate and put to use. This is where ALEF comes in.

Wetsanalyse

A proven conceptual model of how to identify a common structure in laws is described in the book Wetsanalyse by Lokin et al (2021)[12]. Wetsanalyse translates to Law Analysis. It has already been used to model laws for use within Dutch tax services. Besides many insights on how to analyze and interpret laws and common pitfalls, it also provides a conceptual model of relatable high-level concepts with which to identify a common structure within laws. This happens to be comparable yet slightly different and somewhat more evolved and mature from how Calculemus/FLINT currently models laws as a normative system (see next section) and both are partially based on the same research. It could be used to make rules which are based on this analysis and interpretation more explainable. Currently, Wetsanalyse is still only described as knowledge and seems to be focused on Dutch law (though this is more a matter of translation as it will be applicable abroad too).

Note that currently an informal project in higher education has been initiated for the development of an annotation tool based on Wetsanalyse which would allow documenting the analysis of laws using Wetsanalyse making this aspect more at the same level of Calculemus/FLINT. In fact it seems in theory it might be possible to export Wetsanalyse models to FLINT models and vice versa as research by ICTU and TNO indicates.

ALEF

The Dutch Tax services created ALEF and currently uses an adapted form of Wetsanalyse to model a law in a multidisciplinary team and directly use an own RuleSpeak-like language (RegelSpraak which is Dutch for Rule Speak) to express machine consumable rules that include Boolean logic and arithmetic and so on.

For ALEF, an editor based on JetBrains MPS has been created[13]. The ALEF editor allows for unit testing (includes a rule interpreter), can generate developer documentation, and generate services within a microservice backend evaluate given input according to corresponding rules for usage in their backend operations. While the editor allows for documenting source references and comments to explain the model, this seems not to be the main focus. Also, the focus is more on microservices that can calculate and decide on things by executing the logic and arithmetic within small sets of rules, but there is no focus on other types of content, for instance, to explain things to users in a compliant way. It does not persist in a machine consumable form in which the whole model of the normative system, that is identified through Wetsanalyse, is to be taken as input (manually) when modeling in ALEF. This model seems to get lost within informal working documents or separate knowledge bases. Moreover, even though their statement and intention to do so and information on ALEF and the ALEF editor can be found, nothing has been really open sourced extensively yet, except for a concept definition of RegelSpraak as

[12] Ausems, A., Bulles, J., & Lokin, M. (2021). Wetsanalyse: een aanpak voor effectieve en uitlegbare toepassing van wetgeving. ISBN 9789462909373

[13] https://resources.jetbrains.com/storage/products/mps/docs/MPS_DTO_Case_Study.pdf?_ga=2.55354082.8921133.1699441411-1482573785.1699441411

a BNF. In this way RegelSpraak can be seen as separate and could be re-used outside of the context of ALEF being tooling for utilizing RegelSpraak.

Calculemus/Flint

Calculemus/Flint is a combination of CALCULEMUS being a method to identify a FLINT model. The definition of the FLINT model is a solution that can model any description of rules in any natural language based on a conceptual model that consists of three concepts. The first two are “acts” and “duties,” corresponding to two truly fundamental normative relations: the power-liability and claim-duty relations (Kocourek,1930), and the third concept are the surrounding facts denoted in “facts”. Acts are things actors have the power to do, given a precondition, resulting in liability for the same or other actors and thereby possibly resulting in state change within a runtime environment evaluating the model and enabling others to meet a precondition on other acts. A law could state, for instance, that a person has the power to apply for some benefit. Within the context of that law, this sentence would then become the source citation for an act with its own ID and parameters within the FLINT model of that law. Acts are not to be codified within the FLINT model but will refer to Facts that can define a so-called fact function, and these could specify machine-consumable code to execute and evaluate the fact in some runtime context.

It seems where most RaC solutions focus on evaluations of arithmetic calculations and Boolean logic, FLINT focuses primarily on identifying the power-liability and claim-duty relations that, when evaluated within context for a specific purpose, could be seen as forming a relatively simple process description within the law but also as a structure to methodically generate explanatory texts. Moreover, there seems to be a resemblance with the conceptual model from Wetsanalyse as well as some resemblance with SBVR which is often cited as being used in the more proprietary solutions in the list in this paper. Though the latter has not been developed with normative systems and relations in mind. From the start, Calculemus/FLINT is focused on international usage.

Even without codifying any arithmetic and logic, a FLINT model captures the structure of a normative system consisting of a linked non-circular graph of acts, facts, and duties all of which reference to one or sometimes more than one explicit citation from official law (or policy documents). The method to analyze the law as such, called Calculemus, also focuses on identifying discrepancies in law and how to fix them, but for this paper, we focus on modeling.

The way to analyze a selected law and model it into a FLINT model goes something like this: first search for all sentences within a law that clearly states an act, so something some actor has the power to do in some context, then identify most of its non-optional attributes as facts for which there must be some matching text within the law to refer to for each fact. The same can be done with duties. Then it becomes obvious, especially when visualizing these acts and duties, how these acts and duties relate, which fills in even more attributes of the acts and duties. Some acts will become so-called starting acts, which are the acts that do not seem to depend on other acts in their precondition and therefore are the acts where implicit processes compliant with the modeled law could start. FLINT models are stored as JSON files and besides source references it also includes an explanation of why these source citations do form a basis to identify an act, fact, or duty as this is sometimes more implicitly stated within laws. FLINT allows for extending the knowledge with your own information around acts, facts, and duties. With this model, you can codify facts where this makes sense and/or develop compliant explanatory texts that cover the entire or selected parts of the model.

Like with ALEF, at first, an editor based on JetBrains MPS had been developed for FLINT. Additionally, a Visual Studio plugin was created at some point. There was a need for a more lightweight and easier to use editor, however. So, it has been decided to deprecate these and start the development of an online editor for FLINT with the end goal of working towards a more user-friendly online environment in which multiple stakeholders can collaborate on models. Note however this is all done within research setting with limited funding.

Co-funded by the European Union

Calculemus/FLINT is being developed by TNO[14] which is an organization primarily focused on research. Currently, the automated analysis of laws into FLINT models is being researched[15].

More information:

- A third party made a short explainer video about Calculemus/Flint in English: https://www.youtube.com/watch?v=cpcffhinq38
- https://normativesystems.gitlab.io/knowledge-modeling/documentation-website/docs
- Original paper first describing Calculemus/Flint: https://open-regels.nl/en/assets/docs/calculemus.pdf
- In 2024 Robert van Doesburg is expected to publish a book in English on norm engineering using Calculemus/FLINT describing all of it in full detail.

OpenFisca

OpenFisca is worldwide one of the most well known and most cited RaC solution originating in France. It provides a Python library that allows developers to model a tax-benefit system in terms of coded rules which can refer to documents they are based on. These models then can be published, used through an API, and explored through a web application. Moreover, and important, it is free open-source software.

OpenFisca does not explain or describe how to do analysis or interpretation of existing laws and regulations in the form of natural language however[16]. Also, it results in models in the form of Python code which while an API is provided to interact with it and Python is an excellent choice for data analytics and AI projects, it will be less easy to integrate this client side (on the edge) for instance.

Given the solutions described until this point in this paper, it seems OpenFisca models might be generated out of models from solutions that focus on analysis and interpretation or OpenFisca might be enriched by supporting these things on its own.

For more information about OpenFisca: https://openfisca.org/en/

Catala

Catala is a domain specific language especially designed for the purpose of translating statutory law in the form of natural language into formal code (in the form of Catala code). It is well defined and embedded in science[17]. Catala is a research project from India, the French National Research Institute for Computer Science. It is open source and Catala can be transpired into several other programming languages allowing for easy integration. It does not focus on explainability or traceability (yet) however. Also, it is mentioned on their GitHub the compiler is unstable and lacks some of its features[18]. It is a promising solution, however.

More can be found at https://catala-lang.org/en/examples/tutorial and https://code.gouv.fr/fr/explicabilite/catala/ (in French)

Publi.Codes

Publi.Codes actually focuses on explainability of laws and the interpretation thereof for all users including citizens. It enables the definition of rules in an easy-to-use domain specific language embedded within

[14] https://www.tno.nl/en/

[15] https://aclanthology.org/2022.lrec-1.47/

[16] as can be concluded from https://openfisca.org/doc/index.html

[17] Merigoux, Denis & Chataing, Nicolas & Protzenko, Jonathan. (2021). Catala: a programming language for the law. Proceedings of the ACM on Programming Languages. 5. 1-29. 10.1145/3473582.

[18] "The compiler is yet unstable and lacks some of its features." https://github.com/CatalaLang/catala

Co-funded by the European Union

explanatory content and publishing and using these in several ways using online tools. Also, it is open source. The original article with which it started is: https://kont.me/vers-impl%C3%A9mentation-officielle-de-la-loi (French). Started by ex-members of https://mon-entreprise.urssaf.fr/ it is now developed by a community focused on the French government. Find more about Publi.Codes at https://publi.codes/

Datalex.org

Datalex.org is a highly evolved mature academic solution used in Australia which involves tooling around creating RaC models using Yscript. It even allows for defining user centered explanations, interactive testing functionality for end users and report generation to a certain degree. Datalex.org is a work of joint authorship, involving authors employed by UNSW, and authors not affiliated with UNSW" as stated on their website http://datalex.org.

Concordia Legal

Concordia Legal is a Dutch company with expertise in developing solutions for the rapid development of mass-decision making processes. They developed their own method to analyze and interpret laws into several models, starting with a formal rules model and also including a data and process model. Using specific tooling to speed up the development all the way to working API's and forms. They are open in how they do these things, and this is quite interesting. Yet it requires considerable expertise, which is what they offer. More information (Dutch only) can be found on regels.overheid.nl: https://regels.overheid.nl/docs/methods/CONCORDIALEGAL

Avola

Avola focuses on decision services through implementing rules as decision trees using the DMN standard[19] and provides, a proprietary suite of tools and applications to make use of these in business context. When it is able to import from other RaC methods this might be a COTS solution for further integration in specific contexts. More can be found at: https://avola-decision.com/

Semantic HTML vocabulary

In a collaboration by the Dutch Ministry of Finance, TNO and Triply, a new W3C standard called Semantic HTML vocabulary is promoted which focuses on enabling, publishing natural language texts, like laws or any content in an already annotated form from the start. Such that it is already analyzed and can be interpreted and traced back as intended. It is targeted to integrate the FLINT standard, which also uses triply like RDF. It is still in its early stage. More information on this can be found at W3C: https://www.w3.org/community/htmlvoc/

USoft

USoft provides proprietary solutions that utilize a rule engine for rules defined using SBVR with low-code tooling around this. Semantics Of Business Vocabulary and Business Rules (SBVR)[20] is business-oriented and a mature standard by the well-known Object Management Group to capture business rules in a non-technical way. When it is able to import from other RaC methods this might be a COTS solution for further integration in specific contexts. More information can be found at: https://www.usoft.com/

RuleManagement method

The RuleManagement method has been developed by the Dutch companies IAM4 and Rule Management Group which were founded as a result of a major research program by the Dutch Tax Authorities called "POWER". Which uses SBVR and RuleSpeak® and can result in free to use open source models with accompanying documentation. However, it requires a proprietary software tools: the IAM4 RMSuite and significant expertise to utilize effectively. It can export rules into several programming languages and other rule engine environments like Blueriq. More information (in Dutch) can be found here: https://www.rulemanagement.com/

[19] https://www.omg.org/dmn/
[20] https://www.omg.org/spec/SBVR

Co-funded by the European Union

Blueriq

Blueriq is a well known mature and widely used Dutch proprietary solution formed around a business-rules engine, and a comprehensive suit of tools and services to integrate this into back offices or serving as a backbone for online services. It also focuses on traceability and explainability for non-technical experts developing with Blueriq, yet seems to require quite some specific expertise to make use of. When it is able to import from other RaC methods this might be a COTS solution for further integration in specific contexts. More can be found at: https://www.blueriq.com/platform

Sparkwise

Sparkwise is a business oriented proprietary solution focusing on developing, integration and monitoring of (automated) decision services based on rules defined as decision trees, providing easy to use tooling for non-programmers. When it is able to import from other RaC methods this might be a COTS solution for further integration in specific contexts. More can be found at: https://www.sparkwise.io/

Discussion

As explained in the definition of the Affordability aspect, a solution (or combination of solutions) can only be minimally viable when it can be made to use while laws are gradually modeled into models. In the Netherlands, given the size of certain laws, this can only be achieved when initially usable, traceable, and compliant explanatory texts are developed at a higher level of granularity, to be refined to more detail and eventually include machine-consumable rules gradually.

At the beginning of 2023, an experiment to assess the viability of this using Calculemus/FLINT was performed. It showed that it seems possible to use methods for analysis like Wetsanalyse and Calculemus/FLINT not only with a focus on creating machine-consumable rules, but also explanations in many varying forms (such as a very high-level summary or a more detailed explanation of a certain part, both in varying languages at varying levels of language difficulty and from varying perspectives of end users and so on).

Furthermore, these models are to be used in different contexts. For the PRA, they need to be able to be used within the edge device of the end user (smartphones of citizens). This enables optimal privacy and is a lot more efficient and cost-effective. At the same time, they need to be able to be used in operations in back offices and for statistical analysis. It would be convenient if there were a standard from which the models (not only the rules, but also the explanatory texts and so on) could be exported or converted so that they can be integrated into all these vastly different environments.

Calculemus/FLINT currently allows for a choice to use a different language to define more complex fact functions instead of the limited one that it originally defined, which is by design. It could specify fact functions in the form of Catala, OpenFisca (Python), Yscript, and RegelSpraak to define calculations, for instance. The other way around seems to be possible as well, such as exporting an ALEF model to an FLINT model or vice versa, though there might be caveats. Additionally, loading an ALEF model or FLINT model into Sparkwise or Blueriq, or converting them to OpenFisca, seems feasible.

It seems that when a government would publish laws in annotated form such that they can be consumed as a normative system model along with specific interpretations in the form of DSLs (Domain-Specific Languages) to formally and officially specify complex rules, along with matching explanatory content and metadata about traceability, it is possible to use a wide range of other solutions to make use of these kinds of models.

Co-funded by
the European Union

This way, by combining solutions and adding an end user-centered focus on compliant explanatory information to it there are several options that might deliver on all aspects sufficiently. Examples of these are given at the bottom of Table 1.

Future work

A small piece of Dutch law has been selected to assess methods by modeling only this same small piece. This has only been done with a few methods (ALEF, FLINT, Concordia Legal, and OpenFisca) however. Instead of this, a set of more varying types of example pieces might be processed with all methods to assess certain aspects more extensively and objectively.

Moreover, an extensive rationale for why these are the aspects in the framework, not more, not less, is missing. We expect it can be grounded in existing or new laws and/or architecture. More detailed required aspects can be identified as well. Additionally, there is a relationship between AI and RaC and explainability that could be researched more extensively.

Finally, as discussed, it might help to investigate combining solutions and promote the development of interoperability between them, thus improving on the seventh aspect of easy integration and establishing an interoperability standard for publishing usable models to effectively assist citizens with regulations.

Conclusion

To effectively assist citizens with regulations in a safe, trustworthy personal manner, RaC models need to contain more than just machine-consumable rules. Besides crafting these rules in a methodical way within a multidisciplinary team using law analysis methods like Wetsanalyse or Calculemus/FLINT to minimize misinterpretation, these rules need to be explained in a compliant, highly qualitative manner in various forms for end users, which even now is not currently automatable. Models could and should include these explanatory texts to be (re)used consistently in the interaction with users in systems where the rules are evaluated. While RaC methods often focus on explainability for stakeholders involved in the modeling process, the various forms of citizens as end users are not taken into account, and when they are included, it is very unlikely there will be one version of explanatory text that will be usable for all stakeholders (some may find a text condescendingly oversimplified while others may need it oversimplified). Additionally, RaC methods often seem to overlook traceability (in a machine-consumable way such that the traceability is explainable for all stakeholders).

None of the "methods" mentioned on regels.overheid.nl deliver on all the aspects assessed in the initial PRA RaC Assessment Framework presented in this paper, and none of them are minimally viable for use in a PRA on their own. While they do deliver on certain aspects, they are more complementary than being full alternatives to each other, indicating they might be combined into a new single method, as already seen in the case of ALEF. It is not advisable to enforce a single standard in the form of referring to one or one combination of specific selected solutions. When defining a standard for compliance by design RaC models, for instance, in relation to tooling like envisioned in the PRA project, it should be in terms of delivering on required aspects to be covered, as outlined in the PRA RaC Assessment Framework. An important aspect to consider is interoperability.

RaC solutions, or a combination of them that cover all aspects, must include a methodical way to establish compliant-by-design explanatory texts with a certain quality along with machine-executable rules, enabling gradual development, which is essential for them to be viable.

Co-funded by the European Union