BETTER RULES –
BETTER OUTCOMES
SHAPING THE FUTURE OF GOVERNMENT REGULATION

Archived Loomio Forum Discussions

September 2018 – June 2020

ARCHIVED FORUM DISCUSSIONS

In September 2018, the Better for Business (B4B) team within the Ministry of Business, Innovation and Employment (MBIE) facilitated a digital forum discussion on the “Better Rules, Better Outcomes” methodology and its applications. Also referred to as “legislation as code” or “Rules as Code”. The forum brought together a community of government officials, academics and business professionals who shared their opinions, ideas and supplementary resources. MBIE recognises the value of preserving these discussions in an archived format for other interested parties.

Disclaimer:
This document presents themed discussion threads from the digital forum. The views, ideas and opinions expressed as part of the open discussion are not necessarily the views of MBIE.

For reasons of privacy, MBIE has removed the names of contributors as well as any supplementary resources to ensure impartiality toward business entities or individual interests.

MBIE does not hold the copyright to any of the concepts put forward and has taken every measure to remove supplementary links and pages, where contextual information may indicate a preference or anti-competitive practices.

TABLE OF CONTENTS

ABOUT CONCEPT MODELS / BUSINESS RULES / SBVR

2019-05-22

There has been some discussion and mention of concept models, business rules, and SBVR in various threads and during the Rules as Code Show & Tell. (Thx for that. Excellent!) Here is some background you might find useful, along with some (readable) references.

Concept Models: A concept model is about business meanings, which are expressed by business vocabulary (both nouns and verbs) and related definitions and definitional rules. Think of a concept model as a structured business vocabulary aimed at the disambiguation of natural-language textual statements (including business rules). A robust concept model would support Google-like search about the core knowledge of a domain.
More: http://www.brcommunity.com/articles.php?id=b779&zoom_highlight=concept+model

Business Rules: I suggest a quick read of the Business Rules Manifesto, which can be found at: http://www.brcommunity.com/articles.php?id=s030. Just 4 pp. Translated since 2003 into 18 languages. Business rules are criteria that shape behavior or guide decisions.

SBVR (Semantics of Business Vocabulary and Business Rules): SBVR is the standards work that arose to enable robust, logic-based expression of concept models, business rules and other forms of business communications. In SBVR, rules can either be necessities (alethic) or obligations (deontic). The latter are essentially rules that can be broken (violated), with enforcement level and variable violation responses. For more: http://www.brcommunity.com/standards.php?id=620.

Happy to answer questions. Please be patient (going on vacation)!

2019-05-22

Thanks - useful stuff here. I am about to go on holiday too, but will take some time to digest. I know others on this forum are already familiar with SBVR and have mentioned it here & in Twitter discussions, but I would like to get a better handle, from the perspective of a legislative drafter rather than an IT expert, on how it relates to the Rules as Code idea.

On aspect I am still trying to understand what is happening in RegTech (& "SupTech"), where it seems to me there may be scope for cross-fertilisation between work on business rules and work on legislation (with regulator guidance somewhere in between). Are people in RegTech using SBVR?

Meanwhile I think I will find the examples at http://www.brcommunity.com/articles.php?id=b286 a very useful way in to a better understanding of the idea of a structured natural language. It has interesting parallels in the world of legislative drafting.
I have suggested that other legislative drafters might find formal logic useful in helping them do their common sense logic checking of their drafts, but also as a way to understand how coders are seeing legislation - https://www.slideshare.net/MatthewWaddington3/formal-logic-for-legislative-drafters-waddington-2019 . That involves analysing the elements in a draft, which is made easier by the trend of modern Commonwealth drafting towards using a leaner & more standard expression for significant elements (such as "may" instead of "shall

be entitled to").
Back in 1845 George Coode, in the first publication purely devoted to legislative drafting https://archive.org/details/onlegislativeex00coodgoog/page/n7 , started analysing elements of a "legislative expression" into "cases", "conditions", "legal subjects", "modal copulae" & "legal actions" (developed by Driedger in 1956, and taken further since).
More recently the move away from "shall" has led to readier distinctions between "must" (where the drafter should be alert to the need to ensure there are consequences for contravention) & "is" (where the legal effect happens by operation of law, with no contravention possible & no need to provide for it).
I am interested in getting drafters to see how that relates to propositional/predicate logic and deontic logic - that of course has parallels in the alethic & deontic concepts you mention in SBVR terms, and then to the LegalRuleML concepts of constitutive & prescriptive norms - http://docs.oasis-open.org/legalruleml/legalruleml-core-spec/v1.0/csprd02/legalruleml-core-spec-v1.0-csprd02.html#_Toc493074620 .
I am still not sure how these schemes accommodate legislative provisions like "A body corporate, the XYZ Regulator, is established" - where no contravenable obligation is being imposed, but more is being done than simply defining terms or stating relationships - a new legal entity is being created by operation of law. Plenty to chew over.

2019-05-23

Quick note: Have a look at www.RuleSpeak.com.

2019-05-23

Interesting. I'm in the midst of drafting something to be shared shortly. It's an operational structure for a pair of questions.
First considering the narrow case: Is this rule in effect at a given date/time and in a given jurisdiction? Is this rule applicable to a give set of circumstances?
Or in the more general context: What rules are in effect at this particular date/time and in this nested hierarchy of jurisdictions? What rules are applicable to this particular set of circumstances?
Basically, a colleague and I are documenting our approach to creating a semi-automated system for this. I'd be interested if others have framework-level references that would be useful to this end.

2019-05-23

In effect you're talking about rule management. We created a tool for that consistent with SBVR (we no longer own it) called RuleXpress by RuleArts. It's been used to manage concept models and many 1,000s of rules. Business-side management of rules, coordinated with vocabulary, is an important area.

2019-05-23

Yes sort of, but not rule "management". More like managerless P2P rule coordination, distributed management perhaps. Thus, we are working on an (btw consciously not "the") Internet of Rules in a world of self-sovereign rule authors, some of which hold jurisdictional authority in law, some of which hold reputational authority, and so on. The boundaries are not strict. For example conformance with a rule in a voluntary standard or user manual can be considered in legal settings as operating with due diligence or in accordance with a standard of reasonableness, and therefore is functionally equivalent to a statutory regulation. As components and specs for an Internet of Rules, 100% of our work is free/libre/open (Apache 2.0 for all the core elements; AGPL for specialty functions and utilities).

2019-05-23

This is useful: "Business Rules Manifesto" http://www.brcommunity.com/articles.php?id=s030 It is probably equally relevant to computational legislation/regulation.

2019-05-23

Hello, I've been looking at SBVR for a while, am a convert, and am arguing for it to be a foundational specification in creation of a new business rules management/governance capability. However tools to assist achieving conformance, and more importantly retaining conformance over the long term, will be critical. We've had a trial with RuleXpress and while I've got some reservations about the UI (the graphical side of concept modelling in particular needs improvement), and there were a couple of bugs, the design and functionality were good and the publication/output options were promising. We will probably also look at Urequire Studio from Usoft - are there any other SBVR-supporting software options you could suggest?

2019-05-23

See SBVR in http://wiki.ruleml.org/index.php/RuleML_Home#RuleML_as_a_Bridge

2019-05-23

Where I'm focussed, and where SBVR is aimed, is the business activities of capturing and managing business rules expressed in natural language - I'd note the introduction to the SBVR spec:
"This specification is conceptualized optimally *for* business people rather than automated processing. It is designed to be used for business purposes, independent of information systems designs to serve these business purposes"

2019-05-24

I just feel its a perfect match for the world where the "Business" is Government and Regulations that we write and need to get other people to understand both the Context of the words and the Intent of the Regulations. Having a consistent "standard" way of communicating this and being Visual will enable a better connection between the humans which will take us a long way to Matching Intent with outcome.

2019-05-28

We used concept models in a discovery work. It has definitely proven to be the tool to use in any rules area (legislation, policy, contracts, etc)

- Enabled knowledge sharing - team of people with different expertise levels and knowledge brainstormed the concepts

- Ensured common understanding of the area (to both season experts and newbies)
- Quickly defined scope of the area as some concepts were deemed out of scope immediately when brought to the table/board
- Focused the discussion, as everyone could see when the discussion was veering off course

Main challenge:

1. How to digitally store these so that they can easily be re-used, shared and modified collaboratively

2019-05-29

Up here close to the North Pole we're trying to enable this cross-governmentally using the following method & platform
> https://yhteentoimiva.suomi.fi/en/

The method is based on defining concepts in a terminology through concept analysis (http://lipas.uwasa.fi/~atn/papers/artikkelit/OnTerminologySc.html) and then applying these concepts in ... models that are perhaps closer to conceptual data models than "concept models" (inclusion of concept (class) specific attributes in the model). Our upcoming Legal Editor (https://drive.google.com/open?id=1ubW3BgMdUg2UqueOIbv4JFhCuNMF1gRmkngIsEO6tkA) will then be able to make use of both the concepts in the terminologies as well as the (conceptual) data models based on the terminologies utilizing Linked Data principles (persistent URI-references). Not yet a full blown "machine consumable legislation" solution but a good start :-)

2019-05-29

I had the privilege of having teach me his method of developing a concept model. In this case we used the example of a clause from my template IT contract, rather than a piece of legislation - but the idea is the same.
<name> and I have agreed to blog about the process.
The concept model doesn't write your rules, it is about breaking apart the concepts contained in your work (either legislation sections, or clause of a contract, or piece of policy) and mapping the relationships between those concepts.

Going through this process yesterday, I found it helped me to find the pieces of that work where assumptions have been made, and where definitions are incomplete. Where a team are working together on the model (which is ideal) this is where any assumptions about language are unpicked - and discussions had about how the team members have assumed the work should be interpreted.

We are working on sharing this skill with as many people as possible. It is fabulous when you are working on change.

(Thanks for your lesson yesterday!)

BENEFITS OF BETTER RULES

2018-09-20

If all the government rules were openly available in both human and machine consumable formats what would be the benefits and impacts?

2018-09-21

From my experience in building some legislation as code projects the overriding benefit of machine consumable rules is: (and yes it seems counter intuitive) *much* better comprehension of legislation for citizens.
This is because the "human" version is hard to comprehend - because of the definitions and referencing between/within instruments which results in most people needing to pay a lawyer to comprehend/trace the logic for them.

With machine readable legislation - programmers can replace lawyers for that initial comprehension component by creating "comprehension programs" that can adapt to a persons input and illustrate the law in a widely disseminated and adaptable format.
In these scenarios the person using such a program can now trust their "reading" of the logic - and focus on their inputs to their scenario and the outputs/outcomes.

2018-09-26

I think that there's a possibility that extending the visualisation of legal logic past the toe-dip we already did with openfisca could have massive potential in making the complex easy to comprehend Hamish

2018-09-21

I wonder whether it then becomes possible to run test scenarios through to check for clashes with other bits of law.

For example, there's a fairly new reduced-harm tobacco device that heats the tobacco rather than burning it. It is required, under the SmokeFree Environments Act, to comply with packaging rules for combusted tobacco including all of the warnings about the harms from smoked tobacco. But the Fair Trading Act prohibits false representation and misleading statements.

There have to be other cases like this that come up that, under a code-based environment, could throw the same kind of error flag in test scenarios that I get when I screw something up when writing Stata code and ask the machine to do impossible things.

2018-09-24

This doesn't look like a logical error, or a legal one, and maybe not even a policy one. Legally it just means the new product can't be sold, because it doesn't meet the two requirements. It is not creating an impossible inconsistency of requiring someone both to do X and not do X. You can't assume policy-makers would want it to be legal or that there is a god-given right to sell it. If the product didn't exist, and wasn't thought possible, when the rules were developed, then no amount of error testing would have changed minds on it then. Once the product is developed, you can lobby for the rules to be changed to allow it to be sold, but there is no automatically right answer about whether it would be a good idea to do so (it might be a cynical gateway to the smoked products).

2018-09-24

You're right that this clash wouldn't have been caught before the product existed. The clash is revealed consequent to the Court's telling the Ministry of Health that the Ministry was incorrect in its interpretation of oral tobacco prohibitions in SFEA (and that a ban was inconsistent with the purposes of the Act).

Was an example of a fun clash that came quickly to mind because I do a bit in this area.

2018-09-21

France's experience when coding up existing legislation was that they found endless logical loops within enacted legislation. They had to modify the software to exit the loop after an arbitrary three times and note that's what they had done.
Legislation is like the type of software you might write on paper - every programmer knows it's almost impossible to spot all logical flaws without actually running it.
Running solves the pure logical flaws, whereas your example would be harder to catch due to the assumption on input that the Smokefree Environments Act must be making.
Which progresses to the next aspect of legislation as code - the testing library. All coded laws should have test suites written along side them, this gets added to over time with real world exceptional situations. Over time this builds up a hugely valuable repository of scenarios which legislation drafters can utilise when modifying or testing new laws - based on real situations/real people previously encountered. We do that as the first step with the work we've been doing, we write the tests first and then code the law simultaneously to the tests and the legislation. Whenever an issue comes up - we write a new test for it.

2018-09-24

French (& other civil law) legislation works very differently from NZ/UK. (& other common law) legislation, usually in being more general and not assuming the citizen can do anything not prohibited. I would need to see examples of these loops before assuming they reflected problems in the legislation instead of the coding. There are badly drafted laws, but there is badly written software too surely, and cases where the person trying to do the coding has just bitten off more than they can chew.

2018-09-25

Example. This implementation has been reviewed by several people and has given appropriate results for 3 years, as proven by a number of tests.

I have been surprised by how much the differences in the process of *making* law in civil vs common law frameworks seem to disappear when coding the resulting law into software.
For example, OpenFisca was developed initially with only France in mind and has successfully grown to accommodate 6 different countries. Adjustments

are necessary, but we haven't encountered incompatibilities or major inconsistencies yet. Most issues seem to be in *values* rather than *dimensions*.
For example, the time periods that are applicable in each system differ, but the notion of a *time period* itself is common.

One thing I have noticed over years of translating law into software though, is that moral judgement on (either legalese or software) legislation implementation doesn't get you very far. When you consider that the end result is “badly written”, you most of the time ignore most of the constraints that led to such an implementation.
In the above example where a benefit computation had to be stopped arbitrarily after looking 3 months in the past, because the value of the benefit itself is taken into account to know how much one is entitled to, thus leading to recursive calls, you might say the law is badly written (“how stupid is it that we ask for some value in the past when we surely know about it already?”). When you learn more about the concrete activation of the right, you realise that multiple agencies may give away these benefits, and that when a beneficiary moves from one region to another, they might get different values based on locality. The case where one would apply and have their entitlement computed *by the same entity* was not explicitly taken into account by law, indeed, because it made no sense. The fact that the implementation becomes clumsy and recursive is entirely a side effect of legislation as code**.** Who is to be blamed for that? No one but ourselves.

Another case I encountered is when one could get into paradoxical situations due to threshold effects: being entitled to benefit A would increase your income so that you would not be entitled to benefit B which would by law prevent your eligibility to benefit A, which would decrease your income so that you would be entitled to benefit B, which would…
Once again, who is to blame for that? Is it stupid legislation? Or do such cases never exist unless you've built a model that can compute everything in a microsecond? In real life, time of entitlement activation by an agency would be 1 to 2 months, and the feedback loop to another agency could be 6 to 12 months. So you need to add weird edge case management in the code, making assumptions regarding agency efficiency. Is it poor coding? Or is it just part of legislation as code, just like issuing decrees akin to “fixups” are just part of legislation as legalese?

2018-09-21

IDIOM Ltd has spent the last 17 years codifying other people’s rules, including legislation, that are defined in legalese or some variation of logical English. I can assure you that no matter how experienced or careful the authors, matter how many peer reviews, no matter what checks and balances, using natural language to define rules always fails to define rules that have only one possible interpretation.
That is why we have Courts.
The problem is the idiom – what do the words mean in this exact context?
And all of that is before you factor in discretionary input, which is endemic in legislation.
You can refute the above – no problem. But if its correct, then the issue is that the algorithm must actually be the legislation, not a representation of it. The algorithm in this context is the first order predicate logic, the algebra, and the rules (constraints) that convert the real-

world data into the useful outcomes that the legislators require.
If true, it follows that natural language description of the legislation must be derived from the algorithm, not the other way around.
How long before legislators start passing laws that are defined by algorithms and proven with the myriad of test cases that any normal system would require?

2018-09-22

I disagree, though maybe only slightly. I don't think it's necessary that either the natural langue or encoded versions of the law be primary, and the other secondary. Both can be primary, like in Canada where we have legislation that is written in both English and French, and both languages are authoritative. Or, you can develop the encoded version at the time of drafting in order to inform the drafting about the breakdowns in the natural language version in terms of completeness, which would allow the legislatures to include only the indeterminacy that they intend to, because it has a valid legislative purpose. But then we have to approach what it means, in terms of official advice, if the legislative body also provides an encoded version, which of necessity is more specific than the law itself. It has implications for what happens to people who rely on those extra parts to their detriment.

2018-09-22

I think there is also a difference between hard rules and interpretive ones. Rules that are prescriptive, assuming you capture definitions (calculations, eligibility conditions, etc) lend themselves nicely to this domain. But judgement based rules probably should not be fully automated to maintain human accountability. There is a balance to be found.

2018-09-22

There's a difference, yes, but that is not the boundary between what should and should not be automated. I'm currently working on a project to implement a technology for automating the predictions of outcomes of judgement based rules. The theory I'm operating under is that if a person who would give advice as to meaning of those rules, such as a lawyer, can develop a system that gives advice of the same or better quality than they themselves would give, recognizing that it will never be perfect, then that, too, should be automated. We should not set a standard of perfection on automation when all the humans are able to achieve in the same realm is decent averages.

The problem from the perspective of the rule-drafter is that they are intentionally leaving that space uncertain, because doing that has utility for them. Maybe the details are too complex, or change too frequently to be properly laid out, or there is a policy utility to people not being able to predict the outcomes so easily.

So I disagree about whether it can and should be automated. But i agree that automating it at the drafting phase is not viable. But facilitating the automation of those sorts of decisions later IS something that could happen at the drafting phase, such as where legislation provides a list of factors that will be considered in determining whether or not an interpretive rule requirement is met. Having a digital version of that list is a good starting point for building the analogical reasoning tools to automate the predictions later.

2018-09-24

All I've seen yet of “automating the prediction of outcomes of judgment-based rules” ends up being wide scale machine-learning based on a corpus of jurisprudence (precedents). Maybe your case is different, in which case I would love to learn more about it :)
For these cases though, I believe they are distinct from “legislation as code”:

1. On a lexical basis: we're not *coding legislation*, we're teaching a computer to recognise (known or unknown) inputs and give a probability of outcomes within a predefined domain (months in jail, dollars in compensation…).
2. On an attribute basis: such programs can not offer accountability, nor reflexivity, which I believe are properties one expects from software-encoded legislation (e.g. spotting logical flaws in natural language legislation, defining which condition led to some specific result).
3. On a pragmatic basis: such predictions are extremely sensitive to training set bias, and I believe we should be extremely wary of the consequences of automating and generalising such bias (see for example https://www.propublica.org/article/machine-bias-risk-assessments-in-criminal-sentencing).

2018-09-24

I can see no good coming from prediction machines for legislation outcomes. I will outline my reasoning briefly. Machines are brutal in their unawareness and people are inherently drawn to the path of least resistance. If any such prediction machine became prominent then people generally would defer to it and use it's results in argument resulting in circular reasoning.
Now in order to be true/usable in the real world - take NZ for example - you will need the inputs to ask for ethnicity and the algorithm would need to be biased against Maori to give the most accurate result for it's prediction (If I was Maori and using such a machine I would want to know what my outcome was most likely to be - not be set up to fail because the machine assumed I was Pakeha). This only bakes into our wider systems the flaws and bias's that already exist.
Legislation as code doesn't have to fight this issue because it's only concerned about an exact replication of logic - if the law is biased then legislation as code will replicate that but also better enable people to highlight it and draw attention to it to help people comprehend it more easily.
For instance we're looking at coding the NZ super laws, once that's done (and yes this is fairly trivial) we can cross analyse the life expectancy situation in NZ and illustrate the unfairness this creates for groups with lower life expectancy.

2018-09-25

Yeah, it's not ML. It's a declaration of parameters for analogical reasoning, and it also requires an annotated database of previous cases among other things that a human expert must touch. And it is very new, and I daresay its analogical reasoning is probably weak. Nevertheless, it has shown impressive potential when applied to certain common-law questions (e.g. "is this a trade secret"), and it provides reasons, and it does not guess when it is not confident.

You are absolutely correct; it is not law as code. It is encoded analogical reasoning about open-textured pieces of law, which is different. My disagreement was just over whether conclusions drawn on those parts of the legislation are capable of being automated.

But it is capable of explaining the analogy that it has used to come to the conclusion. And it is subject to bias in the database. But then again, so is a lawyer, if they are actually doing reasoning by analogy. What can't happen is that the system would be making decision on factors that the lawyer did not explicitly say should be included, so that reduces the risk over machine-learning approaches, somewhat.

2018-09-25

Let me give you an example of a benefit. In my jurisdiction, if you are married, you know what the rules are when you get divorced. If you are not married, as a lot of homosexual couples aren't, despite it being legal now, the question of what happens when you split up depends in part on whether you are in a common-law relationship. In my jurisdiction, the common law has been codified in a statute. That statute is almost entirely closed-textured, except for the definition of "living in a relationship of interdependence."

If I can generate a machine that can predict whether or not two people are "living in a relationship of interdependence" at least as well as I can, as a lawyer, then I can give homosexual people the option of getting legal information about their situations in as cost-free a manner as married couples can find it.

I think that's a benefit. And again, I acknowledge that this is not on-topic. Because what we are talking about is encoding laws, and laws specifically leave out the algorithm for deciding open-textured issues. So what I'm talking about is actually encoding lawyers. But the point remains that open-textured elements of a law are still capable of automation, and shouldn't be viewed as the point at which automation should stop.

2018-09-25

Thanks for these details! Super curious about your approach and implementation, if you have any elements to share they are very welcome :)

2018-09-25

Yes that's good detail - appreciate it - yes we're off topic but I very much appreciate the wider framing as such a machine could well make use of encoded laws and I find it important as a human being to understanding structurally what direction we're taking.

2018-10-19

I now have something that I can actually show people to help them understand what I'm talking about. Check out the blog post at <link removed for privacy reasons>   for the details. There is a link to a live demo. If you follow the link to the github, you can take a look at the actual code, and the database I was talking about above. Hope that's of some value for some of you.

2018-09-23

Here's what I've written about the possible developmental benefits of creating a ubiquitous "Internet of Rules"... chiefly, more inclusive markets: <link removed for privacy reasons>

2018-09-24

It would be really useful to se an example of what means by this. Yes all attempts to point at bits of the world by using words will carry difficulties. But how can the "real-world data" and algorithms be completely free of that? Someone has to enter the data, relying on natural language instructions. I was asked recently whether I should add a definition of "vehicle" to a new provision in a Law that already used the term extensively, but always in combination ("motor vehicle", "goods vehicle" etc) where the definitions explained the other element but not the vehicle part. But you have to stop defining somewhere and assume people know what the remaining words mean, or you end up in a loop (there are only so many words in a language) - if you define it as a wheeled thing for carrying other things, then do you need to define "wheel" and "carry". It also means opening yourself up to more risk of going wrong - what about monorails and sledges, but it can't just be carriers or you will catch bags, and you can't assume motors because you want bikes and so on. How does code cure that? I do see coding legislation as useful, but I can't see coding as inherently perfect and natural language as completely damned. I also can't recognise the picture in which coders must have all the answers and law must just be the problem - surely any progress will be m

2018-10-20

What you are describing is what we call the IDIOM. The idiom is the precise language used by the rules, and must be defined as part of the rules building process. In your example, what constitutes a vehicle depends exactly on the context prescribed by the rules. It could be any of your suggested definitions – that is why the real-world data has to be transformed into the idiom before rules can be used to adjudicate an outcome. In general, we find that the logic controlling this transformation makes up the majority of the algorithm code. The actual adjudication is usually quite simple in comparison.
In terms of sourcing the real-world data, then the first option should be to get it from an existing source where you can build in transformation rules that accommodate the known vagaries of the source. For instance, it might

be defined as a vehicle if it is described in Red Book. The second preference is for enumerations lists only.
Natural language is my last preference because it is impossible to confirm a person's understanding of the terms they are using.
We published a paper on Modern Analyst that discusses some of this http://www.modernanalyst.com/Resources/Articles/tabid/115/ID/3109/The-Role-of-SQL-in-Decision-Centric-Processes.aspx
Our submissions to the Business Rules Excellence Awards provides some useful examples of the application of this concept.
Available here
<broken link removed >
and here
<broken link removed>
Regards

2019-04-30

I just read your papers. Thanks for sharing. Good stuff.

IMPROVING POLICY, LEGISLATIVE AND REGULATORY DEVELOPMENT

2018-09-20

How do we make policy, legislation and regulation more responsive to feedback loops?
How do we make the process and results more transparent and participatory?
What needs to change in the development lifecycle?

2018-09-21

I still like the idea of coding in any post-implementation review requirements in a way that can be scraped. Then someone could set up a dashboard, split out by Ministry, of PIRs that are coming due, ones that are overdue, ones that have received a positive review, ones that have received a negative review....

I'm not sure that PIRs are actually followed through on all that much. Making it easy to check compliance and to compare records across Ministries could encourage better practice.

2018-09-22

Specify goals, specify measurable data points that represent the achievement of those goals. Hold people accountable for measuring the effects accurately, not just for achieving the goals. Focus on process, not outcome. Iterate. Find the authority to create regulatory sandboxes in order to facilitate experimentation with new sets of rules for experimentation. Find fair ways of granting access to those sandboxes, and ensuring that they don't create undue risk.

2018-09-23

I've written about the channels that policy development could take, namely the coding of "algorithmic trade policy" as part of the process of trade negotiations and the formulation of the terms of an agreement. Here may be an opportunity to express rules in a "lingua franca", computer language, to create interoperability across commercial/legal systems

<broken link removed>

2018-09-24

Changing regulation —let alone legislation— after the fact is immensely hard, and causes many very real problem of equality of opportunities and business environment stability.
I believe the best use of legislation as code in the legislative process itself is in ensuring impact assessments (mandatory in some countries) and amendments are all coded as branches of a main model (“reforms” in OpenFisca).
In order to do that, equipping pressure groups with tools that facilitate the creation and visualisation of such branches would increase the likelihood of one successful campaign being supported by legislation as code. After one such campaign will have changed expected outcomes, I believe most groups will be interested in such tools and the amount of legislation modelled as code from the onset will vastly increase

2018-09-24

Agree. Making it simple to create a dashboard indicating, for each Ministry, PIRs that are coming due, that are overdue, outcomes of completed ones - that kind of transparency can be powerful

2018-09-25

Precisely why a new trade agreement creates an opportunity to develop "digital schedules" to the, non-functional, text.

2018-09-24

Some comments seem to equate procedural rules and AI. They are different subjects and do not equate - but perhaps each has some basis for a separate discussion. These comments tend to sway towards discussion of in-built biases in the outcomes. In my view, the procedural approach at least makes these biases overt and visible. While I am sure this will be considered to be a negative in general, it is already widely used in Government algorithms - for instance, we have implemented ranked lists of race, with the somewhat arbitrary limit of participation in 3 races to get your final personal race ranking (try telling that to ancestry.com)

2018-09-24

The big opportunity is bringing the worlds of policy/legislation drafting and service delivery together. What do we need to have policy designers and legislative drafters express legislation in a different format than text? This needs to happen from the start of the policy design and legislative process. Policy designers/drafters need capability, frameworks and tools.

2018-09-25

What do we need to have policy designers and legislative drafters express legislation in a different format than text?

Deliver value to them.
Policy designers aim for effective public action and stable implementations over time. We can give that to them by building impact assessment tools and demonstrating how digital public services provide effective leverage to deliver policies.
We have proven that this does not *need* to happen from the start of the policy design. It would be more effective to do so (maybe not more efficient considering how many options would have to be coded), so we should probably aim for that. But we still have a lot to learn regarding the best time to encode legislation in software forms. Legislation itself is not written in legalese from the start ;)

2018-09-25

Commercial legislation is typically delivered through proxy documents that seek to represent it anyway. I don't see how this is any different than a menu at a restaurant being different than the recipes (e.g. laws in legalese) to make the food. We don't have to be chefs to eat at a restaurant, so why do we have to be experts to use commercial laws?

2018-09-26

In our work to date and engagement with policy designers and legislative drafters the business rules methods used by Inland Revenue (NZ) based on the Semantics of Business Vocabulary and Business Rules Standard have proofed to be very effective. Concept models, decision-models, and human-readable rules can be easily used as part of legislative processes. By using these tools and methods we could bridge gap between policy and service delivery. Can we agree on a set of (existing) standard based tools like that for use as part of policy design and drafting?

2018-10-29

I read about this the other day, thought it fitted in well with this thread.

A report by the Government Chief Data Steward and the Government Chief Digital Officer provides valuable insights into the use of algorithms by government agencies.

The report also suggests how their use can be improved for both fairness and transparency.

Computer algorithms (procedures or formulas for solving a problem or carrying out a task) have become an increasingly important tool for analysing data.

"The report released today shows how algorithms are helping us to deliver better policies and services, but it also reminds us of the need to take care in their use. There’s plenty of scope to lift our game,"

The report, a first of its kind in New Zealand and internationally, examines use of algorithms in 14 government agencies.

"Algorithms are an important part of government evolving to provide services that work better for all of us, and also make it easier for citizens to engage with government," Government Chief Digital Officer Paul James said.

The report includes case studies that highlight how algorithms are already enabling innovative solutions to complex problems.

One example is an algorithm being used by Work and Income to identify young people at risk of long-term unemployment, so they can be offered assistance. This provides a great example of the way these techniques can help those who may be in need.

Recommendations in the report include maintaining human oversight, involving those who will be affected, promoting transparency and awareness, regularly reviewing algorithms that inform significant decisions, and monitoring for adverse effects.

"New Zealand has robust systems and principles in place around the safe use of data, but as techniques become more sophisticated we must remember to keep the focus on people and make sure the things we are doing are for their benefit," Ms MacPherson said**.**

https://www.data.govt.nz/use-data/analyse-data/government-algorithm-tranparency

2018-10-31

Hi everyone, this is my first posting and I am not a techie, so probably coming at it from a different perspective. The discussion seems to focus on two main points, one is feedback loops and the other is how to best promote the idea of digitising regulation (which I am using as incorporating legislation). I would firstly get very clear on the objectives and develop a strategy as to how to achieve the objectives. Examples of what to include follows. Gather information and categorise it as to how it can be used eg get a list of policies currently being proposed to be developed from a variety of agencies/departments and categorise them as to whether they are good prospects for digitisation. Develop a 'business case' as to why digitise (including success stories and benefits from past instances), get a sponsor and engage with the policy advisers and other stakeholders eg consumers/customers/clients. Provide educational materials, tools etc to get them understanding and motivated. Include a life cycle of the policy development, drafting and implementation process and show them how and where the process of digitisation fits in. Use 'policy speak' rather than any other sort of speak. This would also show where feedback fits in which would be at various points along the way, not just at the PIR stage. Incorporate engagement/stakeholder management techniques to get active participation and include governance especially to get good decisions. Build towards outcomes and measure impact.

2019-04-26

A cautionary Tweet about the dangers of taking humans out of the loop: https://twitter.com/vgr/status/1121538583808974848?s=21

2019-04-26

Thank you for sharing! Totally, we’ve been discussing this for years and there are many examples that are scary... see the old Xalgorithms Tweet I just mentioned you in.

Thankfully, the Internet of Rules tech gives users complete choice and control over how to apply algorithms as rules. This ensures that compliance does not necessarily equal consent!

LEGISLATIVE IMPLICATIONS

2018-11-27

Technology is now an integral part of the way we answer our questions, and access our goods and services.

As many of our rules are set out in legislation, the software that drives that technology needs to reflect relevant rules set out in legislation.

This has created both a new audience for legislation, and a new way in which it is used.

Software, and the algorithms within it, are like an invisible level of law – they operate things, but are not currently seen, or visible in legal terms. They usually have no basis in law, but they are often running the law.

For the majority of people, software provides our day-to-day interface with the law - not the legislation itself. So, we can no longer assume that the primary audience for legislation is a human one – many of our rules are now primarily consumed and used by machines.

This poses 2 questions. First, how do we create machine consumable versions of legislation? This is the broad topic being discussed in this forum.

Secondly, what are the implications for legislation itself? What (if anything) should legislative drafters be doing differently? This is the topic that is to be discussed in this thread.

2018-11-27

I have set up this thread to enable conversations between those who are involved in (or interested in) the drafting of legislation, and the implications of the broader matters being discussed in this forum on the content and drafting of legislation itself.

Most of our rules have been developed for a paper-based world. As a consequence, they have not been drafted to enable digital processes, or with a digital end-use in mind. There are also barriers to digital processes built into our existing legislation.

Further, if we know that the rules in legislation will be replicated in software, what difference does that make (if any) to the way in which we draft legislation and make it available?

My assumptions are that—

1. this is the area where legislative drafters could add most immediate value in advancing digital transformation and open data; and
2. there is potential for some relatively immediate gains across a wide range of legislation; and
3. we can identify principles for drafters and instructors to follow, which can be set out in guidance materials and potentially have a wide impact on legislative thinking across governments; and
4. solutions are likely to be universal - they will be applicable regardless of jurisdiction.

I look forward to others participating in this conversation.

2019-09-21

Wanted to share this book who takes a very interesting anthropological approach to the transition from oral law, to written law and the ways that this influenced law's application (first few chapters). https://lawforcomputerscientists.pubpub.org/user/mireille-hildebrandt

The COHUBICOL project which looks fascinating.

2018-12-04

Does the creation of machine consumable legislation require producing the legislation in a marked-up language such as XML in the first place? Many smaller commonwealth jurisdictions may still be working based on simple word-processing publishing tools. To me, this seems like it would make that translation to 'code' that much more difficult. particularly where it would have to be translated over and over on amendment, rather than being embedded into the mark-up -- assuming that is where some of the keys to the translation to 'code' would reside.

2018-12-04

I do believe that any system of converting legislation into logic will require some sort of structured legislation, for exactly the reasons you specify. I believe (as a programmer working on this for many years) that the rules that are created are likely to be ordered differently than the legislation, such that there will need to be a link saying, 'this block of rules covers this block of legislation'.

2018-12-05

None of the jurisdictions I worked with used machine-formatted legislation as the source for legislation as code. They used legalese as the source.

In France, I worked with a startup (Contracteo, co-funded by the Transformation fund in 2015) and an NGO (OpenLaw) that tried to leverage the Parliament-produced databases (in particular the KALI database) to extract knowledge from them. The conclusion in both cases was that it was more costly to try and extract data from inconsistently marked up semi-structured text than to let humans extract knowledge from legalese.

I believe there is an overestimation of the quality of XML-distributed legal texts. In almost all cases I have seen, they are just wrappers around legalese with machine-readable metadata, while what we want / need / fantasise for legislation as code is that each data point in the legalese would be marked up individually with a unique yet discoverable ID.

2018-12-05

Well said, I agree. I question whether we could ever get to a point where we produce something better than "inconsistently marked up semi-structured text", given the variables at play within the English language (or any other natural language), legislation itself, differing regulatory schemes, etc, all of which are being produced and enacted in a political environment.

The 2 systems I have seen (one of which I trialed) required me to (in effect) re-write the legislation into a format that the system could then convert into a software code-like format. That middle step is inefficient and filled with risk (from a legal and legislative perspective). It requires a translation to produce another translation. And two translations are twice as risky as one translation.

2018-12-05

I work primarily with secondary legislation which is amended far more frequently than primary legislation. It is also where much of the details and formulaic processes would reside. I see it as a significant barrier to the process if every time a piece of secondary legislation is amended, the legislative text (sorry, can't use the term "legalese" -- this is something most legislative drafters strive to avoid in their texts) as well as the code translation need to be updated by humans.

I agree that the legislation needs to be well structured (presumably in XML or LegalXML), but wouldn't it be preferable for drafting offices to concentrate on that as a starting point, so that your data points in the legislative text could be marked up (embedded) and retain that mark up as they are amended?

If this conversation is to explore what legislative drafting offices can do, why not look at more than the organization of the legislative text and create structured, marked up legislation that would facilitate the translation to code?

If it is humans that will be reading the legislative text and parsing out the data points and extracting the knowledge, then what is it that legislative drafters can offer by way of changes to the actual legislative text?

2018-12-05

Hi. Really great post - I think you have got right to the heart of the matter.

I'm going to copy a comment I made in a different thread here, because I think the 4 options I will outline below may help create some structure around this conversation also. I will then add a further post more directly answering your question above.

My post was as follows:

Traditionally, we have drafted legislation and then published it so as to make it available for the world to interface with as they wish. In the past, the main users were lawyers and the courts, who are trained to interpret legislation.

Now, with advances in technology, the rules in legislation are integrated into tools - primarily computers - to assist people to do all sorts of things. This may simply be a website that sets out information, or may be a company's business rules which it uses to ensure regulatory compliance, or by a government department to work out people's eligibility for a benefit, or enforcement officers to determine whether people have complied with the rules, and so on. The uses are endless.

The one thing that all of the above have in common, however, is that they operate using software. So, the question is, what is the best way of replicating legislative rules in software? How do we do this easily, and ensure there are not gaps between the legislation and the software?

It seems to me that fundamentally there are 4 options (some of which have variations, but let's keep it simple):

1. We do what we are doing right now, which is simply draft and publish legislation and leave it for the world at large to solve this problem. For a whole bunch of reasons, I don't think this is an acceptable solution. One of the main reasons being that in places like NZ, the UK, and Australia the push by government to have integrated government services framed around citizens' needs requires software versions of legislative rules:
2. draft legislation as we always have, and then run it through some sort of "translation" tool that can take natural language and turn it into software. As far as I know, no such tool exists - although people are working on it:
3. develop a tool that is used by legislative drafters that allows a single input (ie, legislative drafting), but produces 2 outputs (ie, natural language legislation, and an equivalent software version). Again, as far as I know, no such tool exists - although people are working on it:
4. take the Better Rules approach in which we take a different approach to the development of policy and legislation. The primary outputs of the Better Rules approach are concept, decision and flow diagrams. These are in effect a common language that are then used as the instructions used by legislative drafters, business rules folk, and software developers, all of whom can then draft their particular outputs using their current, usual tools and processes. The outputs all need to be checked and validated against each other, but as all parties have contributed to the development and creation of the concept, decision, and flow diagrams, everyone should be speaking the same language - conceptually and literally - from the outset.

For a whole bunch of reasons (which I am happy to expand on in another post), I believe that option 4 is the most viable, the best, and the most forward-looking approach. At least for the foreseeable future.

2019-01-14

Can you expand some more on how your option 4 is different from something like Oracle Policy Automation? Is it just that OPA is proprietary and opengov people are looking for an open solution instead? If so, what is involved in that, and what precedents are there for govts successfully developing open versions of existing commercial software? If not, what are the differences? If I am barking up the wrong tree (very likely), what is the right tree? Thanks for your work on explaining this by the way.

2018-12-05

If you would like to see a real-world example of what the digital transformation of government looks like, and what the implications of this might be for the legislation we draft, have a play with the financial help page on SmartStart (the website for new and expectant parents) - see here - https://smartstart.services.govt.nz/financial-help.

It takes only a couple of minutes – it’s dead easy (and its actually quite fun).

Go to the “Get started” button at the bottom of the page. Then, answer the questions that you are given by pretending to be someone in fairly dire circumstances. Ie, solo parent, 4 children, 1 with a disability, low income – or similar circumstances.

It then tells you which of 17 different benefits you are entitled to.

This is a great example of—

- an integrated service (ie, 17 different benefits provided by (I think) 4 different government departments)
- a service based around a life event (ie, having a child)
- a service that is framed around the needs of citizens (ie, what am I entitled to?)
- a service where government is invisible (ie, you don’t even need to know these benefits exist, or who provides them, and there is almost no mention of government on the website)
- an integrated public/private service (ie, if you look at other parts of the website you will see that it integrates with services provided by Plunket, the Midwifery Association, baby car-seat rental places, and so on).

The point of this from a legislative drafting perspective is, what do we need to do when we are drafting legislation (and when the policy for that legislation is being developed) to—

- ensure that we are not putting barriers in the way of services like this being created and expanded (ie, by being specific about things like the method of communication, by requiring paper-based forms of application and approval, by having be-spoke systems that are not aligned with existing systems so as to enable them to be integrated, and so on):
- ensure that the language and definitions we are using enables integrated services to be provided (ie, services like this only work if you use the same definitions of child, income, disability, and so on. If each of these is different in each piece of legislation, then the services can’t be integrated as well as they otherwise could. This is the value of a policy/legislative ontology):
- ensure that the rules in legislation can easily be replicated into the software that make this website function (ie, producing machine consumable legislation).

All sorts of other core legislative drafting and publication issues arise also.

There is a tendency in this area for people to focus on the machine consumable legislation side of things - which is what most of this forum is focused on. But there are plenty of other legislative drafting implications that we need to consider also, and that we can advance regardless of whatever is ultimately done in the machine consumable legislation side of things.

In short, how do we draft legislation to ensure it effectively enables the use of machines in whatever way they may be used in the future?

2019-01-13

Hi all,
I am a computer engineer with no experience or knowledge of legislative drafting, still interested in the concept of legislation as code.
I'd like to ask if anyone here could give me a specific example of the process, the steps they take during the creation of an actual new law, or updating an old one.

I am asking this because if the assumption that law needs to be represented in code stands, then I believe it'll be legislative drafters will be the ones who will have to write that code (maybe using some tool for that).
If we want legislative drafters to produce legislation as code, then I'd like to know how they operate to be able to find out how we could make producing code a useful thing for them that makes their lives easier and just an additional task, yet another form to transform legislation into.

2019-01-14

Hi - I work for Inland Revenue New Zealand as a business rules analyst and we've been exploring this space a lot. I would argue that legislation drafters aren't necessarily going to be required to produce "code" as legislation drafting itself is a highly technical skill, and expecting all legislation drafters to be able to produce digital code, let alone legislatively binding digital code, would be asking too much in my opinion. The question the team I work in is currently exploring is more about what can be done differently so that legislation is written in a way that enables the delivery of digital services.

You may have heard of the Better Rules discovery sprint that took place at the start of 2018. If not, you can find a link to the report attached. Last year the NZ government announced it would be implementing a R&D tax credit, and we have taken the findings from the Better Rules discovery sprint to adapt the way we've developed the R&D tax credit policy.

Basically, we tried to follow the Better Rules discovery sprint's findings as closely as possible. That involved getting together a multi-disciplinary team of policy analysts, legislation drafters, service designers and business analysts from the outset. We've been working in sprints, so that we can regularly review and improve how we are working and what we are delivering. Thirdly, we used decision models and concept models to maintain an understanding of the policy. Breaking legislation down into models means that people without a legal background, who may struggle with understanding legislation, can provide meaningful contributions to policy discussions as they are able to understand what the policy is doing and what it is trying to achieve.

Creating decision models that are endorsed by the policy analysts developing the actual policy also means that you have an isomorphic representation of the policy that is technology agnostic. By that, I mean the policy is broken down into yes/no questions that ultimately answer a taxpayer's question of "how much R&D tax credit will I get?".

Hopefully this provides you with a better idea of what "legislation as code" might look like in practice. Legislation is already a code, which is traditionally consumed by humans. The challenge is adapting the legislative development process so that legislation enables the creation of machine-consumable rules, as well as maintaining the requirement for rules to be human-consumable.

Better Rules report

2019-01-14

Re comments below: It would be very valuable if the rules related outputs (concept diagrams, code etc) from a multi-disciplinary team (eg drafters, policy-makers, rules analyst and others) are recognised as part of the legislation or regulation. Even if the drafter doesn't create the code or rule statements, all or part of these outputs should ideally be regarded as part of the drafting process.

2019-01-15

Hello again. I have a few general questions for the NZ government participants in this forum. Certainly your team is doing interesting work in this domain towards automating legislation. What I've not quite understood is, basically, where do you now stand? Has your work of the past few years brought you to the point of trying to keep up with a strong momemtum of proliferation across many domains of legislation, or to the point of being stymied with issues of methodological scalability, or funding, or some other complex bothersome constraint? Having implemented OpenFisca, is your work so far now dependent upon that platform from here forward, or can your work-to-date be readily ported to some other platform that may come along? Are some aspects of your current work causing the team grief, or is it all going along smoothly, and the only constraint is the number of hours in a day?

2019-01-15

Hi. Following on from comment, we've used the Better Rules methodology to develop legislative rules that can be implemented digitally. But right now, we need to do additional work on standardising the representation of rules for consumption by different systems. Ideally the representation would also be understandable by different stakeholders (not just written as code).
We are not at the point where legislation can be directly consumed and implemented by digital systems, but we've used concept models and decision models to demonstrate legislation to stakeholders who have developed their own digital solutions.

Once we have the standardised representation to catalogue rules and concepts across NZ law, and a system to manage the catalogue, then I think the only constraint will be the number of hours in the day.

2019-01-16

A few comments on your questions. As always it is not clear cut.

1. We did the Discovery work in Feb 2018 as a cross-agency group. This resulted in a model how it can be done. Since Feb 2018 we have been working on specific cases to test and demonstrate the model works. Current test cases are in tax, employment and social services. For 2019, test cases are likely for 4 – 5 other domains. We are not set up to "churn" through existing legislation to turn it into "code". Our approach is more targeted towards legislation that is new/revised or legislation for which digital services are redeveloped.
2. Adoption across NZ public sector depends on agencies to see value for their customers, or themselves, and the view of the agencies responsible for drafting legislation in New Zealand. At this stage the use of the approach is voluntary. In summary, the momentum in New Zealand needs to be maintained from the ground up.
3. The proposed model (page 31 of the Discovery Report) advocates for multiple outputs – concept models, decision models, flow diagrams, human-readable rules and code. The first 3 outputs are standards based but are not dependent on a particular technology platform. The Service Innovation Lab used OpenFisca to capture the rules as code in a publicly accessible rules engine. All of this is still an "emerging" architecture and not fully developed nor documented. Our aim is to do more work in this area in 2019 and socialise with this discussion group.
4. The longest track record (6 years) and maturity with this type of model is in the tax domain. There are no strong impediments using this model but it requires a change in culture and capability within government agencies. This is what takes time. There is a tendency to gravitate to a discussion about "what rules engine you use" but we are of the view that the core outputs should be technology agnostic. As we do more of this work our understanding of the requirements and possibilities for future platforms is growing and will help inform future decisions, the expectation being if better options are identified it will be possible to swap them into the work already done. (Note the NZ tax rules are not published through the OpenFisca platform).
5. For this approach and model to be incorporated as part of policy and regulation development process, independence is an important element. The approach/model creates a "blueprint" of how the regulation works and it is important to make sure it can be translated/consumed into the appropriate technology set – whether this is a rules engine, AI expert system with machine learning or a simple core government system.
6. One of our objectives is to increase the utility of the rules and enable non-government entities to develop products and services. For example, pay-roll services or export/import services. As such, we would to know from entities like Xalgorithms, how they would like to "consume" the rules. Do they want the rules as code? Would OpenFisca as a rules

engine work for them? Do they need the rules as code only or do you prefer the whole stack of rule artefacts? This is all part of the learning we are doing as we go, and as the concept is still in its infancy we are helping the wider ecosystem to clarify potential future opportunities and needs.

2019-01-16

When you mention standardization the representation, what do you mean exactly?
Picking or creating a programming language or data format? Or really go heavy with like an IEEE level standard?
Where are you currently at with this? Still looking for candidate languages or formats?

A more general question just as a thought experiment. What do you think could be appropriate action if you pick something now and another, better thing comes along later, years from now? Could you imagine an incremental "refactor" into the new language, data format, when for a time legislation would be described by more than one representation?

2019-01-16

As a business analyst, I'm looking for an IEEE or OMG level standard, along with the software tools to manage the models. I don't want to reinvent the wheel.

I like the decision model notation standard, and it has good software support, but I don't think it can represent all the legal concepts necessary to describe legislation for translation into code. The SBVR standard models the logic of obligations and permissions, but I don't know how easily SBVR models can be implemented as code.

To answer your though experiment, if the standard is suitable I don't see the need to replace it with another standard. And I'd be very interested to see what qualities the better standard would have

2019-01-18

I am asking these questions, because I have been playing around with this idea of legislation as code, but from an empirical direction.
Picking a programming language that is easy to parse e.g. Javascript, with a tool to represent it to non coders, e.g. parse Javascript into AST form and present that as a flowchart.
Later on as new languages come up e.g. Prolog as necessary inputs, the AST would need to be constrained.
Other kind of outputs, beside the flowchart representation e.g. generating back the legal text from code could constrain the AST even more.
In time it'd probably represent a domain specific language that does not necessarily have to exist, just its AST has to be specified.
It'd open up many opportunities obviously. Language agnostic representation, even human language agnostic in part, comparable structures in legislation, etc.

But if the above mentioned level of standardisation is necessary for using a solution at scale then the empirical approach is useless.
What is your opinion about this?

2019-01-19

The OpenFisca team tried to go the AST way to mix different sources of truth (OpenFisca-France, French tax department source code, independent researchers datasets…). What I can tell you is that they spent quite a few months on it until I took the lead on the project and stopped that investment. My main rationale for it is that you still need one source of truth, so if you end up considering your AST as that, you'll end up writing the AST directly… At that stage, you might as well write in any other language, consider it "blessed" and write transpilers from any other language to/from it. Or just go the API way and standardise data and messaging formats rather than a way to write code
References: derelict parsers, OpenFisca documentation, interactive API documentation

2019-01-30

Hi, I've been quiet since my question two weeks ago, which received some interesting and thoughtful replies. I'd like to comment back to one of the points raised by and then to several parts post.

RE: "legislation drafters aren't necessarily going to be required to produce "code" as legislation drafting itself is a highly technical skill, and expecting all legislation drafters to be able to produce digital code, let alone legislatively binding digital code, would be asking too much"

While I appreciate that sentiment, if I understand you correctly I would not agree. Regular schools now treat programming competence as essential (for example: https://www.researchgate.net/publication/284139559_Computing_our_future_Computer_programming_and_coding_-_Priorities_school_curricula_and_initiatives_across_Europe)
And similarly it must be with the legislative drafting profession. I think that legislative drafters really must now at least be able to collaborate effectively with programmers. I'm not saying it's easy. I'm saying it's unavoidable. Those who make this change will be more successful than those who do not.

RE: "One of our objectives is to increase the utility of the rules and enable non-government entities to develop products and services. For example, pay-roll services or export/import services. As such, we would to know from entities like Xalgorithms, how they would like to "consume" the rules. Do they want the rules as code? Would OpenFisca as a rules engine work for them? Do they need the rules as code only or do you prefer the whole stack of rule artefacts?"

Let's find out. Please suggest to us two or three of your genuine segments of legislation currently expressed for OpenFisca, and let's find out what we need to do to ingest those from your OpenFisca platform into our Internet of Rules platform. Can you please choose a simple, and a moderately complicated, and a rather complicated example for us to try? Just provide us your three rules as code -- can OpenFisca run an export of a complete rule including its metadata? We could post our interaction on the "Making Rules Into Code" discussion thread <broken link removed> I reckon it would be interesting for others, and there would likely be some good comments and suggestions from others too.

RE: "It would be very valuable if the rules related outputs (concept diagrams, code etc) from a multi-disciplinary team (eg drafters, policy-makers, rules analyst and others) are recognised as part of the legislation or regulation."

Does everyone here agree that programming code should be in a schedule so that bugs can be readily fixed. (That's what I think.) Or are there different opinions about that?

RE: "we need to do additional work on standardising the representation of rules for consumption by different systems."

Being platform agnostic is one of the reasons Xalgorithms chose JSON tables to express the rules. The metadata about each rule is also contained in a JSON table, since it determines when a rule is "in effect" and when it is "applicable". We call this specification "Xalgo" and it constitutes a domain-specific language with deliberately low expressibility. Really it is just commonly stuctured tables that are lightning-fast for network transmission and in-memory computation, or are simple to automatically port to any programming language.

RE: "Ideally the representation would also be understandable by different stakeholders (not just written as code)."

Our team wants the rules code to be easily verifiable by non-programmers. Authoring does require more skill than reading and comprehending. This distinction applies to any writing: Anyone should be able to read a novel; only some people are excellent authors of novels. To author the computational version of legislation requires sequential logic precision and awareness of available data sources and characteristics that is easily beyond what most legislative drafters otherwise employ. For example, a legislative drafter might say in English that something is indexed to the rate of inflation. To put that into working code is very challenging. (I'll be pleased to expand on that if anyone wishes to explore that particular example.)

RE: Once we have the standardised representation to catalogue rules and concepts across NZ law, and a system to manage the catalogue

The inclination I think I see in the NZ work so far is to have a system particularly for legislation. Also OpenFisca is particularly for legislation. The way our team has prefered to approach this is to solve for the more generic requirement, and to ensure the set of use cases involving legislation are satisfied along with use cases involving, say, collective agreements, loyalty programs, machine control systems, dynamic price benchmarking, and so on. That's to say, our approach is similar to the generic "rules" perspective of RuleML. The difference is that we find RuleML has much too steep a learning barrier for non-programmers just to validate, and for authors in business and government to really embrace in general operations. A significant difference in our approach is to focus on rules that are declarative. RuleML deals a lot with inference. I'll expand on that below.

mentioned Blawx https://www.blawx.com/ which is an implementation of Blockly https://developers.google.com/blockly/ which itself is conceptually a derivative for adults of the original Scratch story-maker for kids https://scratch.mit.edu/projects/editor/?tutorial=all Blawx and Blockly provide a good interface for many legislative authors, but the code it generates 'behind' the GUI still has to be easily verifiable by those authors or anyone else. So I consider that UI to be useful as a optional component on top of any platform (including say, OpenFisca). The actual code that the computers run still must be readily accessed and accessible.

RE: We are not set up to “churn” through existing legislation to turn it into “code”.

In our view, that churn's got to happen early. If it cannot, that implies your solution is too complicated.

RE: Note the NZ tax rules are not published through the OpenFisca platform.

Why not? Isn't that where you're intending to go?

RE: There are no strong impediments using this model but it requires a change in culture and capability within government agencies.

Are you not actually saying that the strongest impediment to your current model is that it requires a change in culture and capability within government agencies? Organizational anthropology has broad foundation, but it's a big mountain to climb: https://journals.sagepub.com/doi/10.1177/001872679705000905

RE: multiple outputs – concept models, decision models, flow diagrams, human-readable rules and code

Professionally I like all these things. In my Canadian Government experience though, I have found that: (a) Most people don't like having to deal with concept models, decision models, flow diagrams (e.g. UML, which I helped to introduced in the Canadian Government around 2003/04) -- mostly, they're jsut pleased that others seem to be looking after all that logical rigor! (b) Such documentation tends to gets out-of-date really quickly, and updating it is time-comsuming. But rarely does anyone have the time to keep all those updates going (esp. amidst staff turnover). So they get out of date... and then one is faced with determining what's worse: no documentation? or wrong documentation? Just the volume of documentation itself becomes an enormous burden. So most documentation of the type you mention ought to be at the level of modular re-usable design patterns. What we have attempted to do in our Xalgo specification is unify the documentation as computable metadata with a tabular stucture that is easy for non-programmers to understand and yet is directly computable, and that includes accurate direct transcription into any other programming language for any platform.

In closing this rather long post, I'll pick up on my earlier comment that a significant difference in our approach is to focus on rules that are declarative. Permit me to draw upon an analogy from outside our domain of automating legislation. The following excerpt showed up in an interesting place...

"That led to some counterintuitive strategies, like minimizing the role of artificial intelligence. “High-end artificial intelligence and deep learning are higher-order functions,” says Van Buiten. “Higher-order functions are difficult to certify. Until we know how to do so, we want to use more deterministic methods.” That means using systems that don’t rely on interpretation or guesswork—which AI is essentially an advanced form of—but on defined and predictable behaviors"

I came across this statement in the following WIRED article about Sikorsky Autonomy Research Aircraft (SARA), i.e. self-flying helicopters...
https://www.wired.com/story/sikorsky-self-flying-helicopter-sara/
https://www.lockheedmartin.com/en-us/products/sikorsky-matrix-technology.html

Facing the enormous and ever-changing dynamic complexity of legislation, our approach has similarly been to stick with declarative programming which allows for deterministic semantics even when the environment might be highly non-deterministic. We match discrete rules (ought) to factual events (is).

2019-01-30

quite a few comments and questions you posted, but that is also why we want to run this discussion platform. Will see if we can use some of the more dynamic features of this discussion platform/software to get responses on your questions/comments.

2019-01-30

Hi

We might be talking about two different things - i agree to some degree that sometime in the distant future we could expect all citizens to be able to understand and write in software code, but right now that is not the case.

There are definitely legislative drafters starting to better understand software coding so they can tailor what they deliver (i.e. legislation) to enable better digital government services. But turning ALL legislation into software creates a new argument - how can citizens be engaged and hold government to account when they aren't capable of understanding the rules they are subject to, because some people don't understand software code. I will note that there is an argument that legislation isn't as widely understandable as it possibly could be currently either.

So maybe we could get legislation written as software code, albeit at the expense of government accountability. What the Better Rules approach is about is that legislation should be modelled before it is implemented, meaning that the legislation can be analysed more widely by people with differing backgrounds, to test whether or not a policy actually achieves what its intention. Having people with diverse backgrounds test the impacts of proposed policy reduces the risk of unexpected impacts. Using concept models and decision models is simply the method we have used to model legislation, as we have found these techniques to be understandable and explainable, but they are also far from perfect.

2019-01-30

RE: all citizens to be able to understand and write in software code, but right now that is not the case
That's not what I said. But it is possible to express each rules in such a way that non-programmers who want to take look can fairly easily understand

how it works. The skill level can be about what's required to check simple formulas in a spreadsheet.

RE: turning ALL legislation into software creates a new argument - how can citizens be engaged and hold government to account when they aren't capable of understanding the rules they are subject to, because some people don't understand software code. ... So maybe we could get legislation written as software code, albeit at the expense of government accountability.

Well, when you or I do our income tax online at present, can we even take a look at the source code that's running? No. Surely a common accessible way to express legislative rules in code provides a pathway towards greater accountability, no?

RE: What the Better Rules approach is about is that legislation should be modelled before it is implemented

As I said, I'm not against the modeling part. But you ought to address the issues I raise about the maintenance of those models once this is widely implemented.

RE: legislation can be analysed more widely by people with differing backgrounds, to test whether or not a policy actually achieves what its intention.

Testing effects can be achieved with automated deployment of legislation combined with automated collection of direct or indirect effects indicators. Modeling can help towards anticipating those effects. But the models themselves will usually become outdated rather quickly. They would be most useful as temporary planning tools, I reckon.

2019-02-06

RE: Please suggest to us two or three of your genuine segments of legislation currently expressed for OpenFisca, and let's find out what we need to do to ingest those from your OpenFisca platform into our Internet of Rules platform.
See https://www.rules.nz

2019-07-09

The European Commission is conducting a survey into 'Legal Interoperability' and from this potentially create a community on the topic. It appears to have relevance to the discussions and work of the people in this forum.
Please take a look and fill it out (they say it should only take 15 minutes)
<broken link removed>

2019-09-21

Hi, I am not able to access this survey. I am interested to take this.

2019-09-21

I think the survey has closed.

MAKING RULES INTO CODE

2018-09-20

What are the different approaches we could use to make rules into code?
What are the pros and cons of different approaches?
What do we need to consider?

2018-09-20

Hi all, In our own work, to separate thinking about a rule, from thinking about how to express a rule in code, we use "a+b=c", where "a" is a price, "b" is some per-transaction flat fee, and "c" is the new combined price. Several additional data elements need to be rolled in: jurisdiction, date/time of this rule being in effect, rule version, etc. For example here's that rule in Xalgo: https://github.com/Xalgorithms/general-examples/tree/master/org.xalgorithms.examples.a_plus_b
I would like to suggest that various methods of "making rules into code" also use the simple "a+b=c" function, with similar context data (which need not be identical) so that we can all readily see the differences and similarities, and learn from each other.

2018-09-22

Some questions that come to mind: Is the code authoritative, or not? What happens when it is wrong? Should it be encoded by a central authority? What is the recourse available for a person who relies on an incorrectly encoded law to their detriment? Should encoding laws be a thing that is done by or certified by insured legal professionals? What standards would those people to use to decide whether it had been coded correctly? To what extent must an encoding be aware of what it cannot do? What standards should be adopted? Should they be applied at the time we are writing the rules, or only at the time they are encoded?

Do we want to facilitate the ability to make automated decisions according to reasoning by analogy, or only deductive logic? How can we do that?

Some of the things that need to be considered are the effect that encoding the laws, or writing them in a way that makes them more easily encoded, has on the discoverability of the law, and the presumption of knowledge of the law. The effect that the automation of regulatory question-answering will have on procedural fairness. How and when discretion should be built into the rules explicitly, where it exists only implicitly now. The utility that vagueness has from the perspective of legislative drafters, and how to maintain the benefits of vagueness at the drafting stage while still increasing the degree to which the resulting rules can be automated. The fact that a written law is not merely a set of rules, but it is also a means of communicating a set of social standards, which could be potentially less communicative if it is written in such a way as to allow automation of the rules. Are readability and automatability in conflict with one another, and what is the right trade-off? Does automating policies risk entrenching institutional bias, and how can we avoid that problem in the course of building these sorts of tools? Is that a proper objective? What role does the encoder play in our legal system, and what qualifications should they have in order to properly play that role?

2019-04-24

We could have a thread for each of your questions.

2018-09-22

Hi. Firstly it is worth reflecting that today the rules are encoding by myriad people across myriad business systems every day, without any checking, authority or qualifications. Lawyers interpret, analysts analyse, software folk hard code the rules into business systems (like government services or banks) and every steps creates challenges of interpretation, consistency, correctness and accountability. So this is not new in one respect, but what is new is the notion that the rules set by govts could be written and provided authoritatively in both code and human readable ways, which improves consistency, accountability and traceability. Hope that helps. Rules as code makes testing easier and confidence higher, and dramatically redices the wasted effort and divergence from source we have today.

2018-09-23

RE: "the rules set by govts"
Often government entities deliberately outsource that responsibility in ways that result in enormous redundancy across implementing platforms, or de facto lock-in to some particular vendor's or entity's platform. The aspect we're pursuing via an Internet of Rules concept is for govt (and any other type of enity) to post and fetch algorithms via any platform. We set out to describe and design the absolute minimum required to achieve that with free/libre/open components and specs.

2018-09-26

The rules to me seem like a set of 'conditions' and 'results'.
So each rule would need to be broken down into these two types.
A+B=C has result of C based on the conditions of the values of A and B when added.
It's easy to generalise this but real world sets of rules add so much complexity and until some rules are translated/disassembled it's hard to get a practical idea of what is needed/useful.

2018-09-26

RE: "It’s easy to generalise this but real world sets of rules add so much complexity and until some rules are translate/disassembled it’s hard to get a practical idea of what is needed/useful."

Can you explain what specifically you mean by a "real world rule" that can't be easily represent in the manner illustrated? Please provide a URL to a section of legislation or regulation with a computational rule, and my colleagues and I will give it a shot.

You will find a genuine property purchasing tax rule from Singapore (sg-bsd-tacx) and a gas tax reduction rule from Quebec (qc-gas-tax) at the following link. https://github.com/Xalgorithms/general-examples We have a few updates to do to bring them up to date with our current version of the Xalgo domain-specific language -- the a+b=c rule is the most current presently.

2018-09-26

Due to the large scope of existing legislation there is not one code rule that fits all and each segment would need to be hand coded basically. http://www.legislation.govt.nz/ is a huge.

What is the purpose of the function of the code, is it to generate some kind of search engine or expert system that when given a well form syntax a result is generated?
Or is it a set of unit tests that are run on legislation to find possible issues/conflicts. Which would be hand coded again.

I don't understand objective as a whole.

2018-09-26

RE: "there is not one code rule that fits all and each segment would need to be hand coded basically"
Certainly, each clause or sub-clause of legislation or regulation with a computational rule will need a corresponding entry in a schedule. It should be in a schedule to the legislation or regulation so that bug-fixes need not go back to the legislature. This is not as onerous as it might seem, since there aren't really that many "rule patterns". This is illustrated in our Singapore property sales tax and Quebec gas tax reduction examples. They happen to use the same adjustment structure. We also have some payroll rules from collective agreements that use the same pattern. We reckon this will end up to reveal maybe two or three dozen rule patterns for all government's computational rules in the fiscal, trade and employment domains which we are addressing first (i.e. where input data packages conform with ISO 19845 or ISO 20022). Automating other rules such as, say, railway corridor clearance standards, or elements of building codes, can also be used to automatically validate technical drawings. It's actually from that domain that we learned and adopted in Xalgo the basic RASE table (Requirement, Applicability, Selection, Exception). What we hope to contribute is a simple way for regulators to supply the basics of such rules to all the competing platforms that implement "building information modeling" (BIM).
https://www.sciencedirect.com/science/article/pii/S0926580515000370
https://www.researchgate.net/profile/Wawan_Solihin/publication/273771092_Classification_of_rules_for_automated_BIM_rule_checking_development/links/5744837408ae9ace84217f9d/Classification-of-rules-for-automated-BIM-rule-checking-development.pdf

RE: "What is the purpose of the function of the code, is it to generate some kind of search engine or expert system that when given a well form syntax a result is generated?
Yes, a generalized way to publish and fetch declarative rules on the Internet, from and to any platform, for any context.
https://xalgorithms.org/internet-of-rules/
https://github.com/Xalgorithms/general-documentation/blob/master/arch-2.0.md

RE: "Or is it a set of unit tests that are run on legislation to find possible issues/conflicts. "
Yes, necessarily that too, with integrated versioning.

2018-09-26

I'll respond

In a way, you can see the rules as a "sort of" expert system. There is a micro-ontology that classifies rules according to effectiveness (whether a rule is valid at a certain time in a certain jurisdiction for a particular actor) and applicability (if a rule is valid in these circumstances, will the rule evaluate to a meaningful decision?).

On the other hand, once this micro-expert has evaluated the circumstances and context of the rule, evaluating the rule is simply a matter of evaluating a set of decision (https://en.wikipedia.org/wiki/Decision_table) and control (https://en.wikipedia.org/wiki/Control_table) tables that encode the "logic" of the rule. These tables are the "DSL" that Joseph references.

As for hand-coding the rules... Yes, someone has to code the rules. Preferably this would be done by the entity that wrote the rules (or the legislation) in the first place. This seems like a daunting task, but really, it's already being done. There are numerous solutions for things like trade, payroll, etc that have already been implemented. The "rules" for the laws or policies that these solutions enforce have been coded, by hand, into the solution itself (as custom work). Our DSL is designed to allow one encoding of the rule to be distributed to many solutions that are required to evaluate that rule.

2018-09-27

Hi. These references are super interesting. I have a couple clarification questions to make sure I fully understand your context :)
Reading the BIM paper, it seems to describe application of compliance rules in a context where the elements to validate (i.e. building plans) have already been fully digitalised by the entity interested in validating, in a domain where all dimensions of the element to model are known and objective (they are the three spatial dimensions). Is that correct?
Reading the description of Xalgorithms, can you confirm its aim is to provide a meta-language to express algorithms independently from any programming language implementation, distribute this meta-language, and possibly provide compilers from this meta-language to concrete languages so that the algorithm can be executed in an interoperable fashion? Or am I missing the point?

2018-09-27

Thanks.
RE: " the BIM paper... all dimensions of the element to model are known and objective"
The origins of BIM rules automation is to take electronic blueprints from CAD drawings and run them through a review for compliance with many straightforward rules (electrical standard specs, building code req's, bylaws, etc.). The conceptual framework which that community has put together has more general utility for other domains.

RE: "Xalgorithms, can you confirm its aim is to provide a meta-language to express algorithms independently from any programming language ... so that the algorithm can be executed in an interoperable fashion?"
Yes, although it is domain-specific and so minimalist that it barely looks like a language. The rules are just tables, really. Standardized digitally signed data packages meet standardized digitally signed tables. On the old 80:20 rule, this method can probably automate about 80% of the rules that have to run, which are declarative, and doing this would free up effort and resources for the rest which are genuinely complex rules (e.g. requiring human judgement).

Your use of the word "interoperable" is appropriate... It's one of the reasons all our work is free/libre/open, so that any rule authority can publish rules to the Internet and any platform can fetch them, run them in their native form, or easily transcribe them into any more expressive language they wish. For example I'm presently writing the Xalgo for a price adjustment clause in contract for use in determining monthly developer fees for a property-financed railway project (i.e. fees will be dynamically linked to attributable increments in property income & asset values). I certainly would not want to require that all the parties across 16 municipalities in two provinces have to use the same software. With similar generic flexibility in mind for commercial trade, I'll be chairing a session entitled "How can the silos that prevent interoperability be removed?" at the World Trade Symposium in London this December
https://worldtradesymposium.com/symposium-2018/agenda/

2018-10-01

Interesting, thank you for these details
One thing I have noticed is that over time some “simple” rules (the ones you could put in tables) can evolve into “complex” rules (ones you need to code algorithmically).
How can you manage such changes in Xalgorithms if they are represented differently? :)

2018-10-02

RE: over time some “simple” rules (the ones you could put in tables) can evolve into “complex” rules (ones you need to code algorithmically)

May I suggest that is a misconception for the following reasons:

1. Rules that can be expressed in clear sentences are either simple, or they are compound chains of simple sub-rules. In Xalgo we can forward-chain and backward-chain any rule table to one or more other rule tables.
2. When you say "complex" I think you mean "complicated", with lots of sub-rules. Complex rules are unique and unpredictable, with multiple feedback loops, discontinuous trajectories, potentially having several variants in play at the same time, and they are fickle in the sense that behaviour of the ruled may affect what the rule requires. Ever tried to step though an immigration process based on the verbal information provided by the people at the consulate or embassy? Ya, like that. They

often don't even know some of the written-down rules that apply to a given scenario.

3. Currently we're extending Xalgo to handle rules that have a nested hierarchy of dependency on other rules and dynamic data streams. This should be suitable for rules as challenging as OTC derivatives. And yet it's all done with repeating a very simple two-table structure. Documentation on that is coming soon.
4. The question you're asking would be best framed by pointing to a sample genuine rule expressed in natural language. Send a URL to a rule, and I'll be happy to express it in computable form to illustrate. (Please don't make up a hypothetical rule. Rules in actual legislation, though often very challenging to read due to the legalese and precision packed into them, are usually not terribly hard to code. But made-up sample rules tend to have far too much ambiguity to work with.)

2018-12-04

I took a look at the examples. But, not being an IT whiz myself - I found myself thinking that perhaps it might be easier to envision how the code is translating the rules if the natural language version was also there to see?

2019-04-19

What if those calculations - say in the Building Code - are dependent on a defined term, the definition of which is vague and somewhat flexible, and up to the inspectors to interpret. Or a very subjective term (we call them "unqualified modifiers") that has no fixed meaning. "Smart" regulations have meant changing rules from "the barrier must be X cm thick" to "the barrier must be adequate". How will a computer handle such a rule?

2019-04-24

Calling out to a specifically identified "oracle" for discretion (usually a human or committee), like your inspector, is one idea. Better to have inspectors judge small details that determine major decisions, than to have them make the major decisions directly.

Often, despite their fundamentally subjective/vague nature, such terms can be partially defined by other objective/precise terms, which in the best case is enough to automatically resolve all but the trickiest borderline cases (for which you call the oracle). For example, we might have a pair of rules that say "if the barrier is at least h/2 cm thick, then it is adequate", and "if the barrier is less than 1 cm thick, then it is not adequate", where h is some related dimension. That rule might handle 80% of cases, leaving undecided the cases of barriers between 1 and h/2 cm thick.

2018-09-26

I differentiate between 'code used to build rules' and 'code use to build tests for those rules'. Even if the former don't exist, the latter can be created. This is much the same as software development, where test coverage is incomplete but key behaviours are encapsulated in automated test-code, usually in an entirely different language than used to create feature-code.

In my current agency, we already do that sort of testing with proposed changes to regulations or legislation; we just use English to do so. This is a rich testing method but ambiguities and conflicts are too easily overlooked, and the very skilled people doing the work have a very difficult time engaging with external *affected* stakeholders.

2019-04-24

I'd love to hear more about your experience. Probably the others would too.

2018-09-27

Re comment: "I don't understand objective as a whole"
Government services are increasingly delivered or managed through digital systems - entitlements, compliance regulations, tax etc. All these services are underpinned by legislation. Legislation is written in text and an army of coders is busy translating the text into rules that can be incorporated in software. The problems with this: translation errors, work duplication and problems with legislation with sub optimal design. Expressing legislation in machine consumable rules or code will mitigate these problems. in particular when policy designers and legislative drafters take ownership of this process. In the near future software and "machines" are likely the primary users of legislation. This doesn't apply to all legislation but most legislation impacting citizen or business entitlements or obligations will.

2018-09-28

view reflects our vision almost perfectly :-) then again I'm from the Tax Agency, which is a highly automated Information Crushing Factory already as it is. But a change in tax legislation, however small it might seem, still leads to costly implementation projects not only in our agency > two small paragrahps of law = changes in the taxation system, costing roughly X million euros. Not acceptable in AD 2018...

2019-04-24

What is the Tax Agency doing about it?

2018-09-27

My take is that we are merging a bunch of domains together here in the one conversation. AI in the discovery layer (citizen trying to understand do they need to do something or need something but don't know where to go) will mean content will need to be structured in a way that can be pointed at and in a way that the Algorithms can join up the contexts of that content. I see this is where the methods we use to create better rules have a way of training these AI systems. This is different from the Transactional or Core systems side.

2018-09-27

Technically it's Machine learning and not AI.. But yes.
Has anyone spent some time classifying the government data into these two areas.

2018-10-02

Anyone else familier with the OpenFisca implementation in NZ. I'm looking at:
https://github.com/ServiceInnovationLab/openfisca-aotearoa/commit/1c7406479878c6e4489b84c0e1a629ab351353e3
and I am also looking at the natural language expression of this compound rule:
https://www.ird.govt.nz/wff-tax-credits/understanding/all-about/iwtc/in-work-tax-credit.html
My questions are:

1. From what source(s) does the data come from for the several data requirements here? For example when we see: "family_scheme__in_work_tax_credit_income_under_threshold: true" ... from what source exactly would that "true" be obtained? (And how would the rule itself validate that it's getting that data element from the correct source?)
2. That github link is for some tests. Can you point to the rule (or rule set) expression alone, in computational form, that we can review in comparison with the natural language text?
3. Looking at the section: "If you work the required hours, and receive qualifying income, you are still eligible if your income includes the following". Can someone please point to the actual section of the appropriate legislation that makes these provisions? This is an interesting section because it provides pointers to other rules. Oddly the webpage is not supplied with links to them, however.

Thanks.

2018-10-02

Hi, thanks for asking and showing an interest.
TLDR: time constraints and external requirements meant this variable exists without a formula until the time/need exists to code it. Some variables represent rather long and convoluted rabbit holes.

Long answer:
The project to code NZ legislation is rather large (ahem). The openfisca-aotearoa project has been currently developed with an eye on the future as much as we can but the primary driver to date has been for it's use in specific online services to demonstrate and explore what this whole field might look like.
That specific variable you're asking about was required for work being done on the http://smartstart.services.govt.nz/ website. The site is being ported from another rules engine. The website application was structured in a way they just needed a boolean value which they would set using logic within the website.
Due to time pressures and the added complication that adding the formula would create (we'd then need to create new variables for hours worked etc.) we use this technique to side step that "rabbit hole", passing it (the problem) to the person setting the value.

There are quite a number of incomplete formula (or missing formula) especially related to the work done for the smartstart website, it's a pragmatic way to isolate the particular aspect of legislation being worked on.
If you have some software experience you'll also realise this creates breaking changes with future releases. Because legislation is both constantly changing and there is no chance right now of completely deploying something like the Social Security Act with a part time team; so deploying instances tied to particular services is the current approach.

There's also another complicating factor in this service delivery space, policy and implementation of services don't often match directly to legislation. This can result in questions needing to be asked that have nothing to do with legislation directly or implementations that contradict the legislation. There's been some talk of a policy/fudging layer that allows the legislation as code project to stay "pure" as well as bringing more transparency to these situations. Discovering these sorts of implementation issues is very much part of the reasons for this particular exploration.

2018-10-02

Two thoughts...

1. What I sense from your comments is the need for a conversation about choosing a design strategy for 'whole systems' (composability). Automating legislation top-down has 'rabbitholes'. Automating legislation bottom-up is like agent-based-modeling -- each discrete rule is like an autonomous agent that responds to data. Please take a look at the "advantages" and "disadvantages" of control tables:
https://en.wikipedia.org/wiki/Control_table#Advantages
https://en.wikipedia.org/wiki/Control_table#Disadvantages
The "External Links" section of that page includes "Table-Driven Design" by Wayne Cunneyworth of DataKinetics (DKL). Although that document is dated and, in their particular case, was mainframe-oriented, nevertheless in collaboration with the DKL founder, we've implemented a somewhat similar approach and adapted it to current generation distributed computing platforms.
2. To what extent has RuleML been employed in the NZ work? Can some examples be posted?

2019-04-24

About "If you have some software experience you'll also realise this creates breaking changes with future releases.", are you referring to incomplete/missing formulas in general (which seem essential to me), or to the inclusion of variables related to the smartstart website? I think you meant the latter, but wanted to be sure.

2018-10-03

Let me illustrate my comment: "Automating legislation top-down has 'rabbitholes'. Automating legislation bottom-up is like agent-based-modeling".

While working on an example for how to automate a collective agreement (e.g. union + government) here, most were talking about the enormity of the problem, saying that amongst all the collective agreements and related legislation there are 80,000 rules to code. Well, Xalgorithms started right at the very bottom of the rabbithole: "base pay". We quickly discovered that the primary reference for base pay that's appended to the collective agreements themselves, posted right on the Treasury Board of Canada website, have choose one]: (a) undocumented rules; (b) errors. So, we focus on getting base pay documented in both natural language and computable form. (Getting those errors & omissions corrected, how many headaches does that solve "up" the rabbithole?!!)

2018-10-03

It would be helpful to make a distinction between new legislation and existing legislation. For new legislation it should be about developing legislation top-down that can be automated. This is the type of co-design that requires policy designers to work with rules analyst, service designers, coders and customers. NZ Inland Revenue has experience with this. It avoids the rabbit-holes and other funny things you find in existing legislation. A bottom-up approach is practical for automating existing legislation. However, it is important to get the "owners" of the policy/regulations involved. At the end they represent the single source of the truth. In particular when you have to deal with undocumented rules and errors. As Joseph mentioned, this is an opportunity for people "up" the rabbit hole , ie policy/legislation designers. One of the other threads mentioned algorithmic transparency and connecting the "top" and "bottom" is a great opportunity.

2018-10-03

I hear what you're saying and think we also need to have a discussion about whether we should be "fixing" current legislation definitions. I don't think it's even an option for existing legislation so from my perspective this openfisca project has been to learn about and explore these sorts of questions. If the legislation is unclear, or apparently illogical - how can legislation as code work alongside that? If legislation as code removed ambiguity in future legislation what would be the impacts? I believe we can certainly improve the processes around how we write new legislation but to get there it has to be very much all about "the people" and experimenting with legislation as code currently is to expand that discussion, introduce people to these questions and develop all our thinking.

2018-10-03

RE: "a discussion about whether we should be "fixing" current legislation definitions"
Not in the way you phrase it. Rather I'd say "filing bug reports" to parliamentarians though. And "providing workarounds" to logical errors and omissions in legislation and regulations. Certainly the optimal scenario is that the 'owner' of a rule takes responsibility to provide its computable expression. It seems literally irresponsible (as in choosing to not take responsibility) to leave a random assortment of developers in other organizations to write the code that implements their rule.

2018-10-03

BTW, here is a typical policy in which a government authority offloads responsibility to the external software developers: https://www.revenuquebec.ca/en/partners/authorized-products/authorized-software/responsibilities-of-software-users-and-developers/

2019-04-24

I'm surprised to learn that there is an explicit policy suggesting that a user of tax software is not responsible for errors caused by bugs in it.

2018-10-03

Should we even be writing code for rules anymore anyway? If we have a look at DMN as an example, an open standard for decision modelling, we can express a decision as a set of rules. The implementation of the rules is then in a modelling notation that is easier for non-IT to understand and update. "Coders" are then left to implement against the DMN standard, with existing tests that prove that it works rather than interpreting rules.

2018-10-03

RE: "Should we even be writing code for rules anymore anyway?"

Well, DMN is code via graphics + text. So "yes". But more specifically:

1. Including the graphical aspect increases the computing and training requirements for implementers.
2. "Decision process rules" benefit from expression in something like DMN. But purely "deterministic rules" don't require a "decision" and thus don't need all the computing weight that decisions require. Deterministic rules can run with just data matching (i.e. map-reduce). Scalability, latency and ubiquity matter a great deal when rules need to be located and run on every transaction.
3. When the parties to a transaction do not know about rule XYZ, how exactly will the correct DMN expression of rule XYZ be "found" by a transaction data package in data fabric of ten million DMN rules?
4. Can you or someone provide a DMN expression of this a+b=c example in a format that any platform can run, or that can be automatically transcribed into any programming language to run natively on any platform? https://github.com/Xalgorithms/general-examples/tree/master/org.xalgorithms.examples.a_plus_b

2018-10-04

Hey. You make some good points, and that there aren't any quick fixes. Thinking about some of the points that you bring up directly:

1. Yes adding graphical aspects does increase the computing required, but the extra power required to run the design time components is irrelevant compared to other costs involved. They can be run on todays end user devices and through web browsers. Yes more training would be a good idea, but then this is true of all standards such like UML. An important aspect that you touch on here is the implementers. The

implementers in this case are no longer developers / software engineers, or business rule execution specialists.

2. I think that this is an interesting point. Certainly a deterministic rule could be expressed in DMN but as you say it may not be the most efficient way to execute this at the moment. I haven't done a complete market scan of all the DMN execution engines, and I haven't benchmarked large scale test cases either. DMN as a standard is quite new, and the implementations won't have had anywhere near the time spent on refining performance that traditional business rule execution or the performance of embedded code has.
3. Trying to a find rule within a large rule set is a different problem than how to represent a rule so that it can be interpreted by both humans and machines.
4. DMN has something they call FEEL that can be used for simple arithmetic, and some not so simple calculations. https://docs.oracle.com/en/cloud/paas/process-cloud/user/grammar-rules.html

2018-10-04

1(a) RE: "but then this is true of all standards such like UML"
Hmm, I led the initial effort to get UML picked up throughout the Canadian government. I like it a lot. But getting others to also like it a lot is worse than pulling teeth -- it's like implanting teeth.

1(b) RE: "the extra power required to run the design time components is irrelevant compared to other costs involved"
I disagree. The methods used for automation of legislation better have intrinsic scalability, or else success will get you into a really bad situation later on. Keep in mind, some poor sucker will be standing in a crowded shop waiting for the transaction to complete correctly with all the required rules. Maybe 5 seconds is okay, 2 is better, but not 30 seconds.

RE: Trying to a find rule within a large rule set is a different problem
Sure, but it's still got to bet solved. What the IoR team decided to do was make each rule fetchable from a data fabric using its computable expression. We deliberately merged the two problems so that finding and computing the rule are accomplished in a single operation.

I hope somebody will show a genuine sample from a NZ (or other jurisdiction's) law or reg, using DMN. I think it's quite hard to do, that's why we went to the trouble of creating the Xalgo spec to simplify the job.

2018-10-19

“Having a discussion about how much we should be fixing current legislation definitions” is something we can do on our own. “filing bug reports to parliamentarians” has a massive dependency to parliament. I've been there (French government cabinets in charge of writing law & Parliament & Senate) and tried to add tooling for feedback loops. It is a massive undertaking on its own, and I believe we would be wise not to put this on legislation as code critical path ;)

In this discussion as in the previous ones (top-down vs bottom-up, where to start, focusing on service delivery vs “pure” modelling, writing code or expressing abstract rules), I feel there is a strong discrepancy in how much we factor change management aspects in the strategy or not.

I'm certain most of the approach I've witnessed in France, Catalonia and NZ (and have thus embedded in OpenFisca) is suboptimal, is missing feedback loops, is putting developers in situations of responsibility when they shouldn't. But at least I've seen it work in contexts where there were no massive budgets nor contractors available to “do legislation as code” ;)

I have not seen digital transformation succeed through other strategies than combined delivery and political will, and I am pretty certain legislation as code is within the digital transformation area. This boils down to the question: why do we want to model? In my opinion, it is to deliver value to citizens, not for the sheer joy of modelling. If this is indeed the aim, then I am ready to go for suboptimal processes until the case is made clearly enough that we can improve them, rather than wait until we have proper processes to support the initiative :)

2018-10-19

RE: “massive undertaking on its own, and I believe we would be wise not to put this on legislation as code critical path"

What's required is a methodology, an agent-based self-organizing incentive structure, and time.

RE: "political will"

What does that mean in practice? I consider that a "nice to have" but not a "need to have". Political neutrality is good enough. Overcoming political will-not can be complicated (Gotta pay attention to: "What is democracy?) but can be addressed through interest-based negotiation.

2018-10-19

Since the whole approach is still pretty much uncharted territory to us in Finland we're probably allowed to make our own mistakes :-) regarding the methodology we do believe in an approach where the legislation is drafted with "semantical annotations", meaning that we consistently use linked data references to a commonly agreed terminology (or controlled vocabulary, if you wish). The terminology in questions also serves as a foundation for a common "data vocabulary" (or data model repository) which then serves as a base for creating interoperability across governmental data vaults/systems, which therefore are left basically untouched - it's just the data exchange layer that is being harmonized (as in US NIEM approach). But yes, just a theory so far - we have the methods and tools to create the "semantic layer" but still lack the backup from the Prime Ministers Office to force agencies into this setup. A new law regarding... "data management" would probably be the correct translation, is going to be introduces in 2019, hopefully that will change things at least a bit (the draft supports compulsory APIs and "semantically based metadata descriptions" extending to the whole public sector)

2018-10-19

Should probably add that I'm currently assisting the Ministry of Environement in their efforts regarding harmonization of data (descriptions) concerning the "built environment" -domain. Including it all - land use, planning, building permits, construction, building maintenance... you name it. New upper level legislation governing the whole planning section due to be renewed in 2022-23 = NOW is the time to start thinking about what are the concepts (and terms representing these concepts) that we want the paragraphs in the still-to-be-drafted law to make use of...

2018-11-21

NZ has created a rules explorer so you can see and explore all the variables of the rules we have coded: http://www.rules.nz/

2019-04-24

Is this an ok place for questions about it? I'm wondering what the teams motivation for having (all?) the variables have default values is.

2018-11-21

Oh, this is really useful thanks. A couple of questions for you:

1. I don't see a start-date, end-date, jursdiction-id, or rule-version. In OpenFisca how is that data (and other metadata) maintained in association with each rule? (An example I am looking at
is: http://www.rules.nz/variables/student_allowance__eligible_for_basic_grant)
2. Can you confirm that some of these links are placeholder rule names? When I click on some I don't get any details.
3. Is there a way in OpenFisca for a user (or a user-agent app) to discover a rule that they didn't know about?
Thanks,

2018-11-21

Hi, dates are baked into OpenFisca, so you supply the date of first enactment, then if it changes you can add to the code the new date and the change, this allows you to build a historical record. The compute engine takes care of the transition over dates.
The meta data is stored along side the variables in the code (and delivered via an http based API)
With regards to your second question - I've done some work with the team and this work is focused on underpinning service design projects - often it's been deadline orientated and the need for function has outstripped the desire of the team to fully annotate; I think it'd be safe to say the desire is to learn-from-doing at this point.
With respects to your third question - I'm not sure I understand - the rules are generally enacted around a specific piece of legislation. I previously put together another view of the same project that displays the data in a relational visual context - it allows the viewer to drill in on act's and sections and see what has been
coded: https://kumu.io/hamishfraser/openfisca-aotearoa (one would have to compare that to the linked legislation to see what is missing)

2018-11-21

Thanks for those details.

RE: "Is there a way in OpenFisca for a user (or a user-agent app) to discover a rule that they didn't know about?" >> "not sure I understand"

Suppose a new tax exemption is enacted for NZ student purchases of computers and computer peripherals. A NZ student is purchasing a laptop through www.noelleeming.co.nz. Suppose the student was not aware of the exemption. How exactly does the tax exemption rule get from OpenFisca to run in this particular transaction? And then suppose I'm visiting NZ, and I purchase a laptop through www.noelleeming.co.nz. I'm not a NZ student. Given the way you describe in the first case, how exactly does the tax exemption rule not run in my particular transaction? By what technical and administrative arrangements would the system running at www.noelleeming.co.nz exchange data with OpenFisca?

RE: "dates are baked into OpenFisca, so you supply the date of first enactment, then if it changes you can add to the code the new date and the change, this allows you to build a historical record"

What's the connection between a rule package and it's metadata? A rule-id? Can a person readily learn rule-id's of particular rules? (Say, to diagnose an issue). Would it ever be possible to migrate from OpenFisca to some other platform?

2018-11-23

Suppose a new tax exemption is enacted for …]

I feel like there is a confusion between making rules available as code and making use of these rules in user-facing services.
This is akin to open data. What you see on rules.nz is a list of available rules for computation just like you could see a list of available datasets for reuse on an open data portal. Independent reusers can then embed these datasets (resp. rules) in their code, but it is not the provider's responsibility to.
So in the example you gave, it would be up to noelleming.co.nz to maybe add a checkbox "I'm an NZ student" at checkout, or maybe store that information in the user profile, or any other implementation they would find appropriate. With that input, they could send a request to an OpenFisca instance to compute the consequent tax exemption and display it in their UI in the way they most see fit.

2018-11-24

RE: With that input, they could send a request to an OpenFisca instance to compute the consequent tax exemption and display it in their UI in the way they most see fit.
Okay. Forgive me for jumping into nitty-gritty detail, but it matters for deployment and we've been working through such data flow. What does the data package actually look like that an external system would send to

OpenFisca. Let's say there are four NZ students: two are NZ citizens, one of them aboriginal; the other is a landed immigrant, and the other is a foreign student. Are all these distinctions going to be handled via the applications (like noelleming.co.nz) or via their identity with the government? If the latter, then the "I'm an NZ student" checkbox would be redundant, since the required information would already be bundled with their identity.

2018-11-26

this is where we are getting into the difference between a simple rule and the complexities of the scenarios that it applies in or not. We would call this decision modelling. If it was a Rules based API engine perhaps you would just send the data points and an answer would be provided. To do this I would think you would have had to construct "products" or sets of rules that need to be grouped together to provide an answer. If it wasn't a transaction and was more a query say via AI this
API approach wouldn't work. You would need to be training the Expert system somehow with the appropriate Word model or decision model.

2018-11-26

RE: "the difference between a simple rule and the complexities of the scenarios that it applies in or not"

How to the digitized rules "in" OpenFisca get "out" into the world? What is the method of transmission -- mainly does event data get pushed to OpenFisca for rule processing centrally? Or do user agents applications pull rules from OpenFisca for processing decentralized? Or some combination?

2018-11-27

What does the data package actually look like that an external system would send to OpenFisca

Dear @josephpotvin, you will find all answers on the “nitty-gritty details” on https://openfisca.org/doc/. For the data structure of the OpenFisca API, the interactive documentation is available
here: https://demo.openfisca.org/legislation/swagger

2018-11-27

How to the digitized rules "in" OpenFisca get "out" into the world?

OpenFisca provides a web API, as well as a Python API. It is up to implementers to interact with it. There can be an authoritative, centralised server operated by government as well as multiple local instances. The software is independent from its operation.

2018-11-26

So I wanted to share a link from the good ole United States (my home town) about the DC Code system which allowed this civic hacker to modify the legislation via GitHub Pull

Request - which isn't all that dissimilar of a system that one of the teams built in the Better Rules hackathon in Wellington.

Very cool read: https://arstechnica.com/tech-policy/2018/11/how-i-changed-the-law-with-a-github-pull-request/2018-11-27

2018-11-27

Interesting, thanks for sharing. I am not sure what the feedback loop would look like for something different than a typo though ;)

2018-11-28

good morning or evening to all.... hope you are doing well. this topic is interesting. I do not know if I am on the same understanding. making rules into code or codification of legislation aims at making laws accessible to users or most of them. in my country Rwanda , the first attempt dated on 1970s, second after the genocide 96 and 2006 … the first editions were in hardcopy/printed out , and in 2006, we put almost our laws in electronic way /website where users can find easily the legislation that was a very good experience... now the website is down due to various reasons (updating issue, hosting, i would infrastructure …). what w ehave to consider in making rules into code on my view: make sure there is an inventory of laws available, separate laws in force and obsolete or repealed legislation, classification in areas of law , and consolidation process ...

2018-12-03

RE: "make sure there is an inventory of laws available, separate laws in force and obsolete or repealed legislation, classification in areas of law , and consolidation process"

Hi. You point to a very important aspect of digital rules management: versioning. For our part, we approach by integrating Xalgo-Author (our rules authoring IDE) with a git (GitHub, GitLab, or any other git variant, or Mercurial, or some other dedicated version management system). From that repo the "in effect" version gets reproduced on INTERLIBR (running MongoDB currently). But all previous versions are just as available (eg needed for reversing transactions, auditing, etc.) and also, proposed (future) versions could be available, for testing and analysis reasons.

BTW, a Masters student in Spain just shared an article about the potential use of an IoR (and blockchain technology) in the contect of economic development in Sudan: See https://xalgorithms.org/news-articles/

Would be happy to collaborate.

2018-12-03

thank you very much for your reaction and advice. I was afraid that my opinion was out of the context.
I will try to understand your advice with our IT service.
thanks again.

2018-12-02

Hello all, We have just posted online a short video that illustrates line-by-line the authoring of a fiscal rule in 'Xalgo'. The sample rule is a Value Added Tax base rate. https://www.youtube.com/watch?v=pcoWgOYvbbk Also we have posted an overview 'brochure' style document that outlines the functional elements now in place (at 'alpha' level) to create "an Internet of Rules": https://xalgorithms.org/wp-content/uploads/Xalgo-_Brochure_Final_SCREEN_2018-11-30b.pdf This is not a software application to download and install micro-level; it's a true meso-level networked process that anyone can freely access. The user-to-network relationship is illustrated at the bottom of the 4th page of the brochure. The working parts are each explained on the 3rd page. The video explains what happens after a Xalgo-fact message is sent by some commerce application (optionally by LICHEN, but it could be any application that generates a well-formed Xalgo-fact message). The Xalgo-fact message is received by INTERLIBR (online) and which uses Map-Filter-Reduce (the original Google search engine process) to find all applicable Xalgo-rule messages. The video illustrates the line-by-line authoring of a Xalgo-rule, testing the result of each line to see what it does. This is only for the base rate. For a zero-rated item or buyerID, that's just another discrete rule that would get 'found'. Same for exemptions, credits etc. A rule package can be create using lookup (for parallel) or chaining (for series) calculations. We look forward to any feedback or questions

2018-12-24

Hi all,
I am quite new to the topic of legal drafting so I'd like to ask any legal drafter here who has the time to walk me through the "traditional" process of it, maybe with a short but specific example

2019-01-05

How many of legal drafters you know have coding experience?
Are legal drafters interested in learning to write code in any programming language?

2019-01-05

I was investigating this topic, and he had taught me a class in legislative drafting in law school, and he was generous enough to share his time. I described some of the technology that I was researching and what I thought it could be used for, and asked him what technology they use in his office, and what technology they would like to have.

He said a few important things that I remember very clearly:

1. Ambiguity is perceived as a tool. Precision in legislative drafting is achieved at the cost of brevity. Brevity is a key to clarity. So legislative drafters USE ambiguity in order to achieve brevity, and increase the degree to which the law can be understood for the 95% of cases in which it is not ambiguous.
2. Budgets are non-existent. This is a department whose importance is not well-understood by its clients. There is no R&D budget.
3. Sovereign governments do not perceive risk in poorly-drafted legislation. Why? They can retroactively re-legislate to fix real problems. So the downsides of making mistakes in drafting legislation and

regulation are not show-stoppers, which limits the motivation to use technology designed to avoid errors.

4. What technology do legislative drafters want? A Microsoft Word plug-in that can do amendment section numbering properly, and then automatically re-number on the next revision of the statute books.

I.E. If an act has 5 sections, and they want to add a section between 2 and 3, in order to maintain consistency between revisions of the statutes, the new section will be 2.1. When the next revision comes along, concordance tables are generated and 2.1 becomes 3, with all the rest of the sections renumbered. In Alberta, at least, those numbering processes are manual.

Let that sink in.

My take-away is that until legislative drafters conceive that human beings are not their audience, or not their ONLY audience, they will not be motivated to adopt structured drafting methods. They are extremely cost-sensitive, and like most people what they want is not to be able to do a thing that they don't know exists, or is possible or useful, but for the most annoying parts of their day-to-day lives to be taken away.

2019-01-05

Thank you for sharing this! It is quite depressing to be honest.
Still, I hope that there are some open minded drafters around who would help in the development of a free and open source solution by sharing their experiences.

2019-01-07

That's a really thoughtful post that challenges a number of my prior assumptions. So thank you for sharing.

I do have a contrasting perspective based on my past 4 years working at a New Zealand regulator, regarding points 1 and 3.

(1) Our policy and legal teams do not view ambiguity as desired in any regulatory area. Less ambiguity means that regulated parties have clarity on what they are allowed or not allowed to do... improved compliance and less burden on the regulator too. An ideal regulation to them would already incorporate rules you could express in pseudocode, eg IF (fishing_area==central) and (fish_type==tarakihi) and (fish_length < 25cm) THEN Must_Throw_Fish_Back.

(3) Also those same teams do spend quite a bit thinking about quality upfront, and I have never heard any one of them say "we can always fix it later". Revising legislation is often necessary based on changing global standards or new government priorities, of course. But revisions that should have been caught before first passage cause agency reputational damage amongst regulated parties, to say nothing of Ministers. It is precisely this situation that is driving our agency's interest in machine-executable rules.

2019-01-08

Valuable feedback, thanks.

2019-01-08

Dear, what are your thoughts about legal drafters and coding experience? Do you know anyone who could enlighten me about the current process of drafting statutes to bring ideas about code and the current actual reality closer together?

2019-01-08

These comments highlight the opportunities and why this work is happening. Also not sure if this is the prevailing culture and practice everywhere. It is worth exploring the concept of "ambiguity" a bit more. It seems "ambiguity" as a tool evolved by necessity and not by choice nor design. By "necessity" I mean constraints put on the legislative drafting environment. With (technological) changes happening in our society these constraints may go away or alter and change the way we look at "ambiguity" as a tool. Generally, citizens or businesses impacted by regulation are generally not well served by ambiguity. It makes dealing with government more expensive, more frustrating and reduces trust and confidence.

2019-01-17

Absolutely agree about ambiguity. We try to eliminate it in all forms. What may be present in statutes, and occasionally regulations, is more akin to broad, open-ended (I won't say "vague", as that's not an acceptable drafting outcome either) provisions -- drafted that way usually because the policy surrounding how to implement the provision is not clear or hasn't been well defined. And in some cases the client department would prefer to "fill-in" that vagueness with policy, although this is also strongly discouraged.
As for brevity being valued more than precision -- also not so here. Clear, unambiguous language often ends up being lengthier.
There's always a balance to be drawn in achieving all the outcomes, but the end result has to be clear and legally effective

2019-01-06

Hi all - I was recently contacted about the World Legal Summit, which I thought may be of interest to members of this group. You can find out more here: https://worldlegalsummit.org/ and if you would like to contact Aileen directly, her email address is a.schultz@worldlegalsummit.org. She has asked me to share it with this group and to share her details. I believe she would like to find a host in New Zealand (not something that LawHawk can offer as a cloud business!).

2019-01-12

Have a look in the legislative implications thread here. Legislative drafting is carried out differently in different groups of countries, particularly Commonwealth, USA, EU, and civil law countries - reflecting different political and legal systems. In non-presidential Commonwealth countries civil servants help government politicians develop policy and then come to us as specialist public sector lawyers to work out how to give legal effect to that policy (where legal effect is needed) in the given existing legal framework. We don't deliberately mess it up, but fitting statutory rules into a common law system is difficult. It is important to remember that old legislation can be badly drafted in ways that don't reflect how we do it today. Let me have any more specific questions and I will try to answer.

2019-01-18

Hi everyone, I have been reading the posts about legislative drafting. The Australian Govt ran courses on legislative drafting and I assume they still do. Other Govts probably do so. I wonder if it would be useful to see if people working on codification of legislation can attend these courses or if they can run a course especially for a group of codification people. Also is there a standing permanent 'Legislative Drafting Advisory Group' as part of this codification initiative. It would help answer some of these drafting questions but also would give insights as to how they think, how best to get drafting people on board with codification and enable them to actively give assistance. It would take some initial set-up and settling in but could really be constructive and supportive.

2019-01-18

Hi, you might want to look at the "Legislative implications" thread. In Commonwealth countries most legislation is drafted by specialist lawyers in government or parliament. There are various training courses, used more in some countries than others, plus in-house training. It is generally accepted that it takes a lawyer several years of drafting experience before they can be given full responsibility for their own draft. Our professional body is the Commonwealth Association of Legislative Counsel www.calc.ngo . I am trying to get members in different countries interested in this and it will be discussed at our conference in Zambia in April. Meanwhile we are encouraging drafters to join this forum.
By the way "codification", to a legislative drafter, means something else - it is what the Romans and Napoleon did, pulling all the laws (including case-law) together and making them into one rationalised "code" (like the French Code Civil or the USA Codes), in natural language rather than computer code. That has not got very far in countries that favour the "common law" approach, and there is a lot to be learned from the history of attempts to rationalise laws - personally I think the digital angle is what is going to make the difference in this attempt.
I think drafters would agree that it would be useful to explain to digital people what we do, just as it would be useful to organise something the other way around. I know Richard Wallace from NZ drafting office is keen not to have groups splintering off, and for everyone to come onto this forum, so it might be worth raising your idea with him too. I am due to meet Pia Andrews in NSW on Monday, and will ask her what she thinks too.
Looking forward to progress in all the different professional groups (including govt policy staff) making themselves better understood to each other.

2019-04-18

Hey all, I'm working through the case studies in the Better Rules report. I'm looking for report on SmartStatute.net (which appears to be offline at the moment). Does anyone know (perhaps?) how I might get my hands on this paper/presentation? https://goto.geek.nz/2019/04/computer-language-model-for-digitising-nz-statute-law-wellington-16-april-2019/

2019-04-18

I have sent you a Twitter direct message with email - despite presenting with him at the CALC conference, I now realise that is still the only contact I have for him. My guess would be that his www.smartstatute.net is offline because he has moved on so much since then. I will email him now to encourage him to get in touch with you.

2019-04-18

One nice thing about mechanising (using NLP techniques*) the semantic extraction of rules from legislation is that if you can automate it you can debug it.

http://aura.abdn.ac.uk/bitstream/handle/2164/9895/FAIA302_0101.pdf?sequence=1&isAllowed=y

*When I talk about NLP techniques I mean hard parsing, with tools like GF, rather than the softer, statistical, ML-oriented type bag-of-words kinds of approaches that are currently in fashion. One of GF's catchphrases "don't guess if you can know."

2019-04-18

We are hoping to have 2 of these authors -(who is another drafter) - at our UK-&-nearby drafters' mini-conf in London in late June, but they are also involved in the "Legislation & Regulation on the Semantic Web" workshop in Montreal on 17 June (the London drafting office will be talking to Adam either way).
I met one of the drafters from Singapore at the CALC conference - are you in touch with them, and is Singapore taking part in the "show & tell" video event planned for mid-May?
The "Rules as Code" idea really seems to be starting to pick up momentum around the world now

2019-04-18

Who from Singapore was at CALC? My co-founder I just moved back to SG after some time away so we're just getting plugged back in to the local scene now. I do plan to show-and-tell some of our (Legalese's) progress in May.

2019-04-19

Haven't got details on me, but will get back to you later this weekend. Singapore has interested me since I was pointed to Sept 2018 CALC newsletter https://www.calc.ngo/publications/newsletters (Chief Legislative Counsel, Singapore Attorney-General's Chambers) continued the theme by explaining how Singapore is adapting its legislation to deal with the challenges of automated decision-making. noted that under legislation, power to make a decision is invariably given to a person, and that person is ultimately accountable and responsible for the decision. She indicated that there may be a need for explicit authorisation for computer-made decisions, and that Singapore may amend its interpretation legislation to authorise computer-made decisions. Automated decision-making has the advantages of improving efficiency and certainty and being available around the clock, but to be accurate it requires adequate data sharing within government. It can also be less transparent in that legislation is available to all citizens, but when it is translated into computer code for automated decision-making, the code is not. outlined how Singapore has responded to this challenge in relation to electronic voting, by taking the unusual step of legislating for the scrutiny of the computer code involved before it can be used."

2019-04-19

I can see how some straightforward decisions could be made by computer, but how would you incorporate the multitude of policy documents that civil servants use to guide them in making decisions under statutes and (more often) regulations? This is usually where you will find the criteria and factors to be considered. And these very often change, depending on the interpretation/limitations/expansion of the rules that government wants

2019-04-19

Absolutely, I have asked for reassurance before, and been given it - none of the people taking this forward seriously inside govts is looking to choke off human input and discretion - it is just about automating what can be automated. The Singapore quote is interesting because I don't see what they mean - I don't see something a non-AI computer does as a decision. If the legislation says "If X and Y and Z, then the Minister must grant the licence", then the computer can check X, Y & Z and print & post the licence, but that is not really a "decision". Even without removing human discretion there is still plenty that can be automated.

2019-04-29

I had a good chat with some contacts in the States the week before Easter. They had been at an AI symposium @Stanford University. Automation in the AI space is going well in the Statistical probability world, ie Im thinking of say understanding what treatments might be useful for this particular condition. However in the Semantic domain its really miles away. This is where we have really started to think less about the Code aspect but more about a common way or model (Blueprint maybe) for how Regulations are newly created and also for understanding the existing ones. This is aimed at Humans getting on the same page

2019-05-14

Item 3.3
of https://papers.ssrn.com/sol3/papers.cfm?abstract_id=3021452 makes a similar point. I agree the semantics have to start as far upstream as possible

2019-05-16

Hi. Yep Im trying to look at this in a very simple way. We have also found that this has been something that the Policy people can adopt as part of their day to day work by just writing on the board or creating post it notes.

2019-04-23

more papers on the subject (of varying relevance to this discussion), as i discover them:
https://orbilu.uni.lu/handle/10993/39326
https://www.academia.edu/38385816/A_Legal_Validation_of_a_Formal_Representation_of_GDPR_Articles
https://www.researchgate.net/publication/317044637_Legal_Ontology_for_Open_Government_Data_Mashups2019-04-24

2019-04-24

Thanks - very helpful. Will read them properly later, but on a quick skim the pair on GDPR look to me to amount to case studies for why the "Better Rules", "Rules as Code" NZ-NSW approach is a good bet. People are going to struggle for some time to develop AI that can read & encode EU legislation, or to get systems for translating legislation into code after it is enacted - but RaC starts at the other end, with the idea that new legislation should be drafted with a coded version alongside it from the start, to nail these translation problems as part of developing the policy, legislation and coding together. It may be sensible to have longer term goals, but it is also good to learn to walk before you run - RaC not only avoids having to work out how to get AI to do the job, but also means you can start with co-working on turning policy into legislation that is already being drafted to standards that make it easier to extract logical rules. EU legislation is notoriously poorly drafted (by committees, with horse-trading over wording, across several languages, with minimal legal input, attempting to straddle different legal systems in MSs), and data protection legislation is even more notorious for being unfathomable - GDPR combines the two, and so is one of the pinnacles of incomprehensible drafting in modern times. In most of the Commonwealth legislation is now drafted by trained specialist lawyers, who take vague policy and insist on making it logically workable - asking many of the same questions as are asked by coders looking at older/EU legislation. So drafters in Commonwealth countries will be able to identify drafting projects that make more sense than trying to start with GDPR.

2019-04-24

I think you should've started with your last sentence!

LINKING CONCEPTS WITHIN AND ACROSS RULE SETS

2018-09-20

What are the impacts of having similar concepts across rule sets that have different definitions? E.g. 'income' is used in lots of pieces of legislation, and can have different definitions depending on the context and requirements of the particular legislation.
Could/should this be managed?
What are the options?

2018-09-21

This is something that we prioritize in our national efforts towards "machine consumable legislation" aka better rules (as Finns we tend to focus on technology first :-D). We're about to launch a pilot project which will focus on creating an editing tool for lawmakers that will enable them to make use of (link to) specific concepts in a controlled vocabulary, thereby diminishing the possibility of "inventing" your own terms in situations where you should adhere to commonly agreed concepts. Updates will follow! (attached another description of the same project, targeted at students at a local technical university - these will run in parallel > btw, did quote your splendid report, thanks for that!!!)

Aalto_CS_Machine_Consumable_Legislation.pdf

2018-09-22

This seems like it is regularly done already in my jurisdiction, where definitions in different pieces of legislation are referred to rather than repeated. There are also generic interpretation acts which serve as a sort of central definition repository, and apply by default to everything. The drafting tools must have the ability to facilitate that. But this can be extended beyond definitions as we understand it. Any legal conclusion that can be reached about a thing can become a definition that can be referred to elsewhere, such as "a person referred to in section 123 of the Something Act". The ability to make those sorts of references to un-named legal conclusions, not merely defined terms, and to be able to do the same sort of tracking of those inferences at amendment time, is also valuable, I would expect

> *2018-09-25*
>
> Indeed. Even basic text references can suffice to guide later implementation and reuse across a model. Example.

2018-09-24

You make an important point here. As you note, there are various ways in which definitions are standardised across legislation. There has always been value in doing this (ie, ease for readers when a particular term has a consistent meaning). What has changed, is the use to which a term can be put. With technology and the use of data, Government can accumulate information and allow you to use that information to make things simpler and more effective for you.

For example, a range of Government benefits may require you to provide your date of birth, your income, and how many dependent children you have. We can establish systems that utilise that information for all purposes. So, once you have provided that information once, you don't need to provide it again (unless the information changes - so, you may still need to verify that the information is correct). This is the concept behind "ask only once".

To do this effectively, you need to be asking the same questions, or seeking the same information. For example, this system doesn't work if every piece of legislation has a different definition of income or dependent child. Accordingly, there is an increased value now in using the same terms across legislation - it can drive functionality that didn't use to exist.

The trick is going to be ensuring that people see the value in this and so choose to use existing standardised terms, rather than viewing each situation as one that requires bespoke terms.

2018-09-24

A fool with a tool is still etc. but in our pilot "tool focused" project we strive to make it visible to everyone drafting legislation that you have to be very careful with your terminology and not hide behide the notion that you are drafting something so unique that you definitely need "bespoke terms" to express the rules in your own domain.

In my opinion it's part laziness, part the fact that comparing legislation (and its terms) from different areas in society is extremely difficult. In our national context laws are usually written with the standard "disclaimer-phrase": "by some term] in this law we mean once again something a bit different than legislative drafters in another department/agency]". That would perhaps be OK even in the future, but this "bespoke term" chosen by the drafter would at least be linked to a common vocabulary (with URI:s) = if someone calls an apple a "round pear" and someone else uses the term "greenfruit" they could both link their preferred term to a common vocabulary entry "apple"

2018-09-25

I believe ambiguity should be handled.
It is part of the difficulties for people impacted by law to fully understand how they are: your common understanding of “income” might be different than this agency your applying to.
It also yields inconsistencies in public service delivery, as different agencies (or agents) might have different understandings of the same terms.
Finally, it obviously hinders the ability to reuse concepts in legislation as code and yields to duplicate work. We know the OpenFisca France model has several duplicates because of concurrent implementations by different agents that did not know which vocabulary to use and preferred to create another concept they fully understood rather than risking to change the meaning of a preexisting one.

Since this is a classical ontology-building problem, we can learn from knowledge engineering and semantic web concepts.
We know that it is unrealistic to require investigation of the whole knowledge base before implementing a new concept, as the rate of growth would linearly (probably quadratically if you consider human cognitive ability) decrease with the knowledge base size.
The most efficient way is probably to regularly review concepts and lead discussions between implementers for all the ones that are possible synonyms.
The most effective way is certainly to allow interlinking and external, post-factum equivalence markup. This helps decrease refactor costs.

I demonstrated the possibility to align ontologies post-factum
there: https://github.com/MattiSG/alignements-loi
You just create sets of descriptors, each of these descriptors being referenceable as an URI. Lookup is dead simple: use one term ("income"), specify its domain ("tax law"), have software transform that into an URI, look up across descriptor sets for a match, return all other descriptors in that set, dereference them (wikipedia page for income in the given country, OpenFisca representation, official webpage of the minister defining income, definition in legalese…).

2018-09-25

Concepts cant be considered just by themselves. The context in which it is used is where the complexity starts to arise. This is where the Decision modelling starts to drive its value. Iterating between concepts and the context will flush out where income for example means the same thing or is different. There are tools out there already such as Rules Express that enable the ability to link Concepts to Decisions to Rules and can be linked to a Organisations Ontology as well. However I think that the biggest progress we can make is to use a common method/approach to facilitating the conversation between agencies

2018-10-05

I think there's a lot of value in doing this, however I'm totally biased in that sense! from the European Commission, and myself have set up a group to discuss controlled vocabularies (currently just broad cross section from the UK, EC, Euro Parliament, Italy, Aus and NZ). Not necessarily in the context of legislation as code - our motivations are different - however this concept of mapping a vocabulary across a domain so that you can connect into that either for open data, for data exchange across countries (where the data exchange can be statistical data or administrative, transactional data) or for reuse more generally is something that is taking hold in government I think. X from the university of Rome is in the group also, and he and the EC have been working on the open source VOC Bench 3 to help facilitate that. The only downside I can see so far, is that I need to import or point to the other vocabularies I want to use, the system doesn't use the LOV (https://old.datahub.io/dataset/linked-open-vocabularies-lov), but it does look good for undertaking this work. I think it could connect to open fisca for example, giving you legislation as code, but with a controlled vocabulary that would let you find where the legislative terms have the same meaning. It would be important to be able to somehow add in that a term is modified by another section of an act, or disregarded when applied in certain contexts, but I'd think that would go into your code, rather than the vocab. is in the group, and we have someone from stats nz, and a handful of DIA folk as well. I've been unwell for a month or two, otherwise I'd be able to point you towards our community and the work! Unfortunately, this has also limited my engagement here. Hope to take part more often soon.

2018-10-23

This "group discussing controlled vocabularies" > can it be reached online on some forum or how do you exchange opinions?

2018-10-23

Hi, we're just starting p really (2 months in with monthly meetings). I'm coordinating with someone from Scot gov and the European Commission at the moment. It is purposefully closed, but we have a slack channel and also the Digital Transformation Agency here in Australia host a community of practice forum for us. I'll invite you to all the bits and pieces if you like. We have NZ, CA, US, AU, UK (Wales, England and Scotland reps), Italy and the EC from memory

2018-10-23

A Slack invitation would be nice, thank you!

2018-12-20

Hi, we in Finland have done a Proof of Concept in this area > http://bit.ly/SemanticLawEditor

2018-12-22

Thanks for the report. It looks very interesting.
We have done a demonstrator led by the New Zealand Lab in DIA (and their team), Uruguay and ourselves in Israel. You can get some insight into the model in the following links:
D7 legislation as code demonstrator
The demonstrator app:
https://serviceinnovationlab.github.io/Piccolo/
The video:
https://youtu.be/bglyliHUonU
We are from the eGov Unit in the Gov-IT Authority in Israel, and would be happy to join any ongoing international activity regarding legislation as code.

STANDARDS AND FRAMEWORKS FOR RULES DEVELOPMENT

2018-09-12

What standards could we be using? What value do they provide?

2018-09-22

I am aware of a few projects out there aimed at standardizing different aspects of the problem. There are OWL ontologies being developed that are supposed to allow different rule-sets to refer to common legal entities, allowing for interoperability between digitized rules. There are standards like LegalRulesML, which is a markup language for natural language rules that makes their semantics more understandable for machines for various purposes. There are also different formal logics that can be used in the systems that reason with digitized rules. Each has a different set of logical implications it relies on, and so the same set of rules would need to be encoded slightly differently depending on which you choose. For example, the tools used by Regulation as a Platform use a deontic logic, that requires that all the statements be framed as an obligation, permission, or prohibition. Other tools, like ErgoAI, does not use deontic logic, and so an "obligation" becomes a thing that you need to model in the rules explicitly. Which logic you intend the rules to be used by has big implications for what needs to be made explicit, and what can be left unsaid, before the rule will work in an encoded form.

2018-09-26

This is an area where could provide a lot of input into what is happening from a Standards view. A small list of various standards that are the basis of some of the methods that we have been using for the last 5 years in either deconstructing Policy or constructing Policy.
sSandards managed by the Object Management Group – the same group that manages the Business Process Modelling Notation standards
Semantics of Business Vocabulary and Rules (SBVR) http://www.omg.org/spec/SBVR/About-SBVR/ This is the standard that Concept Modelling is linked to.
Decision Model and Notation (DMN) http://www.omg.org/spec/DMN/
And the Rule Interchange Format is managed by the World Wide Web Consortium (W3C) – the same group that manages the HTML and other web standards. http://www.w3.org/TR/rif-overview/

2019-11-13

I have spent some time with DMN. Verdict: thumbs up. Here’s an introductory article: http://ceur-ws.org/Vol-1875/paper14.pdf -

2018-09-26

Lets also talk about the need or not for standards and what they achieve. I always try to look at other industries to understand what they have done to understand and create solutions for their problems. Also standards are never static as they evolve over time to meet new needs.
I could talk about the standards that allow the Internet to work the way it does but lets say use the Automotive industry instead. The humble wheel for example has a number of standards that allow independent manufacturers to design and build and as long as it meets

the standards for bolt pattern, width etc it will happily fit the right tyres, meet the spacing required to clear the brakes and also fit the car/truck etc.
So do we need Standards across the Legislative lifecycle to get to a Legislation as code outcome?

2018-09-26

A while back, there was much discussion of 'concept models' in these threads ... the building of robust, structured business vocabularies to support business-friendly expression of rules in natural language. I am pleased to report my new book has been released last month on the topic: "Business Knowledge Blueprints: Enabling Your Data to Speak the Language of the Business". Reviews so far have been quite positive: https://www.brsolutions.com/business-knowledge-blueprints.html Get in touch to discuss!

PROJECT IDEAS TO WIDEN THE CONVERSATION

2018-10-02

Often with new fields of work such as better rules, identifying the right questions to ask is half the battle. Undertaking key experimental projects to assist in identifying these questions can help foster a learning-by-doing process.

2018-10-02

I'd like to see a small research project that models the effects of a proposed (or historical) law change using legislation as code to both demonstrate and further inform the opportunities legislation as code can offer in this space.

2018-10-02

Over the next month I'll be leading a project whose purpose is to demonstrate and investigate the opportunities/problems/questions that legislation as code offers across international borders. Due to timeframes the focus of the project is to (using legislation as code) compare a similar piece of legislation in different countries. We're currently considering pension eligibility. I very much welcome feedback / questions and inspiration.

2018-10-02

Starting off with pension eligibility will certainly highlight differences for interjurisdictional comparison, due to special rules for people with certain types of disabilities, those from military employment, those with First Nations / Aboriginal status, and so on. So yes, if the idea is to explore opportunities/problems/questions, that's good. But for the first comparison I'd suggest something to focus on getting the comparability itself set up, before throwing a great deal of intrinsic rule complication at it from the get-go. ... Or, ya, we can jump in with both feet from the outset! :-P

2018-10-02

Dear team.
Glad to be in this team - will give as much input as I can. I did an initial design of benefits to the senior citizens in Israel after I found it was not structured, especially between different municipalities, and proposed a scheme to structure the local regulation using the same parameters . Later a more detailed study was carried out by the Minister of Social Equality which resulted in a detailed excel file.
I think the comparison should start with a definition of the basic parameters, such as;
age of senior citizen (in Israel it is 62 for a women and 67 for men)
Municipality
Different tax discounts for central and local municipality
Social security benefits related to the income of the person and his spouse.

That's for a start.

Best regards

2019-01-04

I'm working on a prototype, and I'd be interested in feedback from people in the Better Rules community. This seems like as good a place as any to post it, but let me know if there's a better forum.
<link removed for privacy reasons>

2019-01-04

Replied on Twtr https://twitter.com/jrpotvin/status/1080955080902434819 But I'll just reproduce my comment here: The user base for this GUI would be semi-technical people. Yet semi-technical people wouldn't have trouble with the same rule info expressed in ASCII text. Doesn't the GUI add risk that WYSI¬WYG? Wouldn't a rule author check that the machine readable output is exactly correct?

2019-01-04

Thanks,
So I'll expand on what I said on twitter.

First, I think that people who are comfortable coding something underestimate the intimidation threshold that has to be overcome for people who do not conceive of themselves as programmers. I think the experience in educational settings is indisputable... textual languages are not "discoverable". Consider the difference between "learning to code" and "learning to puzzle." No one ever says "learning to puzzle." Because it is obvious on sight how you are supposed to interact with a puzzle. Using Blockly turns the programming language into a discoverable thing, and has a huge impact on how willing people are to start playing with it.

So I think there is a big and important category of person who will not try it unless it is at least this easy on the eyes. There is every likelihood that someone who starts using a blockly version of a programming language will soon realize that it would be faster, and more flexible, for them to simply type out the code. But not everyone will graduate to text, and most of those who do would not go to text directly.

Accessibility is one of the main purposes of Blawx, so even if there is some sacrifice to fidelity, it would likely be worth it for that benefit.

But to your question, there is no sacrifice in fidelity. Whatever you decide to build, it will generate exactly the code it was intended to. There are two other problems. One is that in the graphical version you have to find a balance between "easy" and "powerful." There are ways of doing both, but those increase the total number of visual elements the user has to pick from, and then you have to "hide" the hard ones under an "advanced" tab, or something like that. The other is that there is not, right now, an option to change the code version and have those changes reflected in the graphical version. The translation is uni-directional, right now. Not a huge issue, but bi-directional would be better for teaching people to use the textual version.

On the plus side, the fact that bi-directional means that you can visualize a sub-set of the text language, and still do something useful.

2019-01-05

Sure, that makes sense. We'd be happy to try this as an interface option with XalgoAuthor. If the components you're referring to are all free/libre/open, we can try our examples. Pls see this VAT illustration created by UOttawa student last month: https://www.youtube.com/watch?v=pcoWgOYvbbk I'll be meeting with today, and as soon as we can download your GUI components we can try playing with your method. That way we can provide you operational feedback. If you would like us to follow some "user experience" protocol to support your formal academic research, we can do that.

2019-01-12

I am all for the idea that many non-coder lawyers will be much more likely to grasp the potential here if there is something that they can play around with like this. That also goes for non-coder policy development staff - they come up with the rules that they ask us legislative drafters to turn into legislation. As I understand what is saying in the "Legislative Implications" thread here, the key is for policy staff and drafters to share a common understanding of the rules they are developing, with the coding helping that process and growing out of it. This looks like something we can use to explore how that could work. Thanks

2019-01-16

Hey, everyone. The demo video I shared earlier is now a live site you can try. It is only a proof of concept, and it is very Alpha. But I am told that you should get something into people's hands as soon as you can, and start learning.

To be clear, the concept being proven here is that declarative logic programming for encoding laws can be made user friendly enough to allow subject matter experts to do it without requiring a programmer.

Check out <link removed for privacy reasons> .

All feedback is deeply appreciated.

2019-01-17

Hi - the website is blocked by my org's internet protocols. Any idea why tht would be?

2019-01-18

None. Can you give me any details? I know there is a problem with the configuration for blawx.com (no www). The only other idea I have is that they don't like self-signed security certificates from let'sencrypt?

2019-01-19

Hey - it was an issue with the firewall on my side. My IT have fixed now!

2019-04-01

I've been exploring the different perspectives that people have on what this legislation space involves and posted the following whiteboard image on twitter last week (https://twitter.com/verbman/status/1111352809104957440) which generated some discussion.
This is a systems hierarchy perspective.
I posted an earlier working version in the above thread which included case law but removed it in this model because I think there's a need for discussion around why we would publish a 1:1 legislation as code library if it didn't include the business rules used within departments and didn't include case law.
My perspective on this is it's essential as a public reference layer. It would be utilised internally by government departments in different ways depending on the complexity of their internal business rules. It could also be referenced by parties interested in layering case law generated rules over top of the published rules (and departments may well choose to start publishing their business rules publicly in the same way).
This adds a requirement to the systems needed for publishing legislation as code to preempt and allow for such an "inheritance" model.
Am interested in feedback on this.

Systems Map

2019-04-18

Btw regarding your hierarchy from your whiteboard – I was reminded of the CIM / PIM / PSM distinctions originating in the MDA world. Section 3 of https://drive.google.com/a/legalese.com/file/d/1XKDVeI0IFUlFdkv6dsnwzdg9XExkNL-E/view?usp=drivesdk has a useful diagram.

2019-04-26

I am somewhat stuck on this concept of the ruleset / rulebase - while it appears useful in the diagram for format changing and logic preservation outside of anyone technical stack - I am unsure if those benefits would be outweighed by the negatives/overheads with respect to legislative rules. I am also yet to find an open tech stack that we can experiment with. This lack also suggests it's a nonviable option given it's an historical business rules model that has been by and large rejected by the software development space. Is this the same problem/overlap your addressing with the DSL approach ?

2019-04-26

As for open stacks we can experiment with, I've been having some fun with Ergo Lite and the rule language and reasoner, and Docassemble as a tool to create a user interface to answer specific questions.

Just published a fun series of Medium posts demonstrating the potential of using Ergo Lite for statute reasoning by encoding 5 LSAT questions.

What do you need in the stack for experimentation purposes?

2019-04-30

The two major benefits, as I see them, are support for natural language generation and for interfacing with formal verification tools. The former allows for livecoding IDEs that show English-language (and other languages too) representations in real time. The latter allows those IDEs to warn of errors in drafting as well, for both contracts and laws. I choose to perceive the lack, to date, of such a synthesis as an opportunity, not a weakness :)

2019-04-15

If I may contribute a historical perspective, there is some prior art on the subject of legislative encoding which touches on these questions.
https://sci-hub.tw/10.1145/112646.112660#
https://cgi.csc.liv.ac.uk/~tbc/publications/ICAIL87supp.pdf

2019-04-15

The history is fascinating - from the legislative drafter's perspective it also includes 1978 paper "Normalized Legal Drafting and the Query Method" https://repository.law.umich.edu/articles/29 . For UK legislation there is also a 1986 paper "The British Nationality Act as a logic program" http://opim.wharton.upenn.edu/~sok/papers/s/p370-sergot.pdf which spawned several follow-ups. Probably not relevant to this, but still worth a look. While I'm detouring, it goes back to Leibniz, and then to the influence of Stoic logic on Roman law - (https://www.researchgate.net/profile/Matthias_Armgardt) is a good source on that.

2019-04-18

That Australian paper is good reading, it's encouraging seeing their thinking and practice touch on the same precise issues we've been facing now in precisely representing the legislation. It's great to hear them saying there's a need to separate the legislation based rules from the operational rules.
It makes me want to clarify whats changed and what hasn't with respect to the situation they were facing. I think the concept of publicly publishing the legislation based rules in a web format and in an open "rulesbase" (to use their term) isn't something they considered tackling (being the internet wasn't what it is nowadays and web api's didn't really exist this isn't surprising).
This combination we face now of the publishing opportunity, modern component based software development and the layering effect of business/case-law rules over a publicly published rulesbase lends itself really well to an open format, open source approach for those top layers of my whiteboard pic.

Someone might want to correct me on this but I believe their work in Softlaw/Rulesburst is what lead to the Oracle OPA product today?

2019-04-18

Yes, it really is interesting to consider the question: what's different this time?

I think the idea of a "Github for law" excites people today in a way that was unimaginable 20 years ago, and one necessary ingredient for that vision is an open language for rules with an open ecosystem of tools.

I think you're right about OPA. They are to be congratulated for their commercial success; that being said, the earlier era of proprietary software might have been appropriate for business-internal rules, but where public laws and regulations are concerned, for the sake of civil society itself, we must avoid capture.

Open societies require open laws; open laws require open standards; open standards require open source.

2019-04-18

On the open source point - how strong is the open idea inside govts? One of my concerns is that govts tend to feel more comfortable buying a commercial product where they feel they can sue the provider if it goes wrong. But a bigger concern is that govts are cash-strapped and are already spending on publishing their legislation on the web, which they accept should be available to citizens without charge - but that means they are looking for value-added elements that they can charge professional subscribers for. Currently that could be for elements like annotating the statutes with links to relevant case-law. So the question is whether they can be persuaded not to see Rules as Code as a value-added service that they will charge for - particularly if the coding itself is incomprehensible to ordinary citizens, and if commercial software developers are going to make money out of using the coding, provided to them freely at taxpayers' expense, to make apps/programs that they then sell to businesses. Am I on the wrong track here, has the argument already been fully spelled out (not just nice in theory, but worth the extra cost & potential lost revenue opportunity), and have any govts signed up to the idea or at least made positive noises that can be shown to others? In particular I suspect a lot of civil servants will hear the ideological justifications for open, but think their job is to be non-ideologically pragmatic and weigh up all cost-benefit pros & cons of going down a commercial route v going down an open route. Obviously, as a legislative drafter, I think my product is wonderful and should be freely handed out on street corners to every citizen, but I know the accountants, policy-prioritisers & politicians insist on prioritising spending on other good things.

2019-04-19

From my experience, open source always raises security issues for government, as well as privacy concerns (if any personal information involved).
As for case law - that's huge! Case law rarely results in any definitive "rule" that could overlay legislation. Not to mention the system of precedence and "weight" that decision may have depending on what level and jurisdiction the case is from. It's an amorphous and ever shifting beast!

2019-04-19

RE: "From my experience, open source always raises security issues for government, as well as privacy concerns"

http://wiki.c2.com/?OpenSourceSecurityStrategy

From my experience, information and technology management always raises security issues for government, as well as privacy concerns. :-)

2019-04-19

case-law - I was just mentioning it as something where I have already seen govts suggest they could charge for annotating links between published legislation and published case reports. I agree with you that you cannot extract rules from case-law in the same way that you can from legislation. I suspect many of the digital folk think it is possible but will just be harder (leaving aside the fans of AI who claim it will soon be able to read & extract rules from all old legislation & all case-law). But I think the beauty of "Better Rules" & "Rules as Code" is that the general consensus among the people involved seems to be that we don't need to settle that argument before we start. What is different about this idea is that it starts by looking for what is readily computable in the next set of rules that we will turn into legislation - and it is about co-drafting the policy, coding and legislation so that they are all equivalent in capturing those rules. So it has started in tax & social security, and will probably go on to the business licensing legislation that is covered by "RegTech", before branching out into other areas or existing legislation

2019-04-20

I run openlaw.nz and we have made some software that exctracts legislation references from case law and makes it (and other case law data) available via API. It’s not so much a rules extraction as it is an extraction of information that would assist a readers of legislation in *understanding* it. And seeing how a a rule is applied in practice. Currently looking at machine learning. Would be very happy if anybody who’s interested wants to lend a hand - this stuff is hard!

2019-04-23

a conversation on Twitter some time back resulted in this thoughtful blog post: http://blog.cleverelephant.ca/2018/01/govt-oss-clusters.html

2019-04-25

"Am I on the wrong track here, has the argument already been fully spelled out (not just nice in theory, but worth the extra cost & potential lost revenue opportunity), and have any govts signed up to the idea or at least made positive noises that can be shown to others?"
do we make such a detailed argument in the Singapore NRF grant proposal?
I was struck while reviewing https://docs.google.com/drawings/d/1G3cs66o7u-xwJWr2FT46cd5UivJ-HojpcS3I-Q5bSOY/edit that their discovery seemed to suggest that making the rates rebate program better could cost, in a sense, millions by increasing the number of eligible people who successfully submit a form and get a rebate. If you strongly believe in the merit of the rebate program, then you wouldn't see that as a cost, but I don't know how common that opinion is in the NZ govt.
I suppose it could also be the case that improving the application process results in less, not more, total money in rebates paid out.
In any case, of course there is the added value of requiring less staff time per application.

2019-05-01

Re: " have any govts signed up to the idea or at least made positive noises that can be shown to others?" Quite a few countries haven signed up to the Open Government Partnership (OGP). Although this is not specifically targeted at publishing regulation as code, the principles underpinning the OGP are relevant and can help with the debate.

2019-04-18

RE: "Open societies require open laws; open laws require open standards; open standards require open source."

Having collaborated for two decades with both the Free Software Foundation and the Open Source Initiative I came to see the first as demand-side (i.e. the user's agency) and the second as supply-side (i.e. the programmer agency). This distinction is rooted all the way down to the bedrock of constitutional law. For a system based on "subsidiarity", government is formed and delegated authority by democratically free people (e.g. "a Republic"). For a system based on "paramountcy", constituents are granted democracy and freedom by their government (e.g "the Crown").

Thus RE: "how strong is the open idea inside govts?"

I coordinated the Canadian Government's official engagement of free/libre/open methods from 2002 to 2012.
Sample event: http://www.flora.ca/osss2002/eventssch-e.html
Sample project: < broken link removed >
Still at it: https://www.fsf.org/blogs/rms/photo-blog-2018-april-montreal-ottawa

There will always be forces in play attempting to lock government systems and data into this or that company's restrictively-licensed software and specifications. People of a republic who would defend their autonomy speak to the "free software definition". A government (or other institution) that would defend its autonomy speaks to the "open source definition". In practice, they co-exist.

2019-04-19

I should clarify - I am not doubting the strength of the principled case for free/libre/open, or that that case has been put well to govts - I am only asking which govts have so far accepted it & to what degree. So eg I can see that this would be a lesson the Canadian govt should draw from Phoenix, but is there something to point to that shows that they actually have? From what NZ & Aus digital folks say, it sounds as if the French govt has been happy to use free/libre/open material from OpenFisca, but I am not sure and it would be good to know whether any Commonwealth govts have done the same.

VISION DEVELOPMENT

2018-09-12

This is where we will discuss overall aspects of the vision - where we want to get to in 5 years, what this looks like, and what we need to do to get there

2018-09-24

A group of people – with a wide range of interests and background came together in Wellington as part of a lunch time session. Purpose was to start discussing what could be part of a 5 year vision for this work. Where do we want to be in the next 5 years? Here are some themes that emerged. Feedback, thoughts, comments?
In 2023:

- We will have a repository or knowledge base (centralised/federated/globally distributed …?) with rules related data assets that are open, transparent, can be reproduced, are unbiased, non-political and ethical . The knowledge base is a "platform" or system, is standards based and governed by "rules for rules". The system approach enables us to manage the linkages between different part of the legislation. Algorithms are transparent and incorporated in a machine-based model.
- The knowledge base is "open" which means it is available to the digital ecosystem of service providers and citizens. Citizens are able to engage early in the policy development process, for example exposure drafts, and citizens and others have chance to code and improve the rules.
- There is broad participation. Participants include central government agencies, local government and crosses borders. Participation is across multiple disciplines: Legal, IT, Policy, legislative drafters, Operations, service design. Active participation from citizens and businesses.
- We are trained and educated in using rules concepts and legislation as code. We have the skills and have built experience across multiple professions.
- We have more citizen-centric and service design processes; co-design is what we do. "Policy and operations are friends again". Rules techniques are pro-actively used as part of the policy, legislative and regulatory design. We use rules and code to test policy proposals. We use rules, algorithms and code as an input for testing the logic of the legislation, scenario modelling, prediction and forecasting.

2018-09-26

The format of the document of the rules need to be well structured. They need to be tagged with metadata to add clarification to the document in some kind of git format so others and review tags and update. Also voting on what are good tags will also help define what is important about the rule. Adding the extra layer of metadatas will make it easier for ML to understand the context of a rule.

2018-09-26

I see a future in which changes to rules and regulation are modelled and widely scenario tested before they are even put to governance in an incredibly rich way - it should help to warn us of unintended consequences

2018-10-03

I see rules as code as a fundamental aspect of Government as a Platform. What we're aiming for at the Service Innovation Lab is similar to write up here: https://public.digital/2018/09/25/making-government-as-a-platform-real/

2018-10-06

My thoughts are really very much aligned. I agree regarding the metadata, but feel it ought to be driven by a core vocabulary (engineered ontologically) that is created as a collaborative exercise. That means holding fast to the vision because the road ahead is *hard*
Be clear what you think you'll gain, why you think that's important and stick it somewhere, on a toilet door so you see it every day of the journey.
Government transformation requires a transformation of our dna. Think of a person seeking to 'transform' themselves. Does a haircut, possibly a change from one hair colour to another, loss of a few kilos and some new clothes achieve the thing? No. Transformation is a long, hard process. It is an internal one, with ever a watchful eye on who or what we are trying to transform into.
FWIW, my vision, the one that guides me, even though this stuff isn't part of my day job is this:
I want a more equal society. I want people to be able to be unhindered by our structures that exist today. As a PhD student looking at gender equity, I have no clear vision for all the things that need changing, because the structures of our institutions and societies are obfuscated. We are guessing every step of the way, because we don't know which tiny 'levers' or changes might be connected to enough of the hindering structures to bring them tumbling down.
We have all the power of the internet and all the world's knowledge at our fingertips and yet the power to wield that only belongs to the people who know how to connect it. For that reason, and that the obfuscation enables those with power to continue to hold it, inequality can only get worse, not better, as our lives become more and more complex.
I want to do the incredibly difficult work of creating a foundation of metadata in the form of core vocabularies. I want them to be open source in the LOV, and I want systems that are able to reach into those things, re-purpose the terms within and be used not only to visualise what our social structures look like, but see where the connections are. You don't need to be able to see all of the thing- the top few layers is enough, but if our legislation, policy and some processes are visible, that in itself is enough to push forward this work.
Linked Data doesn't have to be open, it just works better if it is.
This would deliver a democratised and open government, which, eventually, if we try hard enough, will mean that our very structures will be exposed to our citizens.
I believe in core vocabularies that can be run in a centralised, international way. I've seen enough of government functions vocabularies to know that despite our differences, the function of government is remarkably similar all over the world. As it should be.
Governments only exist as a structure we create to serve humanity. We're not so dissimilar.
It isn't imperative to do this globally, these are oceans to be boiled, but governments all over the world are doing this stuff. We can work smarter at this. This is the reason I said yes to starting the international vocabularies group. Now the thing that hinders us is our capacity to be open about what we're working on. If anyone would like to help us fix that, I'd be happy indeed! It is just metadata on publically available information, it shouldn't be so hard.
I believe that the most effective way to run these core vocabularies is as an ontology, where we define not only the things, but provide them with URIs and PIDs, so that they may be discoverable, knowable and 'queryable'.
The beauty of doing things this way, is that we can accommodate difference in language and difference in terminology. It's a simple thing to assign alternate labels to things if we agree

on the definitions. If we don't agree on the definitions, then we can create new terms where the difference is real and important. It also means that where the vocabularies are linked to systems, then the update is done once only, on the central thing, and then this flows down the chain automatically. We've had that capacity and technology for years now, we should be using it.
I think doing this vocabulary work is very hard, because it means a lot of discussion. People get very very attached to their ways of doing things. It's easy to be scared.
I think it's worthwhile remembering that the primary thing that stops us doing this work isn't that it is impossible (we built all these systems, we can know them, define them, change them). It's a matter of agency, authority and autonomy. There's no single person that I know, who has sufficient agency by themselves to do it (which isn't in and of itself a bad thing, we just need to be hyper aware of the space in which people work). Authority is a fleeting thing, and it is difficult in the kind of short term political environments we live in to be given sufficient authority for a sufficiently long period of time. Autonomy is something I think many of you may have, but I'm guessing that's subjective, and we're all subject to the needs of those around us, no matter how high up the chain we might be.
It's also a matter of power- and that's the bit I can't work out how to fix- this does all mean a loss of power, and that's the primary thing stopping all of this. It requires funds, and it requires those with the funds to be ok about the fact that their opinion of their own power is going to be changed by the work.
In sum (apologies for the long meandering ramble), we need to stop working in silos. We need to be able to build robust, flexible systems on really solid metadata.
When we agree that yes, we will do this thing, we need to do it in a new way. Our current ways of starting with the legislation is somewhat problematic. You're only boiling half the ocean then. What we need is a team of people building a human centred vocabulary for government. I think many of you agree with this, and I know several of you have started- I know the NZ life events work for example starts that. But when you look into the data model, they're still just a business process model. That's important- delivering a core vocabulary , data models and ontologies from the perspective of processes, legislation and policy is enough to deliver you a knowable, 'queryable' government. But it doesn't deliver you a transformed one.
The work of GaaP is to centre the platform on the person themselves, with the services acting as satellites to the core domain of 'person'.
If you look at the CPSV, and other 'person' domain vocabularies that are being developed across the world, you can see these come from a statistical background, rather than a transactional one. My vision is to bring these together with vocabularies built through user research and the social sciences (there are so many of these) to help create something truly human centred.
I believe in a centralised knowledge bank and platform for government. I believe in de-centralised management of the way those technologies are used- place matters as does the environment people find themselves in.
The humanity of our work and that human connection should never go away, but we must build systems and foundations that enable us to do that human work in a way that uses our collective knowledge, that can be updated simply, that can cope with the complexity of the lives of the people we serve. I think connected, machine readable and open is the only way to achieve both.

2018-10-19

Hi, could've been my ramble too, as I agree on close-to-everything you're saying in your post. Regarding the "lack of] capacity to be open about what we're working on" > please do check out our Finnish National Semantic

Interoperability Initiative (not an official translation...)
> https://yhteentoimiva.suomi.fi/en/ > the method and tools developed are all Open Source which means that you could set them up in an environment of your own OR just start using them through our service. Do change the language in the tools to English, since Finnish is probably as uncomprehensible to outsiders as Māori... and btw all tools have a xxxxx-test. and xxxx-dev. version too, accessible without registration or logging in. Try Sanastot-test.suomi.fi and create a controlled vocabulary or "terminology" of your own. Check also out the metadata model of the tools at > https://tietomallit.suomi.fi/model/iow/ (showing basically the connection to international standards)

2018-10-20

Hi, thank you for the links, I'll take a look! Open source is always good. In the community we've just started, we're looking at VOCBench3, also open source. Primary driver there I guess, is that we have in the group (author of VOCBench) and also the vocabulary publications team from the EC. (You can find VOCBench here: http://vocbench.uniroma2.it/). In Australia, the dominant platform is Pool Party (seen here in action: https://data.naa.gov.au/def/agift.html).
Anyway, I'd be interested to hear your thoughts about why you opted for building a Finnish platform and didn't go with the EC one?

2018-10-22

Both VOCBench and Pool Party are, if not familiar, at least something I've heard of and even seen - it's been up to my more semantic-tech oriented colleagues to decide on the tools to be used. Then again, our approach is, how to put it, a little bit more "holistic". By this I mean that the core and common vocabularies as such only provide the terminology basis for domain-based data modeling, including both what we call data component libraries and application profiles built on these. And on top of that we add a repository for codelist to it all, meaning the framework and the tools supporting it are both tools for agreeing on the concepts used in a certain context as well as tools for data models based on these agreed concepts - all linked together as linked data.
Here you can find an explanation that is perhaps a bit more consise and scientific: https://www.sciencedirect.com/science/article/pii/S1877050917303009

2018-10-22

Hi that does make sense, and I really love the approach you've taken. Would you like to join our group? And would anyone in your area like to join too? We'd love to hear how you've approached it and the decisions you've made, any obstacles or things you've learned along the way. My email is you want to send me your contact details?

SPREADING BETTER RULES

2019-07-09

A conversation about resources, advice, strategies, risks and opportunities for governments that want to follow in the footsteps of NZ's better rules project.

2019-07-09

I'm hoping to contribute to something similar to NZ's Better Rules discovery project here in Canada. While most of the threads here seem to address the substantive knowledge and information that might be dealt with in such a process, it didn't seem like there was a place to talk about how to build that sort of process itself.

Some of the things I'd love to learn from people who have been involved in NZ's project or similar projects elsewhere:

How was the time spent, and how did you decided how to divide it up?

NZ's approach was an intensive 3-week process. What are the benefits and costs of that sort of intensive approach? If dedicated time isn't possible, how does that change how you structure it?

What would you change in order to do the project remotely?

What design processes have you used, and what would you recommend?

The NZ report talks a lot about the difficulties in getting multi-disciplinary teams talking to one another about the "user-journey" of policy-intent. Is there a different journey that would be easier to discuss? Is policy-intent the right journey to frame it with?

How familiar were your participants with design methodologies, how effective were they, and what helped?

What resources (apart from the NZ report and this forum) would you recommend for getting project members up to speed on the background info? What are the categories of info they need?

Is there work that just shouldn't be duplicated? What should the next country to look at the topic do differently? Do the Digital 7 have a forum to share what is learned?

How, if at all, does multiple official languages change the Rules as Code landscape? Are there other people to involve in the conversation?

Anything else you can think of!

2019-07-10

How, if at all, does multiple official languages change the Rules as Code landscape? Are there other people to involve in the conversation?

As part of the Canada case, would you be interested in collaborating on a tiny proof-of-concept piece in which we translate abstract rules into both English and Français Québécois? I could bring some GF expertise to the table, and you could bring some expert-system skillz, and with input we could see how the abstract rule language shapes up to answer both sets of requirements. I think we've all gotten to the point of knowing enough to be dangerous. Of course I'd have to clear all this first and find time in the work schedule but in principle, as said below, we think there is value in defining some standard challenges which at least make it possible to illustrate different approaches to cooking the elephant.

2019-08-06

Hey,

Sorry for the really late reply. Catching up on emails while I'm on vacation (boo).

I'd love to do something like this if I felt like I had the time, but I don't. The Transport Canada project should be up and running soon, and suggests on Twitter that there might be 2 or 3 other possible projects inside GC in the next few months, so whenever I can I'll let them know it's a possibility, and I'll let you know if anyone bites.

2019-08-06

My company, is biting hard!

We are fully in the proof of concept stage of releasing our 'Better Rules' product to market. In other words, we are looking for commercial opportunities in this area. Our technology stack is fully developed, automating translation of law into source code transparently and isomorphically. we also provide integration services for law as code, for complex digital ecosystems.

If you could give us the details of any tendering process, or even just a contact at Transport Canada for a start, that would be great.

2019-08-07

This platform is a discussion platform and not a marketing and sales platform. Input from private sector companies is appreciated and encouraged to get to a better outcome. But if you want to pursue commercial opportunities this should be done outside the discussion platform.

2019-07-09

Just an initial response, the rest of the team can fill in the gaps. The initial 3 week investigation pointed the Better Rules work in the direction of the cross disciplinary team working around the same table to document and understand the legislation. If I was you I'd take the approach that that 3 week process discovered (by process I mean specifically the

multi-disciplinary team, the concept models, decision trees and code as is in the report) and set up such a team in Canada to follow that process with a local rule-set. The reason for this would be to experience that for yourselves, and to get a group of people on the same page. We just did this recently over 6 part time weeks with another team (roughly 8 people in the core team). I'd recommend from our experience that you lock in full days to do this as the thinking is often quite intense and to many breaks really slows down the process.
Currently I'm involved in the most complex piece of work in this space I've been involved with - with a team of four over 6 weeks at 3 days a week.
Currently there's a group of us here in New Zealand who are exploring now how to have the conversation that sees this approach further developed in a way that allows the whole of government to benefit from it and is probably the best person to give you a status update on that.

2019-07-09

Hi,
In my job, I'm focusing more on data integration issues so I am a bit downstream compared to the general objectives of the Better Rules, Better Outcomes community. But I can claim I'll be a consumer of rules, if and when available.
This presentation from Finnish colleagues illustrates how rules are used in statistical data infrastructures (there are derivation rules for individual variables but a lot of the effort (on VTL) is going towards aggregated variables).

- Antti Santaharju & Toni Räikkönen 2018) Social Statistics Integrated Information Architecture and metadata driven services https://ec.europa.eu/eurostat/cros/system/files/ses5.1_-_social_statistics_integrated_information_architecture_finland.pdf

But my main goal here is to share a few pointers which I hope are providing a different perspective - more for future stages than for the initial bootstrapping phase?

- Teemu Kämäräinen 2018) Managing Robotic Process Automation: Opportunities and Challenges Associated with a Federated Governance Model Master thesis Aalto University https://aaltodoc.aalto.fi/handle/123456789/32257 (covers the organisational challenges of getting the work done when the competences and skills are distributed across multiple organisations).
- Actionable Intelligence: Using Integrated Data Systems to Achieve a More Effective, Efficient, and Ethical Government. https://www.aisp.upenn.edu/wp-content/uploads/2015/03/Chapter1_ResourcesPg.pdf and other resources from https://www.aisp.upenn.edu/ (esp. their work on Governance models e.g. https://1slo241vnt3j2dn45s1y90db-wpengine.netdna-ssl.com/wp-content/uploads/2016/07/Governance.pdf)

AISP is interesting is interesting as slightly different kind of journey, a bit more focused on the data side. Here is another report on community learning data driven discovery in the same vein: https://vtechworks.lib.vt.edu/bitstream/handle/10919/77696/VA%20Tech%20CLD3%20National%20Strategy%2012-19-16%20%28Final%29.pdf?sequence=1&isAllowed=y

Finally, if you are interested, I can also share my experience in the spreading of ontologies: I've been involved in W3C activities and also in what is known as the Semantic Statistics

community. I have started sharing what I know
here: https://github.com/laurentlefort/semantic-statistics/wiki and
here: https://github.com/laurentlefort/data-landscape/wiki

I hope this helps.

2019-07-09

I'm launching a project on algorithmic governance solutions to interprovincial trade barriers in Canada, but these commercial rule sets seem to be different than most of what has been examined by RaC initiatives (the project is focused on improving real-time domestic commerce as opposed to individual entitlement calulations, etc.) I would say: find which 'rule sets' in Canada may benefit from computational rules and build on a public/private network of expertise.

2019-07-10

Me too! I'm based in Toronto, and a full time employee of Legalese. My main work, which I'm presently busy documenting, has been on an open source language and tools for computational legal contracts, but recent changes have made it suitable for regulations as well.

I would love to have (even make, with at least one other collaborator) a challenge task for RaC, which different vendors of tools could all implement to demo their respective approaches.

2019-07-10

As the lead for the original Better Rules discovery in NZ, I'll do my best to answer your questions. I'm also currently leading the 4 person team mentioned. We're working on a proof of concept for a very complicated rule set: a local government's District Plan. We're now in week 4 of 6 of this proof of concept. Prior to the start of the PoC I spent several weeks with my service design hat on, understanding and documenting the problem (from multiple user perspectives) developing a future state proposition and a series of testable hypothesis to determine the desirability, viability and feasibility of components of the proposed future state. This first PoC is testing just the first hypothesis and only looking at desirability and viability.

Over 6 weeks part-time (3 full days per week) the team of me (analyst/scum master), (rules as code developer), user interface/GIS developer, and subject matter expert (town planner) are doing the following activities:

- forming as a team
- agreeing a sufficient scope to test the hypothesis
- understanding the rule set by developing (and iterating) a concept diagram, decision tree and user questions (to determine if the rules are met)
- coding the rule set into a rules engine (this time it's not OpenFisca) and a separate question set to pass to the user interface
- creating test suites
- creating a user interface that accepts the questions
- creating a user interface for a planning consent application
- functional user interface testing

- limited user testing to indicate desirability of the concept
- capturing all the lessons and decisions as we go
- drafting a full report, including recommendations on how to improve the structure, language and guidance of the rule set (District Plan) - which is about to under go a major review

Your questions:

How was the time spent, and how did you decided how to divide it up?

For the original 3 week Better Rules discovery I'm sure the project log I wrote at the time will be enlightening (note: read it from the bottom up). This was sent out every few days during the project to a mailing list of interested people (I think over 100 people): https://docs.google.com/document/d/1u355lL71ZNf0YnRHK6RSYyqrosaZdrT7qgEb4Um_Ay0/edit?usp=sharing

For that discovery - it was new territory, so we didn't have so much of a plan at the beginning. We wanted to understand the space, come up with a hypothesis and explore the opportunities. We did originally plan to spend at least 1 of the 3 weeks experimenting with turning rules in to code.

NZ's approach was an intensive 3-week process. What are the benefits and costs of that sort of intensive approach? If dedicated time isn't possible, how does that change how you structure it?
What would you change in order to do the project remotely?

As said, without full team focus for full days you lose momentum. It has to be a whole team experience of understanding, questioning, learning, creating, challenging, iterating. You have to move forward together otherwise you lose time in ensuring everyone is up-to-speed. The initial premise is that the team is as lean as possible, which means that each person has value to bring to almost all decisions, which is why we have to move forward together.

For the original Better Rules discovery we had a bigger group than we do for our current PoC. People had to dip in and out for other meetings, or tag team with other members of their normal work team. We had clear morning and afternoon sessions and people said the day before which ones they would be at. However, I had to constantly make sure people were up to speed with the group. As a protector, we had included a statement in the team charter we created at the beginning that said something like: I trust the decisions the group makes when I'm not here.

For our current PoC we just had a day yesterday when remote and we communicated over Skype. Since we can co-work on Google docs and in Trello in real-time it's not too bad, but you have to schedule your time well. E.g. mornings are for heads down work - put all your questions up in a common place and then have a scheduled Q&A session.

What design processes have you used, and what would you recommend?

I can't name specific processes as I'm not a classically trained service designer - I just picked stuff up from service designers I've worked with that seems to work. As I said keeping things as lean as you can helps (team and process). I can say we use aspects of agile, lean and human-centred design. And I use a lot of sense and respond - in other words listen to the team about what's working/not working and make changes immediately. Ensure the process fits the scope and scale of the work. E.g. in a 6 week part-time PoC we're doing 1 week sprints, but only two-weekly (brief) retros, no standups, but lots of tracking through Trello and verbal check-ins that everyone is on the same page.

The NZ report talks a lot about the difficulties in getting multi-disciplinary teams talking to one another about the "user-journey" of policy-intent. Is there a different journey that would be easier to discuss? Is policy-intent the right journey to frame it with?

A lot of the difficulty was that we all came from different perspectives of the problem, different domains of expertise and had different language. The easiest way forward was to find common ground by understanding the needs of people (and systems) impacted by the policy/legislation/rule set and ask if the original intent was being met.

It depends on what outcome you are wanting from your exploration of Better Rules. If you are aiming to solve a particular problem then I would suggest a design-led discovery to understand the problem space should be conducted before a Better Rules exploration. You need a good understanding of the problem to inform your Better Rules approach and scope. I.e. what hypothesis you are testing.

How familiar were your participants with design methodologies, how effective were they, and what helped?

For the original Better Rules discovery - most of the participants were not familiar with design methodologies. As part of the tam charter, we asked them to 'trust in the process of exploration'. The unfamiliarity did make it uncomfortable for some, good constant communication and checking-in on each person helped, and by the end they agreed that they should 'trust the process'. Everyone was surprised by how much was achieved and learnt in such a short time.

What resources (apart from the NZ report and this forum) would you recommend for getting project members up to speed on the background info? What are the categories of info they need?

This might help:

From page 103 https://www.oecd.org/innovation/innovative-government/embracing-innovation-in-government-global-trends-2019.htm

I can't think of anything else right now

Is there work that just shouldn't be duplicated? What should the next country to look at the topic do differently?

Have a look at the areas to investigate further section of the report. There are still some valid questions there.
https://www.digital.govt.nz/dmsdocument/95-better-rules-for-government-discovery-report/html#areas-to-investigate-further
There is still debate about the value of the pseudo code element. For our current PoC we are mapping a very complex rule set, so we have a spreadsheet that maps out the rules (as currently written), the user questions (to determine if the rules are being met), the

conditions for when the rules are not met, and the cascading logic that determines the overall status of rule being met or not met (a bit hard to describe this in words!). So I guess it's similar to pseudo code in that it is something that the subject matter experts can understand and interact with and the developer can code from. It s also used for developing the test suites for Test Driven Development.

One thing we haven't done here yet is actually involve end users in the rules as code development process. It's debatable if that would be helpful or not, but if there was no problem understanding discovery before-hand then the end user perspective could be fundamentally missing, and might only come in towards the end when you've got an interface to user test - which I would argue is too late in the process.

Do the Digital 7 have a forum to share what is learned?

We did have a D7/9 forum for a little while. NZ produced a legislation as code demonstrator for last year's summit. We collaborated with 2 other countries on that.
But I closed down the forum (Slack channel and Basecamp site) when I left my job at the end of last year (I'm now an independent contractor) because there didn't seem to be interest in someone else picking it up.

How, if at all, does multiple official languages change the Rules as Code landscape? Are there other people to involve in the conversation?

We haven't yet tackled multiple languages. You could argue that if through the process you can help the rules become more end-user friendly in plain English then it should be easier to translate into other languages. However, there is still the technical issue of managing the same rule set in multiple forms (code, English, other languages) and maintaining isomorphism. Maybe this is where Canada should be exploring?

I hope this all helps.

2019-07-10

coding the rule set into a rules engine (this time it's not OpenFisca) and a separate question set to pass to the user interface

If I understand correctly, it sounds like an ideal technology set would include three essential components working together:

- a rule language (usability being important; SBVR, DMN, OPA, iLog might be candidates)
- an expert system interface (that answers user questions via the web or even a chatbot interface, and explains its answers)
- a natural language generation engine (with multilingual support to deliver isomorphic translations into English, French etc which could flow into published regulations)

May I ask which rules engines you've evaluated and which ones you've chosen to work with for this project? I know of several contenders including digital-legislation.net and AustLII's DataLex, as well as the Prolog family (Flora-2, ErgoAI, LPS, Epilog/Worksheets etc), but I'm confident there are others I don't know about. Forward-chaining, backward-chaining, ECA, constitutive, regulative, there's lots of jargon out there.

2019-07-10

is using JSON Rules Engine

2019-07-10

Hey - I decided for this short 6 week project it'd be good to try a light weight javascript based engine given the prototype it's powering is in the browser (https://github.com/CacheControl/json-rules-engine/).

It's been fairly pleasant to work with given what we're working on.
I think it's only fair to add I'm partly holding you responsible that I'm also now teaching myself

2019-07-11

I think it's only fair to add I'm partly holding you responsible that I'm also now teaching myself prolog and haskell

Delighted to hear it!

The Prolog wheel is a good old one and it deserves to not be reinvented.

There's a description of an expert system shell in this paper which illustrates the heritage of more recent evolutions like DataLex.
http://opim.wharton.upenn.edu/~sok/papers/s/p370-sergot.pdf

Familiarity with basic Prolog makes it easier to appreciate what's happening under the hood in demos like
https://www.youtube.com/watch?v=b6kkvvHfEOo and systems
like http://worksheets.stanford.edu/homepage/directory_publicsector.php

Evaluating a constitutive rule corresponds to backward chaining: given these facts, does this proposition hold? Why? Why? Why?

Planning corresponds to variable unification: what would it take for this proposition to be true?

Haskell is good for appreciating the principles of purity, immutability, and determinism that make FP an important paradigm for really being able to know for sure, and being able to reproduce results. But learning Haskell is a long road, and I still hesitate over whether to recommend it over Ocaml or Lisp/Scheme/Racket.

Our language L4 will borrow ideas from Prolog so we can express rules and facts in a way that is amenable to both natural language generation and automated extraction of an expert system UI.

2019-07-15

Following on from the comments made, here is what we are discussing to move the Better Rules, Better Outcomes approach forward. Feedback welcome.

We frame Better Rules - Better Outcomes as an approach that is integrated with the regulatory process to express regulation simultaneously as natural language and “code”. We include in the “regulatory process” (1) policy-making, (2) drafting of legislation and (3) regulations.
If we want to be successful in the long run this approach needs to part of “business as usual” of how we do our work. To start thinking about this, we are looking at framing Better Rules approach as a cross-government capability. With capability, we mean a combination of people, process and technology that produce outputs that form part of a repository that can be used by a wide audience (see attachment Better Rules as a capability).
Some of the elements of the capability are already existing but not always ‘configured’ in the right way or at the right time and some elements are ‘new’. For example, the process we use for Better Rules is based on existing agile methods used in software development but only occasionally in policy-making.
New skill sets we see emerging are that of “policy architecting or engineering” and the skills of “rules analysis”. These skills set can be incorporated into existing policy professions – through upskilling - or separate roles. People in these roles need to be analytical, structured, system thinkers, ability to visualise concepts and be able to facilitate the creation of the different outputs. Having “code” available early also makes it easy to run test cases and scenarios to optimize the design of regulation.
A key requirement is to have consistency across government but also be technology agnostic. Using a standards based approach is a way to achieve this. Standards like SBVR and DMN popped up as possible standards. We need tools to do this too. Ideally a tool or tool set that (1) maintain traceability between concept and decision models, rules and code and (2) maintains the relation between the natural language version of the regulation and the coded version.
We found that the value or importance of the different outputs varies depending on where you are in the government “value chain”. Policy-makers get a lot of value out of concept-models but “code” is less critical to them. For others “code” may be more critical than a concept model. For a multi-disciplinary team to be effective we need all the outputs.
The outputs should also be technology agnostic so that they can be “consumed” to develop government services but also “consumed” by non-government entities including the public.
We are still debating how that final output, “code”, could or should look like. It needs to be something that can be easily used by everybody or can easily be transformed into what people want to use.
We still have a similar discussion around the “repository”. The repository should be technology agnostic as much as possible, not be part of a specific application layer (inside or outside a government agency) and be able to managed within a policy domain (see attachment Better Rules as part of government).

And last, Better Rules as a capability isn’t going to work without buy-in from the policy-makers drafters and regulators. It can’t be driven by software developers only. We see the need, and are working on, support from the policy and drafting professions and are starting to work with agencies responsible for “capability” in government.

2019-07-15

In Jersey we are looking at a test run of the co-drafting approach, with a real or dummy piece of new legislative drafting. Currently at the stage of canvassing several interested policy depts and looking for sources coding support (all expressions of interest welcome).
Not looking for the perfect coded version at this stage. More to get a feel for how the policy development & legislative drafting processes could be impacted by taking the approach of building in, from the start, some element of establishing rules that are codable (as well as transformable into working legislation).
Second stage plan will be to obtain funds to try a project with full coding support - hoping later this year.
All signs positive so far. Will report back.

2019-07-15

"collaborating on a tiny proof-of-concept piece" - this is what we are looking for in Jersey, but not the 2 natural languages part - more like expanding the NZ Rates Rebate twitter discussion to cover a live creation of a rule, code & legislation. Is that of any interest to you?

2019-07-16

Absolutely! Very interested. What's your timeframe for this work? If we're looking at 2020 or beyond we may be able to arrange a budget internally to support collaboration. If it's to happen in the near term (2019) we might need to seek funding to cover the cost of the R&D.

2019-07-17

timeframes - now till Brexit we will play about with it internally; post-Brexit we would be looking to do something. Brexit supposed to be 31 Oct, but has been postponed twice already - hard-hard Brexit might mean busy Nov/Dec anyway, so could be 2020 by time we have capacity even for something tiny.

2019-07-16

Hello

I'm so excited to see piece above and discussion.

I think the "Better Rules" approach is 100% the best way to do things.

I've made a short video on the themes in postings. It was shown at the New South Wales Parliamentary Counsels' IT forum ...I couldn't present in person because I was presenting to the UK PCO at the same time in London!

<link removed for privacy reasons>

Anyway, I would like to emphasise one theme. That is traceability between the law and the source code. Without isomorphism in the sense of legal effect metrics, law as code is of marginal utility.

< removed for commercial reasons >

In the video, I briefly run through our 'technology stack'. One of the key components is "LogLaw". LogLaw is an open source, free to use standard for the representation of logic in law. It is totally transparent and fully traceable back to the law ISOMORPHICALLY. And the best part is that it can be complied, using open international standards, to multiple source code languages …. This gives complete transparency from law to bits and bytes.

< removed for commercial reasons >

A final note. We have proven our approach out a number of times recently, end to end, and would be happy to share!

A paper (presented to the Commonwealth Association of Legaislative Counsels' biennial conference at the beginning of this year) is also available on request. It is due for publication (along with Matthew's excellent logic paper) soon, and was also accepted at the International Conference on Artificial Intelligence and Law in Montreal last month.

Regards

2019-10-17

I've put online "The Practical Better Rules Workshop Manual" - a draft manual I've been working on and which I'm really keen on feedback on.
You can find it here - feedback here would be brilliant.
<link removed for privacy reasons>

2019-10-22

Hi all, it's been a while since I've posted :) But after making some Better Rules and Rules as Code progress in NSW I've been focusing back on the Federal Australian Government for the next month or two. There are still significant opportunities for service delivery, but am providing some advice around implications for regulation, trade and agriculture. I've been invited to speak at a number of regulator events, including the recent National Regulators CoP in Australia and the upcoming Canadian Government Regulators Innovation Showcase. The deck I have been improving over time which seems to work to get the discussions flowing and to help non technical audiences to see the opportunities is at <link removed for privacy reasons>
and I've got links in there to the excellent manual Hamish pulled together plus the ACC report done in NZ and a few other resources. I hope this is useful to others. I'll try to keep this group more posted. A few of us are also keen to do a little coordinated work to improve documentation and tools for newbies in mid November if anyone else is interested.

2019-10-23

Great deck. One question: you say that RaC is about putting rules into a machine consumable form, and allows you to create an API quickly, but then you exclude OPA or RaaP from the definition on one slide. I'm not sure what distinction you're drawing there, because both OPA and RaaP are used to encode rules, make them machine consumable, and build APIs (though OPA isn't usually used that way). Can you help me understand that?

2019-10-24

Hi, thanks for the question and it is one I get asked a lot. I've thought carefully about how to answer it and it comes down to a few things. Let's talk it through when I finally see you in person in a few weeks :)

Firstly, business systems that use rules are not the right place to provide authoritative management, provision and access to rules, as those applied usage of the rules create too easily a pressure to encode the rules bespoke to purpose. OPA/RaaP are tools largely for the applied use of rules where they are already provided in a human readable form and need translation into a machine form, but you get variance of interpretation this way, whether it is a machine translating it (OPA, RaaP, etc) or a human interpreting it. If you had just the rules available as machine consumable code, consumable by business systems that then apply the rules according to the specific context, then the translation gap is avoided, the rules are applied consistently across very different use cases, and everyone is using the same version of those rules.

The second different is the someone nuanced discovery we made about the necessity to split rules from business logic. OPA, RaaP and many others keep rules hardcoded into the business logic which ends up creating minimum logic statements that are hard to maintain and track beyond a medium level of complexity and scale. If you had the rules in one place, and then business systems that consume the rules and interpret them into business logic for that particular business system, then the rules are much more easy to maintain. For instance, in RaaP, you have to define a minimum logical construct like "theLightIsGreen = Yes" might be a condition of being permitted to drive through an intersection, but the concept of light and green are not reusable objects, so you now need to also maintain the rules "theLightIsRed = No" and "theLightIsOrange = Yes", and when you expand from lights to something else you are now having to create even more logical constructs. Basically, tools like openfisca are more able to be managed as highly modular object oriented logic which is less complicated to manage for very large amounts of rules, such as for taxation and social services rules.

You could suggest that RaaP or OPA could be used as the authoritative source for everyone to consume, but even if the technologies supported and were affordable for the massive amounts of systems that would need to consume the rules in realtime through APIs, that brings me to the third point: traceability and explainability. Any tool that just translates english into machine language may not ensure traceability of authority, or explainability of decisions, which will increasingly be critical to ensure auditing, appealability and legitimacy of those decisions.

The final difference goes back to the Better Rules part of this which aims to draft rules in a better way in the first instance to be human and machine consumable, rather than the OPA/RaaP models which assume the rules aren't machine consumable and need to be converted. This methodology is largely technology agnostic, but there are no standards for legislation as code atm, just standards used by those programs. So ideally we'd have a common protocol for describing rules as code that was understood by all tools, an HTML or TCP/IP for legislation, so anyone could talk that protocol and integrate rules into whatever platforms they used.

OPA and RaaP are usually examples of interpretation engines, but could in theory be used to host digitally born rules as code, but I was trying to distinguish in the presentation between rules that are born digitally consumable, and those made machine consumable after the fact. I hope that makes sense.

2019-10-24

Thanks for raising that question, and thanks for the great answer.

I haven't posted here before so for context: I'm currently the Business Rules Team Lead at Inland Revenue NZ, working with OPA and rules modelling, and was peripherally involved in the first Better Rules Discovery Report and the ACC one.

That background's led me to feel that to some extent, 'rules as code' is re-solving a solved problem. We're all used to deploying rules as code. You need to see the full explanation that Pia's provided here to see why this is different.

OPA can: split rules from business logic (have the underlying rules in one module and the business logic in another); store rules that different business systems consume (we've done this with income tax and child support rules); explain decisions well (in the 5 years that it's performed child support formula assessments for all customers in NZ, there have been a handful of unusual cases which we've always been able to explain and resolve).

However, OPA does generally assume "the rules aren't machine consumable and need to be converted", because that's generally how people think, certainly the people we've worked with and for. So although we've done good work implementing rules as code solutions, it's been within the status quo. It's not easily scalable across govt, even NZ's relatively small one, hasn't shifted anyone's thinking much, and hasn't got us much closer to simultaneous drafting of human- and machine-consumable rules.

Now, changing directions…from that experience, the angle I'm most interested in from here is how we use Better Rules/Rules as Code to make a difference to end users. Why is it better than what we're currently doing? How do Better Rules lead to Better Outcomes?

I know we've all been trying to tell that story, but on returning to look at the existing material after a little while away from it, it strikes me as quite technical and needing more at the human level. If we do RaC, how will this make a difference for families? For small businesses? I see comment above picking up about involving end users in the process - how would that work?

Our team has some space to explore these topics over the next few months, so will keep in touch with everyone and report back. Really keen to hear from anyone with ideas or user stories, please let me know if you're interested.

2019-10-24

Thanks , and I really look forward to hearing how you go! This is an article I co-wrote with someone I brought into this world fresh, so he saw it quite differently to either a technologist or legislator, which was fascinating. It is currently behind a paywall (not for long) at so I've copied and pasted below for convenience. The article started with regtech context only as that is the framing that rules as code often inherits in Australia due to statements from our politics, but he gets into broader user benefits that I think are quite compelling.

When we code the rules on which our society runs, we can create better results and new opportunities for the public and regulators, and companies looking to make compliance easier -

Teams all over the world, including in Australia, are already experimenting with coding prescriptive rules in legislation, regulation and policy. Beyond just 'regtech', the approach of coding the rules on which our society runs promises better results and new opportunities for the public as well as regulators and companies looking to make compliance easier.

The Prime Minister made waves recently at his annual address to the Australian Public Service when he declared a personal fondness for 'regtech' and said that he hoped we will see "within the next decade... legislation

written in computerff code". Well, in this instance, we don't need to wait a decade. In fact, the future is now. Teams all over the world, including in Australia, are already experimenting with coding prescriptive rules in legislation, regulation and policy. Beyond just 'regtech', the approach of coding the rules on which our society runs promises better results and new opportunities for the public as well as regulators and companies looking to make compliance easier.

Why should we do this?

For the uninitiated, 'regtech' refers to software that make it easier for companies to comply with their various regulatory requirements. Companies and government departments have to continually translate the relevant legislation, laws or rules into software and machine consumable languages — code — and this allows them to use digital systems and platforms to automate some of their compliance obligations.

For example, banks routinely develop systems which automate or assist with the recording and reporting of large transactions to help them comply with anti money-laundering and counter-terrorism Tnancing laws.

The private sector is currently investing heavily in regtech because it will save them money by reducing compliance costs. Deloitte Access Economics estimates that Australian federal, state and local government rules and regulations cost $27 billion a year to administer, and $67 billion a year to comply with. So it's no surprise that the private sector is enthusiastic in looking for new solutions in this space.

However, making laws and other rules machine consumable could also be used to deliver a host of beneTts for governments and the community.

Rules as Code would enable better, faster, more transparent services

Firstly, legislation isn't a scintillating read — it's drafted for precision of meaning, not readability or accessibility. This means that anyone outside of the legal industry (and many people within it) can find laws difficult to understand. If we publish machine-consumable version of our laws, we can build systems and services to make them easier to understand and apply.

An excellent example is the ATO's eTax offering. By coding its taxation rules, the government has created an easier and more efficient way for us to lodge tax returns, and has even automated certain aspects of the process by enabling 'preTlls' via integrations with banks and insurers. Also in the world of taxation, the French government has used its coded taxation rules to build a series of 'simulators' to help French taxpayers understand how tax laws apply to them.
We can take this approach with other systems and services, and use coded rules to deliver services that are not only faster, but fairer and more transparent.

That is, we can use coded rules to build automated or semi-automated systems that deliver a result, an explanation of the rules applied to get to the result, and all inputs and evidence considered. It's essential that decisions be transparent and explainable, especially for governments, whether those decisions are made by a person or a machine.

Rules as Code would eliminate needless duplication

Secondly, it would be far more efficient. Currently, we have numerous businesses each coding their own version of the same laws. This creates the risk that translations will be incorrect or misinterpreted. In contrast, a single government-provided and assured translation, made available via Application Programming Interfaces ('APIs', which make it possible for machines to speak with one another and transact) would cut down on this needless duplication. Regulators would be able to see the rules being consumed, and the community would have certainty that the rules being used by automated systems were the correct interpretation (or even certiTed to be correct). A single set of government-assured coded rules would also be a boon to the private sector. They wouldn't have to devote resources into translating the rules into a form their systems could use — saving money, increasing productivity and proTts and, therefore, increasing the tax base.

What’s more, individuals, small businesses and startups could use the APIs as well, removing expensive barriers to competition and creating a more even playing Teld. For example, at this year’s GovHack, one team used the eligibility rules from ServiceNSW’s cost of Living Service, coded by the Digital.NSW Rules as Code
project, to create an education support tool to guide parents through the available rebates and services from kindergarten to high school. And they did this in a single weekend.

As a bonus, when the rules or laws change, the code can be amended and the linked systems would be updated automatically. Everyone wins.

Rules as Code would give us a new way to model and test laws Thirdly, if our laws are machine consumable, we can take a whole new approach to testing them, and modelling different legislative approaches. SpeciTcally, we can use coded rules to rapidly test proposed rulesets against dozens, hundreds or thousands of test scenarios to see if they will operate as intended — what’s known in the software world as ‘regression’ testing. We can also model more efficiently — try out different options, and see which gets us closer to the desired effect.

Of course, this ability is only useful if there’s an opportunity to change the rules. If the ruleset is contained in legislation that has already been enacted, it’s going to be more difficult to amend them. This means that we need to rethink how we draft rules. It won’t be sufficient to draft and pass legislation, and circle back to do the translation — we have to draft the code and the human-readable text at the same time, and allow them to inbuence each other.
In this respect, we can learn a great deal from our antipodean cousins in New Zealand. The “Better Rules” work being done by the New Zealand Government has
shown that unless we modernise how we draft policy and legislation, then we will miss the opportunity to make fundamentally better rules in the Trst place.
As our Kiwi colleagues are currently ably demonstrating, effective test driven regulation and legislation means Trstly assembling a multidisciplinary group of policy, drafting and rules consumers (service designers and developers) to understand and agree the purpose, concept and logic behind a piece of legislation with an accompanying coded ruleset. By collocating with drafters and coders, this group can then simultaneously co-draft human and machine readable versions of the rules for testing — with humans and machines. This allows for more holistic modelling of impacts, and provides and the opportunity to test the coded rules with end users (regulated entities, service providers, etc.) before publication.

Ideally, if dealing with a legislation ruleset, the draft legislation would be published for consultation together with an API enabling access to the draft coded rules, and stakeholders could test the rules and use the code to inform their submissions. Once enacted by Parliament, the machine readable form (the API) could be publicly available immediately. Regulated entities could link their systems to the ruleset instantly, reducing the time and cost to implement and reducing the risk of mistranslation or variability in interpretations.

We still need human-readable rules

Of course, this doesn't mean that we should only write legislation or policy rules in code. Humans still need to be able to read and interpret rules. Further, machines can't do nuance or interpretation. They only deal with absolutes. The rules we can currently code effectively are prescriptive — black or white, yes or no. Many of our laws are not prescriptive, but require subjective perspectives and nuance, consideration of the various circumstances of the case. That is, we need humans — administrators, regulators, lawyers, judges — to interpret and apply them.

Even in those scenarios, coded rules can help — by automating the black and white aspects of the question, we can escalate the parts that require nuance for human consideration. This will allow us to dedicate our human resources to the difficult and complex work, leaving the process-driven drudgery to our robot friends.

How do we do this well?

We need to be open.

As a society, we've put a lot of effort in making sure our laws are available to everyone — we publish them on websites in full, such as legislation.gov.au, and make them available for use and reuse. We need to do the same for our coded rules. The code needs to be visible and inspectable — if our rules have an error, we want and need to know. We also need to make it easy to use — it can't be proprietary or expensive to adopt. So it has to be open, and based on an open source platform.

We can learn from the world of encryption, where the community ensures that new algorithms are thoroughly inspected, tested and debated. This means some are
eventually cracked but, ultimately, the most robust implementations survive.

We need standards. We need to establish standards or users are going to run into problems with interoperability. We need to think about minimum standards for the logic of legislation so that all legislation as code APIs behave predictably, regardless of the platforms used. Such a standard does not yet exist, and most 'rules as code' platforms focus entirely on the translation of rules into code rather than taking a born digital approach to rules in the Trst place. Worse, most current platforms don't allow for easy and consistent API access to the rules independent from business logic, which limits the reusability of rules.

It is also worth noting that everytime we encode government rules in yet another business system, we are contributing to the complexity. Ideally we would have api.legislation.gov.xx available where government is the authority of those rules so the rest of us can simply consume from government.

We need a multidisciplinary approach. Right now, there are very few lawyers who can code, and there are precious few coders that can grapple with public policy and laws. The principles of service design have also yet to enter law school curriculums. One day, that may change, as law schools are currently busy training the next generation of tech-enabled lawyers. But for now, the only way to get the necessary skills in the drafting room is to be multi-disciplinary — take the drafters, the coders, the policy experts and the designers and have them work together to create the rulesets.

Where to start?

We should begin in logical areas where strong prescriptive rules exist already. Financial institutions have led the way, and there is a lot of existing work that can be used to forge ahead quickly. There are many regulatory frameworks where strict rules apply — for example, building codes, food safety, logistics. Ultimately, the only way to eat an elephant is one bite at a time. We just need to make a start, and keep building.

It’s here we’ll start running into some resourcing problems. To learn to code rules well, we need to experiment. We’ll need to devote some time and money to trying different approaches. We’ll need to be able to fail, and fail safely. This will require a shift in approach and in risk tolerance from most governments.
This is a new discipline, and we’re going to have a skill shortage. Of course, this is hardly unusual. Whenever new technologies are developed, we have to build the skills in the work to use them — we’ve done it with everything from shipbuilding to cyber security, and we can do it here. The challenge will be Tnding the resourcing to provide the right people with the time and space to learn.

We don’t have all the answers yet. But we mean to find them, and at the very least it promises to be an interesting journey.

2019-10-24

I really enjoyed the way this article frames the topic, from a communication perspective it's been tricky determining which end of the topic to start at (depends on the audience) but this is a good execution of the Rules-as-Code first and using that to introduce Better Rules

2019-10-24

This is great, thank you for sharing here.

It's definitely been our experience that when you build code/products off previously drafted legislation, it really limits how customer-friendly you can make it. So if we can do service delivery at the same time as policy/legislation development, and make the service *the law* rather than an interpretation and simplification of the law, that will be transformational.

It's challenging to tell those two stories at the same time – this is what we've done so far and how it's made a difference; this is how much more of a difference it could make with full buy-in. But the article does a good job of bridging that gap, and I guess we just need to keep getting chunks off whatever part of the elephant is closest.

2019-10-25

So correct me if I'm wrong, but here's what I understand you're saying, and I'm stealing here from some ideas that I read recently in a paper on ensuring quality in automated legal services...

There are three different kinds of logic involved. There is legal logic, application logic, and interface logic.

Legal logic is "a person over the age of 18 is a minor". It is a statement that comes from the law, and relates to the legal consequences of facts.

Application logic is logic that is required for the use the rules are being put to. For example, if you are asking whether there is a legal marriage, you need the legal logic about what constitutes a legal marriage, but you also need facts about two people that you can use with those rules. If you have facts about one, or three, it doesn't work. It needs to be two because of the purpose, not because of the law.

Third, there is interface logic. Here, you might have something like "if I already know this person is a minor, don't ask their age." This is logic that doesn't have to do with the law, or with the purpose to which the law is being put, but just to the interface between the user and the application for collecting information.

When you say "applied usage of the rules create too easily a pressure to encode the rules bespoke to purpose", what I hear you saying is that "Rules as Code" should include only the first kind of logic, and the second and third kind should be separate, to make the code more re-usable.

You seem to be saying we should go from the natural-language rules to an encoding of just those rules, and have the translation of the rules from NL to code happen before we have a particular use in mind for that encoding.

These two things are 100%, wildly, incredibly correct. I will die on those hills.

It is NOT correct to say that combining legal logic with application and interface logic is what OPA or RaaP are "for". With regard to OPA, it is 100% correct to say that is how it is typically "used". But you could theoretically do exclusively legal logic in one OPA encoding, and include those rules in an OPA application that keeps the application and interface logic somewhere else. But RaaP doesn't even have an interface layer. So at most RaaP applicatons confuse legal logic with application logic, which is better but still bad. But again, they don't need to.

So it's not that RaaP and OPA are "not Rules as Code". They are not Rules as Code as they are typically used. Rules as code in RaaP and OPA is possible, but you would need to use them differently.

The second thing you seem to be saying is that there are aspects of OPA and RaaP that "force" you to make the mistake of combining legal and application or interface logic, which makes the logic statements complex and hard to maintain.

I've experienced this in OPA, so I know what you're talking about, but it's two different things. First, OPA is not 'forcing' that behaviour, it's just rewarding it. You could do it the other way, it would likely be harder.

But the second issue you raise with OPA (and I don't know if it is true of RaaP), is the complexity of the statements. This is true, but I think you're attributing it to the wrong cause. There are techniques that you can use in OPA to break down complicated rules into smaller ones, which enhances how they can be reused. But it creates a more complicated ontology. It's a trade-off. But, neither OPA nor RaaP have a feature called defeasibility. You can't write rules that are exceptions to other rules in the code. We do that all that time in natural language rules. Because you can't do it in the code, in OPA and RaaP you have to include things that have been written as exceptions as a part of the original rule. In fact, they have to become a part of every other rule to which they might apply. And they no longer are a rule by themselves in the encoding. That makes the encoding of the original rules bigger, and more complicated, and makes the code look unlike the original law, which makes it harder to maintain.

So this complexity problem, in OPA and RaaP, exists. And it can be made worse by combining legal and application or interface logic. But you don't have to make that mistake. The rules will STILL be unnecessarily complicated because of the lack of defeasibility, though.

The lack of defeasibility is a reason we should look at RaaP and OPA as "not good enough". But not a reason we should look at them as "not Rules as Code".

I'm not familiar with RaaP enough to understand what you're saying about the lack of reusable objects. I know Neota Logic has a similar problem. It's not possible to determine whether or not the person they want as their beneficiary is also the person they chose as their witness, because it's impossible to tell the difference between two people with the same name, and the same person.

But I know that RaaP does not use full first-order logic. Whereas OPA is able to to ask questions like "is it true for all people that they have beards", RaaP can't, because it doesn't have universal or existential quantifiers.

Again, that is a good reason to say that RaaP is "not good enough." But that's different from saying that something you do in RaaP is of necessity "not rules as code".

"Basically, tools like openfisca are more able to be managed as highly modular object oriented logic which is less complicated to manage for very large amounts of rules, such as for taxation and social services rules."

This is likely true. Everyone who has played with openfisca reports that they love it. I haven't tried it, yet. And part of the reason I haven't tried it yet is because I was under the impression that it, too, forces you to combine application and rule logic, but in a different way.

Maybe someone who has used it can tell me, but in logical programming languages, if the rule says "when it is raining, the road is wet," then you can use that rule for two different questions. You can say "it is raining, is the road wet?" and it will say "yes." And you can say "the road is dry, is it raining?" and it will say "no." The logic goes in both directions. My naive understanding is that when you encode things in OpenFisca, they are unidirectional. It can take you from income to entitlements, for example, but it can't take you from entitlements to an estimate of your income.

That, too, is replacing the logic of the rule with the logic of the application. Just more subtly. If you want openfisca to be able to answer both questions, the way a logical rule does, I think you would need to write a totally different set of code.

Maybe I'm wrong about that, and if I am, great. OpenFisca will be the next thing I learn. :) But I think it's incorrect to say that RaaP and OPA combine rule logic with application logic and OpenFisca doesn't. They all do, but differently.

On traceability and explainability, OPA is actually better equipped than anything out there for explaining its results. Neota Logic has similar features. They both work quite well for that purpose. Generating explanations from defeasible tools is possible, but there are no open source implementations of it. That's another big hole in the technological landscape for rules as code. And of course, proprietariness is a problem that make RaaP and OPA and Neota poor choices for public service applications where transparency and accountability are key. But again, it doesn't make it impossible to apply them to "Rules as Code", or to use them to express purely legal logic. You could. They are just "not good enough".

Your last point on the standard is a good one. In order for an expression of a law, whether in natural language or in code, to have meaning, the language that you are using needs an agreed-to semantics. Natural language semantics are agreed-to, but we agree that they are fuzzy. Programming language semantics are also agreed to, and are strict. But the syntax usually makes the language hard to read. There are people working on that problem, like ACE (Accepto Controlled English).

It would be better if we had a standardized semantics that we had all agreed to write legislation in, and then we could write both natural language legislation and encoded legislation in that semantics. If you used something like ACE, the natural language version of the law could even BE the encoding, itself. But that's making the perfect the enemy of the good. If you talk to the people whose job it is to study the semantics you need to encode laws, their favourite thing in the world is to disagree about what the semantics should be.

If we don't HAVE an agreed-to standard, we can't point to RaaP and OPA and say "that's not rules as code" because it doesn't comply with a standard that doesn't exist.

Here's my point. "Rules as Code," as I understand it, is the idea that the legal logic, and only the legal logic, can be made machine usable, and more reliable than the interpretations we have now because we choose to write the natural language laws in the same semantics the encoded version uses, or a very similar one. So we know the two representations mean the same thing, and the interpretation only needs to be done once. THEN you build applications with it.

Nothing in that sentence precludes you from using OPA or RaaP or OpenFisca, or anything else to do it. If it's technology agnostic, it can't be technology exclusive.

And I think there's really good reason NOT to be exclusive. Saying "RaaP and OPA are not Rules as Code" discourages people from trying two of the best tools currently available. RaaP is designed to allow you to use their code, and their reasoner, for free, over the web, in your app. Nothing else does that. OPA makes it possible for subject matter experts to write the rules in Microsoft Word, in a more user-friendly way. Nothing else does that.

We want people to try these things, so that they will feel the possibilities. We don't want to tell them "don't try unless you're able to install a python development environment." That's counter-productive.

Now, none of them are good enough. That's basically my LLM thesis in a nutshell. I'm not here to say "everyone should be using RaaP or OPA". If I thought that, I wouldn't spend so much time building Blawx.

But I also recognize that all of these tools do something none of the other tools do. RaaP creates APIs. OPA uses controlled natural language in Word. Neota uses flowcharts. ErgoSuite has explanations, Flora-2 has defeasibility, NAI uses text-highlighting, Blawx uses drag-and-drop puzzle pieces, OpenFisca leverages object-oriented programming techniques like inheritence to make it easier to test amendments. All super cool stuff. I want to try all those things, and see which ones matter to me. I want other people to do that, too.

What's more, and perhaps more important, is that all of these technologies are not only technologies. They also represent a community of people. A community of users, or developers, who might get excited about they idea that the skills they have could be used to make the world better with better rules. We need all those people to feel invited to the party, and there is no reason to exclude them.

2019-10-25

Heya, there is a lot here, thank you for the thoughtful input. I think the thing missing in your response is less about the technologies or language, but more about roles and responsibilities. What I am proposing is that we need, quite literally, api.legislation.gov.xx. A government provided version os, what I think equates to somewhere between your legal and application rules in a machine consumable form (to use your minor example pun alert], age => 18), because government is responsible for those rules, and so maintaining a usable, consumable form that applications can draw from should be the role of government in the 21st century. Then everyone can use whatever tools they want to deliver their business needs, compliance, services or products on the back of well maintained programmatic rules.

People quickly get into what the tool should then be for the rules, and I would suggest we don't need to solve that problem immediately, but we do need to ensure this model of rules provided by gov and consumed by others is maintained, because the alternative is where we are now, which is myriad and variable interpretations of the rules applied in myriad (and often non compliant) ways, often with no traceability or explainability of authority for the decisions or actions taken on the back of the applied rules.

OPA and Watson and various other rules engines are a bit too black box in the explainability department, and currently too expensive to consider for scaling to be an authoritative source of rules that everyone could draw from, and they also combine the different forms of logic you outline, so aren't easily consumable rules in a standardised format. RaaP is more transparent which is great, but the logic structure is a bit too rigid (could be more object oriented logic, if that makes sense). But I do take your point, and I need to improve the nuance in the presentation to make it clearer that it is in how these tools are currently used to be interpretation engines that we are missing out on rules born digital, which is more in the Better Rules end of the spectrum than simply rules as code.

Indeed, it has been very surprising to me to discover that it is new to legislators and regulators to consider that their beautifully crafted prose is currently translated into code every day. This is why we need to bring policy and implementation teams into the same room once in a while :)

2019-10-28

Quick comment: I read (part of) essay as an argument for the layering principle, as encountered in network architecture, and for separation of concerns, as found in the MVC pattern: https://en.wikipedia.org/wiki/Model%E2%80%93view%E2%80%93controller

This argument shows up in the model-driven-architecture world, e.g. on page 189 of https://drive.google.com/open?id=1XKDVeI0IFUlFdkv6dsnwzdg9XExkNL-E

2019-10-29

Thanks so much for sharing your presentation. Some really important food for thought and I love the work that you and others are doing in this area - hard to overstate how important of a foundation this will be for some of the ambitions as to how government can and should be transforming in the years and decades to come. Hopefully will get a chance to connect and discuss while you are up in Ottawa next week for FWD50!

2019-10-23

Hey, everyone: I did some updates to Blawx.com last night that are designed to turn it into a sort of proof-of-concept for a combination of what "Regulation as a Platform" (RaaP) from Data61 does (answer legal reasoning questions over the web, so your app doesn't have to encode the rules itself), but with a more friendly user interface for encoding the rules. I have also released a module for using Blawx in Docassemble, as a way of demonstrating how the integration works. I'm going to be demoing Blawx+Docassemble at the Regulatory Innovation Showcase (where I finally get to meet!). If anyone here is interested in playing with Blawx.com, and you have any difficulty at all, there is a slack channel for Blawx at blawx.slack.com, and I'd be happy to walk you through whatever you're trying to do.

2019-10-25

Thanks You are, as usual, more concisely articulate on this matter :)

Hi everyone

We are getting lots of questions from NZ and globally about Better Rules. What it is or isn't. We have put an animated video together that will help us to tell the story. Video link:

https://www.youtube.com/watch?v=6oSWUglaXXs&feature=youtu.be

This is one of the resources we have been working on. Over the next couple of months we are working on further developing www.betterrules.govt.nz as a place for resources, tools and stories.

2019-12-11

Started to spread the video actively in the Finnish Public Sector; an extremely helpful marketing tool to boost the movement also up here in the Arctic :-) keep up the More Than Good Work!

Off topic: both FIN and NZ receiving lot's of simultaneous public attention due to the change of Prime Minister & the "viral" tweet with our main party leaders & ministers in the same photo collage > https://www.theguardian.com/world/2019/dec/09/finland-anoints-sanna-martin-34-as-worlds-youngest-serving-prime-minister

2019-06-xx

Hi everyone,

I have written a guest blog for the OECD - OPSI on Rules as Code.

The blog covers using the Better Rules approach during a COVID-19. We supported the development of a digital tool for businesses to check their eligibility for the Wage Subsidy using this approach.

Please have a read.

<link removed for privacy reasons>

2019-06-xx

Hey, Great work! I'm very interested to know what you mean by a rules engine built into SilverStripe.

RULES AS CODE – IMPLEMENTATION

2019-05-31

Just flagging another proprietary provider (Datalex) in this space with some interesting alternative perspectives. Be keen to find out more of how they're achieving what they're doing from anyone involved with them.
Aslo you mentioned you wanted to see something of their's on twitter - I got some links from TJ in NSW.
http://austlii.community/foswiki/DataLex/
http://austlii.community/foswiki/DataLex/ElectKB
http://beta.datalex.org/app/?rulebase=http%3A%2F%2Faustlii.community%2Ffoswiki%2FDataLex%2FElectKB

2019-05-31

Thanks - One comment: 'DataLex' is not a' proprietary' provider (in the sense of 'commercial') since it is part of AustLII - free access and part of two Universities. Also, because we are partners in the operation of NZLII, we have strong NZ ties, and are very interested to see the DataLex software and Communities environment used on NZ legal projects. The link you gave - http://austlii.community/foswiki/DataLex/ - gives access to demonstration DataLex apps (only for training) and development tools,

2019-06-24

For those interested in testing AustLII's DataLex software, the Developer's Manual, 1st Edition, has now been completed (70 pages) and is available in two formats:

1. a PDF version of the whole manual is at https://papers.ssrn.com/abstract_id=3408555
2. the wiki version is at http://austlii.community/wiki/DataLex/DataLexDeveloperSManual
   Test apps can be written and run at http://www.datalex.org/dev/import/

Feedback welcome on all aspects.

Progressive updates, examples etc will be added to the wiki version, and periodically collected into new editions of the PDF version.

RULES AS CODE – LITERATURE / RESEARCH

2019-05-03

A thread to discuss interesting research and literature in the Rules as Code space

2019-05-03

I hadn't seen this computational law paper from 2005 until just now:

in Proceedings of the International Conference on Artificial Intelligence and Law, Bologna, Italy, 2005. http://logic.stanford.edu/publications/love/computationallaw.pdf

ABSTRACT: Computational law is an approach to automated legal rea-soning focusing on semantically rich laws, regulations, con-tract terms, and business rules in the context of electronically-mediated actions. Current computational tools for elec-tronic commerce fall short of the demands of business, or-ganizations, and individuals conducting complex transac-tions over the web. However, the growth of semantic datain the world of electronic commerce and online transac-tions, coupled with grounded rulesets that explicitly refer-ence that data, provides a setting where applying automatedreasoning to law can yield fruitful results, reducing ineffi-ciencies, enabling transactions and empowering individualswith knowledge of how laws affect their behavior.

2019-05-20

I was browsing past RuleML webinars (http://wiki.ruleml.org/index.php/RuleML_Webinar) and came across this paper by others: https://www.researchgate.net/publication/319903831_Legal_Patterns_for_Different_Constitutive_Rules "Legal Patterns for Different Constitutive Rules". It makes some remarks about SBVR which may find useful.

"Several rule languages exist that manage legal rules, and rest on solid logical foundations (e.g. LegalRuleML, see survey 13]). Unfortunately, those layers of technology are difficult for lawyers to grasp, and the related solutions are still out of their reach."

That links in turn to https://www.researchgate.net/publication/311451642_Requirements_for_an_Intermediate_Language_Bridging_Legal_Text_and_Rules which describes Mercury, a language built on top of SBVR.

RULES VALIDATION

2018-11-19

A conversation around the validation of rules.

2018-11-19

Hi all, I'm involved in the elicitation of rules from income tax and social security legislation. The catalogue of rules runs into the thousands and reviewing of the rules by the business is a fairly arduous process. Our next phase of the project is to validate the rules that have been captured. Our rules have been categorised into guidance, process and system, and we're largely focusing on the system and process rules. I'd be interested to know how you guys have been validating the rules that you have captured to ensure they're accurate. Have you relied purely on sign off by the business, or have used a program of testing to validate the rules vs current processes and data? Thanks in advance!

2018-11-20

At Inland Revenue (NZ) we do a similar sort of thing at the Business Rules Centre. Our elicitation of rules is driven by what is required by the business, and what we deliver specifically will change depending on which business unit we are delivering for. Generally we will have our rules signed off by a legal and technical services team, because Inland Revenue is governed by legislation and the rules are usually used to ensure compliance with said legislation. We will also occasionally get sign-off from a business owner too, but this is on a need-be basis. The more sign offs you need the longer it takes to deliver outputs.

2018-11-21

Hi, thanks for the response. The scope of our elicitation is rather broad... the entire income tax and social security legislation, so rather than pulling out the relevant elements to deliver a packaged element of functionality, the entirety of the legislation is being delivered in one artifact as a business rules catalogue. Hence why sign offs are taking a very long time.

2018-11-21

As we code the rules we run automated unit tests against expected outcomes. By creating a user interface for a rule set then anyone can functionally test the rules against test scenarios. So we get the business process owner to test the rule set and then sign-off. You can see and explore all the variables of the rules we have coded here: http://www.rules.nz/

2018-11-22

Hi, this is rather radical, but I'm wondering if an "executable specification" style approach has been considered?

This would embed examples inside the legislation rather than as separate unit tests. The examples would be discussed and signed off prior to, during or after implementation of the legislation. Any implementor of the legislation as code would be able to automate and run the examples against their implementation.

Examples alongside the legislation would have really helped us at the Better Rules hackathon when we were trying to calculate the number of weeks in a year where the number of weeks means the number of full weeks plus any part week. In a leap year, could this make 54 weeks, or could we hardcode to 53?

I realise that's a single, atomic example and not sure how this would scale or work across different, complex rule sets? Part of the art is to choose the minimum set of examples that describe the intent, including boundary cases.

I've seen this approach work well in business scenarios, particularly with financial rules.

I'd be keen to hear any thoughts on this approach?

2018-11-23

RE: "wondering if an "executable specification" style approach has been considered ... embed examples inside the legislation”

This is exactly what Xalgorithms Foundation has designed (currently in 'alpha') and is currently testing with some genuine fiscal rules. If you would like to participate in expressing and testing some rules, please let me know. The non-specialist-human-maintainable executable expressions in the "Xalgo" specification are intended for use in schedules to legislation -- they should never be in the body of the legislation, because you don't want to have to go back to the legislature to fix a bug. From our website: "Xalgo is a generalized means of expressing any of the computable functions of legislation, regulations, policies, standards and agreements. Official fiscal instruments, standards, and regulations could in the near future come with an attached ‘schedule’ containing any relevant computational tables under free/libre/open or public domain terms. Legal authorities could treat validated table-oriented algorithms as de jure official translations for the automated deployment of computational rules of commerce." https://xalgorithms.org/xalgo/ To summarize the basic format we created simple a+b=c rule, such then when a properly structured data package from an "invoice" of a transaction with certain dates from an certain jurisdiction, which contains a value "a", arrives to the data fabric (called Interlibr), this rule will be discovered, it will add "b "to "a", and return "c" along with the whole a+b=c rule code. Here is what this rule looks like in "Xalgo":

```
EFFECTIVE
IN "CA-ON", "CA-QC"
FROM "2018-04-01T00:00"
TO "9999-12-30T23:59"
TIMEZONE "America/Toronto";

META
VERSION "0.0.1"
RUNTIME "0.4.0"
CRITICALITY "experimental"

WHEN envelope:type == "invoice";
WHEN item:quantity.value > 0;
WHEN item:id.value == "a";

REQUIRE org.xalgorithms.examples.a_plus_b:all_bs:0.0.1;

ASSEMBLE changes
COLUMNS FROM table:items
COLUMNS FROM table:all_bs;

FILTER table:changes
WHEN @id.value == @code;

MAP table:changes
USING new_price = add(@price.value, @b);

REVISE table:items
UPDATE price.value
FROM table:changes;
```

< removed for privacy reasons >

Here is a conceptual discussion of various types of rule systems: https://github.com/Xalgorithms/general-documentation/blob/master/concepts/on.rule.systems.md

I will be sharing a draft paper on 5 December at the World Trade Symposium (London) which goes much further on the concepts, particularly outlining our Xalgo data model for expressing 'worth' in 'money' through 'price'. I'll share a link here when it's out.

2018-11-22

This is similar to issues legislative drafters have grappled with, about whether to use examples in the text. It goes back to the question of what has authority if there is a mistake, clash,or unanticipated scenario . There has to be a hierarchy system so you know whether it is the text rule, the text examples, the code rule or the code examples, or whatever combination or ranking. And the effects of that have to be clear enough to whoever passes the legislation. Natural language legal text is hard enough for most legislators, but asking them to approve code looks like too much - that might be where examples could help.

2018-11-23

RE: " the question of what has authority if there is a mistake, clash,or unanticipated scenario" This is no different from current implementations in executable code. If the income tax SaaS I use to file my taxes today make a mistake, it's the natural language text that is used to determine the correct answer -- and if something is ambiguous, then a human judgement is made by a tax officer. By adding non-specialist human readable execcutables in a schedule to legislation, this will help legislative drafters discover impractical requirements in the natural language expressions.

2018-11-23

Thanks. Do you mean you don't see the coding ever being authoritative (just government-endorsed, not Parliament-endorsed, which is very different), and if so does everyone else agree? Then "machine-consumable legislation" is short for the much less ambitious "machine-consumable non-authoritative supplement-to-legislation". I am all for helping drafters discover the impractical elements, but I thought NZ were being more ambitious than that, though I may be completely wrong.
Can you point me to an example of a "non-specialist human readable executable", as I don't know what it means? Putting it in a Schedule comes

back to my point though - normally Schedules are as authoritative as the rest of the legislative text (at least in Commonwealth jurisdictions), and it is no longer normal to put non-authoritative text into the legislation itself (whether in the main body or in a Schedule). Instead there are Explanatory Notes/Memoranda, which are non-authoritative guides that the Parliament is taken to have seen, but which you can't rely on in court.

What is an "income tax SaaS" - is it a government-endorsed IT system for completing & submitting your tax return? If it makes a mistake and lands you liable to a penalty for late payment or a prison sentence for failure to declare, is that hard luck on you or do you expect to be let off? I can see you could be let off, but that is because the agency you have to pay is the one who gave you the faulty software to use - the government.

But what about when the Parliament makes a law about money employees can claim from employers, and the relevant Govt Dept codes the law non-authoritatively and invites software developers to provide programs for employers to invite their employees to use to make claims. If a fault in the Govt coding (not the program developed from it) leads to employers or employees breaking the law when they use it (or use programs based on it), then Govt aren't in a position to let anyone off, and employer & employee are not at fault. Do Govt have to pay back everyone's losses? If so not many Govts are going to want to code any law except tax and social security (where Govt is the payer/payee). Sorting those would be an admirable end in itself, but I am trying to get an idea of the aims here, and I keep getting the impression that the ambition is wider. That seems to me to involve working out a way for the embedded code to be authoritative (equally with the text, or above it) - that might mean watering it down to pseudo-code so that the legislators are no worse off trying to understand than they are with legislative text, or it might be that visualisation tools make real code understandable, or there might be some tech solution I have no idea about (you can see I am thrashing about on the IT side).

2018-11-23

RE: Do you mean you don't see the coding ever being authoritative (just government-endorsed, not Parliament-endorsed, which is very different), and if so does everyone else agree?

1. It is not practical to make digital literacy a prerequisite to running for elected office.
2. A "schedule" to legislation is enforceable. And the executable item in such a schedule shall ONLY implement what the natural language description of the legislation states, nothing more and nothing less.

Please clarify the distinction you are making.

RE: Can you point me to an example of a "non-specialist human readable executable" and "might mean watering it down to pseudo-code so that the legislators are no worse off trying to understand"

I thought I did show that in: [broken link] This is practically psuedocode, but it runs.

RE: " and the relevant Govt Dept codes the law non-authoritativel"

What is non-authoritative about government official translations in to executable code?

RE: "If a fault in the Govt coding (not the program developed from it) leads to employers or employees breaking the law when they use it (or use programs based on it), then Govt aren't in a position to let anyone off, and employer & employee are not at fault. Do Govt have to pay back everyone's losses? "

We address this with templates for services such as Xalgo-Verify and Xalgo-Indemnify (scroll down to the last two of four)
https://xalgorithms.org/niche-business-service-templates/

RE: "income tax SaaS"

"Software as a Service" tax applications, like:
https://www.taxcalc.com/cloud
https://tax.thomsonreuters.com.au/onesource/e-filing-manager
... I think in NZ online tax returns are hosted directly by gov, no?

2018-11-23

Thanks, yes I was angling towards text examples to accompany text rules in (or adjacent to) the legislation. Examples are used all the way from politicians (eg. the examples that accompany budget announcements) to developers (running as automated tests). By combining specific examples with the general rule, we improve and share our understanding. Examples are easier for people to understand and pick holes in. They encourage people to consider other examples - "what if ....?".

My understanding of legislation as code is that it is aiming to implement code-based rules, which appears to be where xalgorithms is aiming too. Is anyone working on implementing text (or code) examples alongside the legislation?

2018-11-26

- others on this thread. One thing Id like to understand Is what you think of when you talk about "code" . I understand that in the standard programming language, but what about a Natural language based Rules engine or an "Expert AI System" For an AI system I thought you trained it rather than coded it?

2018-11-26

Thanks - you are stealing my question, only mine is much more basic and involves wanting people to explain to a non-tech person what is new about all this. I am a legislative drafter trying to understand what this might mean for us, whether it can improve the work we do, and whether we can do something to contribute from our end (first by helping tech people understand how legislation works, then by drafting in ways that help).

I can see AI might one day work to comb over old legislation, but my work means my interest is in using human intelligence in the drafting stage so that we don't need AI for new legislation.

At the lowest end that might mean just a bit more tech-friendly drafting, such as trying to isolate the discretionary elements from the calculatable elements, with the policy & tech people coming up with a separate non-authoritative version that is published.

I think the next level up is having a way that, while we are writing it, we can mark-up the legislative text for a tech "ontology" to make it more readily machine-consumable. We are already benefitting from work done on mark-up to tell machines "this is a sub-paragraph" and "this is a cross-ref to this other piece of legislation" and "this amendment changes this legislation from this date". Equally there have been companies producing programs to execute legislation for decades, both for governments and advisers and businesses - but doing it in isolation by themselves interpreting the legislation after it has been enacted. But I think what might be new in the Better Rules &/or "Legislation as Code" ideas is about the mark-up telling the machine "this is a defined term and this is its definition (and it is the same as that one in that other law over there)", "this is a duty, this is the person it falls on, this is the action they have to take, and this is the consequence if they don't", and so on. I am assuming that could mean apps could be produced that read that mark-up and can then answer user questions like "do I need a wotsit licence if I only wotsit once a year (or as a hobby, or if I already have a thingy licence or our parent company has a dingbat permit)?".

A level up from that might be having pseudo-code, or some basic executable code (whatever that might mean), published alongside the legislation and having some formal status. The formal status might range from having been endorsed by the legislature as an equal/overriding part of the enactment, through being published by government with some indemnities for users, to being an open source public joint effort at best capturing the effect of the legislation.

But all of this may be wrong - I am the tech-ignorant onlooker who is trying to understand what the active people are talking about. I have just used a bunch of terms above, but not because I think I have a clear idea of what they mean. It is just my attempt to grope around and have people correct me. So it is not just "code" - I really don't know whether I have a correct grasp of what tech people mean by "mark-up", "ontology", "pseudocode", "API", "app", "open source", "pull request", "AI" and the rest. I am trying to learn this language by speaking it among native-speakers and waiting for them to correct me. Obviously it is really about understanding concepts, not just grasping a language - I do think for these initiatives to make progress there must be a point at which tech people, policy people and lawyers all have a basic understanding of each others' concepts & language so that we can have fully informed discussions. I think we need more legislative drafters involved, to try to help explain to tech people what the legal concepts mean and how they work - but first we need to be able to explain to non-tech legislative drafters what this is generally about and why they should take an interest (and why they should commit to the effort of understanding the tech concepts and explaining the legal concepts).

So the shorter answer is that when I talk about "code" I am only parroting what I hear, and I am trying to see what other people mean by what they are saying.

2018-11-26

- authoritative v government-endorsed. The distinction is about the way that in Commonwealth countries (and in USA primary legislation), the government may drive the process but the Parliament does the enacting and the court interprets what the Parliament enacted, not what the government thought it was getting the Parliament to enact. So the government's view of what a piece of legislation actually means is just one view, with no more legal force than anyone else's, even if the government sponsored the legislation. By authoritative I mean forming part of the enactment itself, as passed by the Parliament and then independently interpreted by the court (so it does have legal weight) - whereas by government-endorsed I mean something that could be wrong. A schedule is part of the legislation, on an equal footing with the main text, and that is why you need to know the status of the "executable item" in the schedule. It is a principle of drafting that you don't say the same thing twice unless there is a good reason, and that if you do then you make clear what the relative status is of the 2 versions. You cannot be certain no errors will creep in, so you cannot assume the code part will perfectly match the natural language part. So I think you mean the code is subordinate to the natural language - if there is a discrepancy the natural language version is the law and the code is just a faulty explanation.
In theory it could be set up the other way around - the legislature could say the natural language is subordinate to the code. Then if there is a discrepancy the code is the law and should be followed, with the natural language being treated as just a faulty explanation (in the way that Explanatory Notes currently are). But as you say, it is unrealistic to expect the legislature to be able to understand code (or perhaps we should just say it is much more unrealistic than expecting them to understand legislative "natural" language).
Equally in theory the natural language and the code could be given equal status - as is often done with legislation enacted in 2 natural languages (as in Canada or Wales) or in one brave case (EU) 23 languages - but again there are drawbacks (especially if there is complete uncertainty as to which of 2 a court will follow).
Coming back to the coding being subordinate - in Commonwealth drafting

traditionally we would not put merely explanatory material in an enacted piece of legislation at all, not even in a Schedule. It would go in a separate document, like an Explanatory Note, that has no legal status and is not voted on by the Parliament (but can in some circumstances be used by a court as one source of help to resolve ambiguity in the enacted natural language, along with statements in Parliament by the enactment's sponsor and so on). I was just trying to see whether anyone is looking at making the code part of the enactment, as then the legislative drafter is more directly involved, or whether it is running in parallel through the policy development, into the instructions given to the drafter, then as a supplement to the explanations given to the Parliament, and then out the other end as a supplement to the legislation as enacted.
I will post a reply in a minute, but what I think he is talking about is a model in which the code is subordinate to the legislation and is separate from it.

2018-11-27

RE: "A schedule is part of the legislation, on an equal footing with the main text, and that is why you need to know the status of the "executable item" in the schedule."

Yes, that is exactly where Xalgorithms' discussions are focused with some Canadian Members of Parliament. It also means that the computable expression must be readily understandable to non-programmers, and part of the reason we use a tabular declarative programming (not script procedural programming) because human readability is enormously improved. Also, the method of expression must be genuinely cross-platform. JSON seems appropriate. It is not sufficient to say that the computable expressions can run on this or that free/libre/open rules engine, since there are multiple competing free/libre/open rules engines and de facto lock-in to any single solution from here to eternity would be unwise.

RE: " relative status is of the 2 versions. You cannot be certain no errors will creep in, so you cannot assume the code part will perfectly match the natural language part. So I think you mean the code is subordinate to the natural language"

No, the code would have equal legal status as you commented is "done with legislation enacted in 2 natural languages (as in Canada or Wales)". The computable segments ** are translations ** of the natural language texts of those sections. If there is a discrepancy in some aspects between translations, then they need to be fixed. The difference in putting the computable code into a schedule is that it can be fixed more readily, without a Parliamentary amendment. But since it must be a translation, there's no room for the maintainers to diverge from the more anchored natural language statements.

RE: " the government's view of what a piece of legislation actually means is just one view, with no more legal force than anyone else's"

Yes, I understand that.

2018-11-27

Good question.

2018-11-27

Im a long time technologist but who is more interested in the Human side of this. We have already been using Business Architecture or models to help get a common understanding at a human level, in my way of thinking this is coding (logic) the law/drafting. This is then being used as a way to communicate and keep everyone with the same understanding so that the logic doesn't break. Its also now being used to communicate as part of the commentary sections of the Legislation.

2018-11-23

My conversations with the drafters here in NZ is it's envisaged that code would not be made authoritative.

Now what's considered authoritative in software world? Is it "who published it" or how widely tested, shared and open it is.

This is where I believe an *open source platform/structure is going to be crucial to really realise the benefits. Far more so than whether it's "enacted" or not. It also allows the code to be freed from the constraints of the legislation and in a way (funnily enough) be more human. For instance bugs and refinements could be added to the code without it having to proceed back through Parliament. A member of the public being able to make a pull request and have it accepted is going to be vastly more future focused and challenging to the existing processes than tying the code to Parliamentary process.

We also don't lose any of the benefits as outlined in the "Better Rules" report or any of the benefits in having the code published alongside the enacted legislation.

As the ramifications of having authoritative/enacted code is unclear I see no need to rush on this point without a decade or two of experience in drafting legislation/software in combination and again - we can still receive all the benefits.

Another perspective I have is viewing legislation as rules for humans, software as rules for machines. Since machines aren't ever going to be taken to court anytime soon - I don't think enacting their rules is necessary. This seems somewhat flippant but I think it allows those who aren't as deep into the thinking on this topic to more sharply focus on what's really happening here and the ways it can transform how we see our institutions.

2018-11-27

Thanks very much.

If I have understood you correctly, what you are talking about would have the code as subordinate to the enacted natural language text. As a drafter I assumed that would mean we would next have to go for the 2nd prize of a government-endorsed status for the code. But I think the light is dawning on me now. I think what you are talking about is a government-sponsored open site where the code can be worked on by anyone who thinks they can improve it (government officials could also have a go, and their changes would have no more standing than anyone else's). "Improve" there means make the code more accurately (& simply & efficiently) reflect the actual legal meaning of the natural language text (contributors are in effect saying "look, if we changed the code like this it would better reflect the existing legislation"). But presumably there would be a separate element where people can say "look, if we changed the code like this it would not reflect the existing legislation, but it would give better results in the world, so let's lobby Parliament to amend the legislation on these lines". That would also explain how this is different from the way software companies already produce programs based on legislation, while still not forming part of what has been enacted, but in ways nobody else can replicate.

I am still interested in how it works with the drafting work on the natural language version. I think there might be 2 aspects to that - the process that runs from policy formulation through drafting & enactment to implementation and back again, and the possible mark-up/coding applied as the drafter goes along.

Process - If I am getting hold of the right end of the stick, it would mean we could we have the coding used in the development of the policy before it reached the legislative drafter (including public consultation using versions of the code), then the drafts of the legislation would be tested against the coding over the course of producing subsequent drafts - with the coding being refined as the policy is refined when the drafter insists on getting into the detail. The Parliament could then be presented with the legislative text that they are being asked to enact, along with background materials including the traditional natural language explanation of what the government hopes will be the legal effect, plus the coding as another explanatory tool (neither the natural language explanation, nor the code explanation would be enacted as such). After the legislative text was enacted, the coding could then be hosted for people to improve, as described above. Is that about it, somewhere vaguely near, or still thrashing about?

Mark-up - Getting back to the drafter's work, might part of it involve applying some mark-up to the natural language text of the successive drafts? Would the point of the mark-up be to identify key elements in a standard way, such as perhaps the defined terms, and how they relate to a tech "ontology", and the key functions like imposing duties, powers and liabilities, along with if-then(-else) elements, conjunctions & disjunctions, etc? I tend to think in terms of predicate/deontic logic, but I suspect that is not too far off computer logic - I have just seen some work that came out of the Scottish drafting office that suggests this might work. Is the idea then that the coding itself uses that mark-up info to produce a separate coded version of the natural language rules (an "API" for others to produce "apps"/"programs" from)?

As ever, I am only guessing what any of the tech terms in this mean, just to try to ask the questions to get an understanding that makes sense to me as a legislative drafter.

2018-11-27

“.., it would mean we could we have the coding used in the development of the policy before it reached the legislative drafter (including public consultation using versions of the code), then the drafts of the legislation would be tested against the coding over the course of producing subsequent drafts - with the coding being refined as the policy is refined when the drafter insists on getting into the detail.”

Yes. In fact the French incubator, beta.gouv.fr, who deliver OpenFisca are about to test out this process with the French parliament next year. They have already used OpenFisca as a modelling tool for legislation. See

https://www.ipp.eu/en/news/11-oct-evaluating-the-2019-budget/

and the slides:

- https://www.ipp.eu/wp-content/uploads/2018/10/ipp-menages-budget2019.pdf
- https://www.ipp.eu/wp-content/uploads/2018/10/ipp-entreprises-budget2019.pdf

2018-11-27

Yes! that's very satisfying to read for me as communication can be quite the challenging aspect of this and reaching common understanding by having what you've said repeated back is so satisfying. My ego as a programmer wants code to be enacted, but when I try and survey the wider landscape I get concerned that just inserting code into the existing processes will mean that the collective communication tools and procedures of software/internet world won't be so effectively incorporated - being constrained from the get go by what's existing.

It relates to the thought process: if we were to start from scratch and design a parliamentary process in the internet age; what would it look like.

My thinking is keeping the code separate from the enactment process for now would allow the collaborative nature of the internet+software learning to continue to lead the way "by example" for the existing legislative processes. At some point they may become indistinguishable and then we could debate it from a philosophical perspective with everyone having a much deeper understanding of what's happened.

From the drafting perspective (and the markup) - I share the same mirrored interest - it's two complimentary fields looking over a fence at each other thinking "we should talk".

Also keeping an eye on the end goal - for me a big win in the legislation as code space is building comprehension machines. Law is more difficult to comprehend than it should be - I'm up for debating whether enacted parallel code would complicate/compromise the comprehension or not.

One other thought - I think our spoken languages are more "sacred" (finding the right word there is beyond my ability right now while under the timeframe of my children's breakfast) than we necessarily acknowledge. I view code as a language tool rather than a language outright. What I mean by this is I don't often see programmers communicating just in code. It's recognised that the code is for the machines. Does this mean we might enact code just for machines? I'm perhaps more comfortable about enacting pure logic for humans (Vulcans anyone?) - but I'm not confident that programming languages are the best tool for this.

2018-11-27

I still would like to know what "Code" means to you? Is it a specific language? I think it would be interesting to see what people think this actually is. Is is cobol, machine code, or some compiler?

2018-11-28

Thanks - good to get confirmation that we have some shared understanding.

I have started a group of Commonwealth drafters sharing email addresses, and my fellow drafter (from NZ) has just set up a "Legislative implications" thread here. We are going to encourage drafters, via the Commonwealth Association of Legislative Counsel (https://www.calc.ngo/), to join the forum and start contributing. That is limited to Cwlth legal systems (not USA, Napoleonic, Islamic, etc), but I am sure that will raise issues that drafters will want to add a fresh slant to.

Some drafters are already very tech-savvy, but my interest is in being able to explain it to those like me who aren't, and who may be put off by incomprehensible tech detail (while of course relishing incomprehensible legal detail) as much as by hype about replacing legislation (or even all the law) with code and replacing judges (& lawyers & juries) with robots. While avoiding the hype, we still do need to explain how this project represents a new element over & above the way govt depts already commission software to calculate benefits & tax, and how it goes beyond just ensuring legislation doesn't tie people into using paper instead of email - I think I have a clearer idea of that now.

2018-11-28

Hi

The way I think about this is as follows:

Traditionally, we have drafted legislation and then published it so as to make it available for the world to interface with as they wish. In the past, the main users were lawyers and the courts, who are trained to interpret legislation.

Now, with advances in technology, the rules in legislation are integrated into tools - primarily computers - to assist people to do all sorts of things. This may simply be a website that sets out information, or may be a company's business rules which it uses to ensure regulatory compliance, or by a government department to work out people's eligibility for a benefit, or enforcement officers to determine whether people have complied with the rules, and so on. The uses are endless.

The one thing that all of the above have in common, however, is that they operate using software. So, the question is, what is the best way of replicating legislative rules in software? How do we do this easily, and ensure there are not gaps between the legislation and the software?

It seems to me that fundamentally there are 4 options (some of which have variations, but let's keep it simple):

1. We do what we are doing right now, which is simply draft and publish legislation and leave it for the world at large to solve this problem. For a whole bunch of reasons, I don't think this is an acceptable solution. One of the main reasons being that in places like NZ, the UK, and Australia the push by government to have integrated government services framed around citizens' needs requires software versions of legislative rules:
2. draft legislation as we always have, and then run it through some sort of "translation" tool that can take natural language and turn it into software. As far as I know, no such tool exists - although people are working on it:
3. develop a tool that is used by legislative drafters that allows a single input (ie, legislative drafting), but produces 2 outputs (ie, natural language legislation, and an equivalent software version). Again, as far as I know, no such tool exists - although people are working on it:
4. take the Better Rules approach in which we take a different approach to the development of policy and legislation. The primary outputs of the Better Rules approach are concept, decision and flow diagrams. These are in effect a common language that are then used as the instructions used by legislative drafters, business rules folk, and software developers, all of whom can then draft their particular outputs using their current, usual tools and processes. The outputs all need to be checked and validated against each other, but as all parties have contributed to the development and creation of the concept, decision, and flow diagrams, everyone should be speaking the same language - conceptually and literally - from the outset.

For a whole bunch of reasons (which I am happy to expand on in another post), I believe that option 4 is the most viable, the best, and the most forward-looking approach. At least for the foreseeable future.

2018-11-28

RE: #4 " The primary outputs of the Better Rules approach are concept, decision and flow diagrams. These are in effect a common language that are then used as the instructions used by legislative drafters, business rules folk, and software developers, all of whom can then draft their particular outputs using their current, usual tools and processes. "

Agreed (as probably all in this discussion would). Let's now describe the different ways of pursuing #4. But before that, let's jointly assemble some criteria for weighing the strengths and weaknesses of the different approaches to #4.

2018-12-06

I think the different ways of describing the approach to #4 crosses over with the discussions about standards and frameworks. To me, a common language implies a standard framework. Concept models, decision models and flow diagrams can all be expressed in a standardised language. One of the criteria for weighing the strengths of a particular approach would be a published standard. I don't think we have to reinvent a new standard for this approach.

2018-12-06

I agree. But that is where my knowledge ends, and where your expertise (and that of your colleagues) comes into play. I'd really love to see this conversation now get into some details to take this to the next level.

What standards already exist out there?

Is there (almost) universal acceptance of a particular standard, or are there lots to choose from?

If there are lots to choose from, what are the implications of choosing one over another?

The advantages of a standard being chosen and used universally within a single country/jurisdiction are obvious, but are there advantages in the same standard being used by as many countries/jurisdictions as possible?

If a particular standard is used, how will that impact upon teams like yours that already use particular software tools to produce business rules - would it require a change in your practices and tools?

And so on.

2018-12-06

That sounds like a really sensible next step.

Do you have any initial thoughts on the criteria for weighing the strengths and weaknesses of the different approaches to #4? Even if only an initial, rough, first-stab in order to get the conversation and thoughts flowing for everyone else to contribute to?

2018-12-11

Hi,
We ended up going for option 4, at least for our validation. We're not at the point yet where our legislators have the resource to be actively involved in wholesale revamping of our legislation... because Brexit.

We're focusing heavily on the Decision model approach (DMN) and are finding it to be a very useful tool as it presents complex inter-dependencies in an easy to understand, graphical way, but also allows "call outs" for human/committee decisions and other "fluffiness" that legislation can allow wiggle room for.

2018-12-11

Hi

It's interesting to hear that you're using DMN to model your legislative rules. When you said a few days ago that your scope is the entire income tax and social security legislation, are you modelling all of the decisions that you've identified using the DMN standard? Lately I've seen more and more software supporting this standard. And I've started to see it used in other Government departments in New Zealand.

2018-12-11

Hi,

No, we are not modelling all decisions. Our approach is to categorise any business rule into system, process or guidance and only take forward system and process rules for decision modelling.

We've been very impressed with the standard so far. It has catered for our need to ensure traceability whilst presenting the rules in a visual and easy to understand manner. Further down the road it's hoped that these models could be directly called upon by web services.

RULES AS CODE - RESEARCH & DEVELOPMENT

2019-05-14

New thread for discussion related to research and development after the Global Show and Tell.

2019-05-15

Hey all, it was illuminating to see the variety of other approaches; some I had been following, some were new to me. So thank you to the organisers, and thank you to the presenters, for helping bring everybody together and thereby move things forward.

And thank you for the positive feedback on our prototype; it was encouraging. Going forward, I would like to hear specifics about what exactly you found valuable in our demo.

Did you like the isomorphism – the generation of English that matches the legislation, verbatim?
Did you like the automated production of test cases that spanned an exhaustive range of inputs along multiple dimensions?

Did you like the idea of interoperability – reading and writing multiple formats so that multiple environments (e.g. OpenFisca, XAlgorithms, Accord Project) can all play well with one another? LegalRuleML was intended as an interchange language, so maybe we should all explore the possibility of just coordinating via that, if it overlaps sufficiently with our respective semantic models.

If Legalese had some R&D energy we could put toward this, what directions would you suggest we explore, and how should we prioritize the work? What kinds of deliverables would you actually use and incorporate into your own work?

My slides btw are available at https://drive.google.com/open?id=1U4pQFXuVAocbwzF1nPtyhxH2SBza_7nEErLPE2lnAAw –

2019-06-20

You asked "Did you like the isomorphism – the generation of English that matches the legislation, verbatim?" My general answer: Compared to translation and implementation into most programming languages today, of course!
But then we get into some specifics. The principle does raise some interesting questions. What happens if the legislation is riddled with exceptions and what you later call “counterfactual conditionals”? e.g., "A widow or widower of a spouse who died during military service shall be entitled to a benefit if they would have been entitled to that benefit if their spouse were still alive.” (Let’s assume a perfect world where even in spite of the exceptions and counterfactual conditionals, there are no conflicts.)
What if the given piece of legislation has to ‘work’ in concert with rules from some other (possibly many) piece(s) of legislation? It seems to me it might become easy to get ‘lost’ in the logic(?).

SBVR did not assume nicely organized sources of rules (such as legislation, contracts, etc.). It’s certainly nice when you have them. IAC, SBVR features several principles for wrestling with business logic, as follows:
16.3.2 The Accommodation Principle: An element of guidance whose meaning conflicts with some other element(s) of guidance must be taken that way; if no conflict is intended, the element(s) of guidance must be expressed in such a way as to avoid the conflict. Exceptions to elements of guidance must be accommodated explicitly; that is, cases where exceptions to elements of guidance are intended must be worded in such a way to avoid any conflict in the meanings.
16.3.3 The Wholeness Principle: An element of guidance means only exactly what it says, so it must say everything it means. Each element of guidance must be self-contained; that is, no need to appeal to any other element(s) of guidance should ever arise in understanding the full meaning of a given element of guidance.
The issue with exceptions and ‘counterfactual conditionals’ is that you cannot ‘trust’ the underlying statements (the ones at which the exceptions and counterfactual conditionals are aimed) because they do not represent the complete logic. How would you deal with that?
P.S. BTW, SBVR does not standardize any syntax. It is aimed purely at the capture of semantics (from natural language). So, in the literal sense, it does not really make sense to ask, “Does] SBVR offer a compact notation for expressing such "ceteris paribus" type ideas?”.

2019-05-16

Kudos to others who worked to convene the online meet-up.

RE: isomorphism
My colleagues and I like what you had to say about that. Further to that end, we suggest to distinguish between the generic and the particular. This is why we have come to distinguish between "rules as code" (that runs in a particular environment) and "rules as data" (for transmission between rule sender and rule receiver, or amongst anyone, that can be accurately auto-transcribed into any programming code).

RE: automated production of test cases
Just I should mention we're in the midst of updating Xalgo documentation on Github. We have simplified how the MapFilterReduce design pattern would function, amongst some other version updates.

RE: interoperability ... multiple environments (e.g. OpenFisca, XAlgorithms, Accord Project)

In the collaboration towards a general framework for this domain that several expressed interest in (see near the bottom of the meeting notes https://docs.google.com/document/d/1d0lHW87lgTjpHe88WAfBS3g0wJrPH62ySN2RUSmM0MQ/edit) we should eventually be able to come to a shared view of how diverse initiatives relate to each other. For example, there is intentionally no Xalgorithms 'environment' as such. All we're producing is a minimalist means of getting algorithms (which implement rules) communicated from any rule author to any rule user. For example, if NZ expresses Rules as Code in OpenFisca, which is an excellent environment for managing Rules as Code, how exactly would certain of those rules actually get discovered by and delivered to a vender's or payment processor's hosted transaction at the moment of transaction? When NZ signs a new multi-country trade agreement, will that agreement first be coded for OpenFisca, or would it first be expressed in a manner that OpenFisca, and Thomson-Reuters, and Vertex can all automatically injest, and then independently auto-transcribe into the code that they function with? An Internet of Rules is intentionally bounded to not be an environment, but only to support any such full-service environment. Utilities such as Lichen and XalgoAuthor are being provided for convenience and validation, but there's no reason anyone couldn't just write a better app to do what those do in their minimalist way, or to use them as the bases for more advanced derivative applications.

RE: LegalRuleML was intended as an interchange language, so maybe we should all explore the possibility of just coordinating via that, if it overlaps sufficiently with our respective semantic models.

Well, LegalRuleML in JSON for computational performance. I'm presenting in the 31 May RulesML conf call if anyone would like to join that.

RE: What kinds of deliverables would you actually use and incorporate into your own work?

can you please clarify what part if your team's contributions are shared under free/libre/open licences, and what's not licensed that way?

Oh, Thanks for getting to participate in experimenting with writing that rate rebate in Xalgo! Ya. :-)

2019-05-16

All our foundational language infra will be opensource. There is a distant plan to achieve sustainability off tooling or maybe subscriptions for firms to stay up to date on rules as they are promulgated or revised. Or apps and app stores. I have already uploaded to Github most of the stuff I demoed. Before we release the rest of it we want to write some documentation and howtos and case studies so it's more than just an opaque handful of libraries. –

2019-05-16

I would like to chime in … In my recent paper, presented at the Commonwealth Association of Legal Counsel's biennial conference in Zambia, I stress the importance of isomorphism between legal text and code. To isomorphism you can add 'transparency". I argue that an algorithmic representation of law is perfectly possible in a transparent and isomorphic fashion, and propose a language, as a translation layer (something akin to L4, although I haven't seen L4 yet).

The paper is due to be published very soon, and if anyone wants an advance copy, then they are welcome to it. I think you have a copy already.

On the back of the theory of isomorphic algorithmic law (and the language proposed), < removed for commercial reasons > has a technology offering … This is composed of a parser front end, for virtually any structured language (so we could parse L4, for instance), followed by logical mapping. The logical mapping allows the algorithm of a law to be held in a language- and standards- agnostic way. The logical map allows source code to be written out isomorphic to the inputs, using any language of choice …

openfisca, sql, python etc. This is a transparent, secure, cloud-based process using patented technology.

Our research and development pathway is to improve the translation layer … we have several lines of investigation going, and would appreciate any help whatsoever. I am talking to Professor in Florence in the coming months. could be useful, and several of the xml approaches from the show and tell could fit well with our transpiler technology.

We are doing demos in London in late June, and we have mooted a demo in NSW, Australia for later in the year. The technology has been successfully applied in several case studies in New Zealand.

Drop us a line!!

2019-05-16

Hi. Regarding an advance copy of your paper. Yes please!

2019-05-16

Sorry if my newness to this subject means I miss something obvious, but here's my train of thought.
The process of developing policy and writing future legislation could be made to include the generation of models that represent (appropriate pieces of) legislation using standardised natural language. These models could then be hosted by government in a public repository to be accessed by anyone who wanted to transform or translate them into code that could be run on a specific platform. If I'm not missing something important here, there would be considerable value in doing only that. However, if we also had tools that could take those models and automatically transform them into a variety of machine consumable forms that would add huge additional value: i) as part of policy and legislation development (as it would make it easier to run rules on a modelling platform to test the impacts of various policy settings) and ii) for the agencies that needed to implement rules embedded within parts of the legislation inside their own technology environments, and iii) for businesses that wanted to build products and services that incorporated those rules. If I'm not already missing something important in all this, then the things I'm curious about are:
Standards - Understanding. I've read about SVBR and DMN for structured vocabulary and decision modelling. Is L4, which refers to in his presentation, fulfilling a similar function to what these might provide in such an arrangement? Are there other candidates (standards based or otherwise) that are relevant?
Standards - Selection. What standard or standards would be best to act as the 'Rosetta Stone' for vocabulary and rules modelling, assuming that we wanted to be able to publish models that could be ingested either directly, or through transformation tools, by the widest variety of current and future, commercial and open source rules engines, business process management systems, and similar platforms?
Or am I missing something fundamental here and am asking the wrong questions as a result?

2019-05-17

I am way out of my depth for your more detailed questions, but it might help if I chip in on the first element. Yes the idea is that, while the policy & legislative draft are being refined, somebody is also refined a machine-consumable ("coded" for short, in my terms so far) version of the rules. Then when the policy, legislative draft & code are settled, the coding is published for free via a constantly updated API as a govt-endorsed best attempt to capture the rules that are in the enacted legislation.

It is important that this "coded" version is as neutral as practicable so that it can be used by any app/program that anyone cares to hook into it. But I don't think there is any commitment to having it as "models ... using standardised natural language". The IT folk might go for that in the end, or for something more obviously "code" - it makes little odds from my end, as long as it is machine-consumable.

But my impression for now was that any "standardised natural language" version would be as an aid to legislators/officials/lobbyists/public in understanding how the coded version related to the enacted version. So it would be like the similar idea that the govt might also put out its own illustrative app, perhaps designed for particular types of legislation, which the public could use to play with tweaking the variables in the Bill/code. So the consultation would say "our Bill says you get benefit if your income is below £X and the benefit is for 25% of your outgoings above £Y - do you think X, 25% & Y are appropriate - tinker with them in this app to see the results and suggest something different". So at this stage I would still see "standardised natural language" as a prop rather than being the main "coded version".

That may just be my ignorance about what a coded version could look like, and what a model in standardised natural language could be, and what it takes for something to be machine-consumable (and whether there can be an ultimate neutral holy grail). Equally work may already have inspired the NZ-NSW folks to shoot off in directions I haven't grasped yet. As a drafter, I leave all that to the experts, and I just want to make sure it all works OK with the legislative end. "Standardised natural language" just makes me wonder whether that would really be an advance, because of course we draft legislation in what amounts to a relatively standardised natural language already.

2019-05-17

you are pretty much on the money.

< removed for commercial reasons > uses a structured intermediary language … it is akin to L4. If you read my recent CALC paper, you will see the spec for a large part of the structured language.

This serves as a cleaned natural language version of the statute.

The idea is that using workshopping and flow diagrams, concept diagrams and structured language (< removed for commercial reasons > uses LogLaw), you can capture and feed back all at the same time as policy developing and drafting the statute.

everyone gets on the same page using the process tools provided, and at the end of the process you have a logical map and "English" drafting to match. < removed for commercial reasons > uses the logical map to generate source code of any variety (it is a "transpiler".
So, in summary, the policy development and drafting process creates the logical map and the "English" product simultaneously, as part of a team effort. Then < removed for commercial reasons >, which can be used as part of the process itself, "LIVE", like demonstrated, transparently and isomorphically generates computer source code in whatever language desired.

We will be demonstrating the product and process in London in June, and later in the year in New South Wales. The new Zealand government is also very interested in the approach we are taking, and it has been recently trialled here in Wellington. Much of the process work we do is based on "better rules", which ran in Wellington a year or so ago!

Drop us a line if you would like to know more

2019-05-17

Thanks. I think your comments get to the crux of the issue. The following may be correct - or may reflect holes in my understanding of this technical domain - or some misguided assumptions on my part... hence me putting this out there so people can put me straight. My assumption up until now has been that something that is 'coded' to the point of being machine consumable on a specific technology platform (as part of the development of policy and legislative drafts, for example) is only going to be useful to others if they happen to be running the same rules technology as was used for the version we first coded. While it may well be that some number (maybe 10 or 20%?) of organisations might run the same technology, and be able to ingest and run whatever specific 'coded' version Government published, most would have to have the coded version translated into something their (different) systems could actually run. That's one of the things that was so interesting about the translation tooling was talking about in his presentation. I had been assuming up until now that the use of a standards based modelling language that was designed to be programmatically transformable into a 'coded' version suitable for any specific rules platform would be a more useful thing for government to publish than a 'coded' version, as this would allow anyone to download the models and then either import them directly into their platform (if it had built in model to code transformation capability that supported the standard we used), or could use a tool like the one demonstrated with L4 to generate specific versions that would run on their platform, or in the worst case scenario, they could use the models to manually generate code for their platform much more easily than through direct analysis of the legislation. If I am understanding things correctly this is very similar to what says above about the need for a language that acts as a translation layer and what says about distinguishing between "rules as code" (that run in a particular environment) and "rules as data" (for transmission between regulators and other parties, so they can be auto-transcribed into any particular code language). I think it's also similar to how was talking about translations from

L4 into various other 'dialects' in his demo. To express this thinking as a (bad) analogy, if we wanted to help car manufacturers to be able to make carburettors, it would be more useful to publish a description of what a carburettor is and how one works, rather than the designs for a specific example of a carburettor, e.g. for a 2010 VW Golf GTE. The former is general enough to have wide applicability, while the latter is too specific to be useful to other manufacturers without further translation anyway. Again though, I stress that I might be missing something, so am really looking for help in understanding if I've got this wrong. If all the above is right though, it still leaves me searching for the right modelling languages and standards. Hopefully someone ? has some views on the strengths and weaknesses of things like SVBR and DMN

2019-05-20

if I understand you correctly, the vision you're talking about, with clear layering, is very close to the Model-Driven Architecture vision <broken link removed>

From the perspective of a working programmer I wonder: where are the tutorials? Where are the SBVR libraries for languages like Python and Javascript? What does isomorphism from SBVR to English look like?

I have come across some literature describing SBVR in action: http://ceur-ws.org/Vol-1004/paper6.pdf "Interpreting Regulations with SBVR"

L4 may evolve to capture even more legislative expressiveness. For example, in legal writing, we often see counterfactual conditionals. It is my understanding that LegalRuleML and SBVR do not support that kind of reasoning: a language might require some level of homoiconicity to support it. What do I mean by "counterfactual conditional"? "A widow or widower of a spouse who died during military service shall be entitled to a benefit if they would have been entitled to that benefit if their spouse were still alive."

Do LegalRuleML or SBVR offer a compact notation for expressing such "ceteris paribus" type ideas?

2019-05-20

This paper from about 10 years ago surveys some of the existing languages at that time and offers requirements for any interchange language for rules &
norms. http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.377.4658&rep=rep1&type=pdf

2019-05-21

RE: Requirements for Rule Interchange Languages

I looked up the lead author to find out what he's been working on more recently...

http://www.tfgordon.de/publications/
http://www.tfgordon.de/software/

And his colleague
http://www.dougwalton.ca/papers.htm

My colleagues and I will consider req's in relation to the design principles used for Xalgo. A quick first thought is to emphasize the difference between procedural and declarative expression of rules.

https://en.wikipedia.org/wiki/Procedural_programming
https://en.wikipedia.org/wiki/Declarative_programming

Xalgo is declarative.

Here's a good Hackernoon article explaining when procedural is the correct choice over declarative programming.
https://hackernoon.com/when-procedural-is-better-than-declarative-51b24aaaf227

2019-05-18

Thanks
I like the "rules as data" idea, but I can see it still needs to come down to something in a format that can be used by all the different programs - then I hear about everyone claiming their system can translate everyone else's and I wonder whether that means it doesn't matter which brand you publish the "coded/data" version in. Whether that ends up being "logic map", or someone else's structured language, or something specific from OpenFisca or Xalgorithms isn't really my concern. Where I do have a strong interest is in the idea that the policy rules are the common source, and that the "coded/data" version and the English-legislation version just both have to embody the policy rules accurately - but don't need to do it in the same way. I wouldn't want to see our English drafts becoming stilted to fit the computer's approach, and I don't think I need to insist on the "coded/data" version being stilted to fit the English approach - they just need to produce the same effect in a way that is endorsed and transparent enough to give confidence.

To take the NZ Rates Rebate Act example - https://twitter.com/mattwadd/status/1125471545432924160/photo/1 - the English words "exceed" & "reduce" do a lot of work which then doesn't need to be flogged in English (I would just add "if any"), whereas the computer needs a "clip" to say if the results are negative you ignore them. I would want us to be able to keep using clear, concise, unambiguous expressions like "the amount (if any) by which item X] exceeds item Y]", instead of being pushed into saying "the amount obtained by subtracting item Y] from item X], and treating that amount as zero if it is negative" or even "if item X] is greater than item Y] the amount is zero, otherwise the amount is the result of subtracting item Y] from item X]". Both of those last two may reflect what a computer needs to do in order to process our human concept, but a human is going to grasp the first version much quicker. The English legislation version should be expressed to be human-graspable as far as is consistent with being accurate, and the "coded/data" version should be expressed to be computer-graspable again as far as is consistent with being accurate.
The point about the co-drafting is that you are not fixing the English legislation first, with the coding having to trot along afterwards (as happens now). But equally the co-drafting idea

seems to me not to be that the coding is fixed first, with the English trotting along after (let alone that the English text of the legislation is automatically generated from the coding). As I see it they would both be done together and both render the same policy rules with the same effect, but in the separate ways most appropriate to the different needs of legislation & code.
In principle that means for me that the starting point should be that the principles & style of good modern Commonwealth drafting do not necessarily have to change to fit the digitisation. But the continuing improvement of those principles & style may be helped by the lessons learned in the co-drafting, which is a significantly different point. To make legislation more readily consumable by machines, the RaC model seems to focus, not on changing the text of the legislation as such, but on changing the process by which policy is turned into law. So the legislative text would still be produced by drafters (in dialogue with policy staff & now with coders, if no merging of roles) & then enacted by legislators (in Cwlth systems, as opposed to eg USA). But it would be supplemented by a government-endorsed coded/data version, produced by coders (engaged in the same dialogue) - with the policy, coder & drafter staff mutually agreeing that both versions correctly embodied the policy (& so have the same effects).
Currently the drafter & policy officer just have to wrestle unaided with working out whether they are each understanding the other properly and talking about the same thing. With RaC the drafter & policy officer may or may not be able to understand the code/data version, but one of the advantages is that successive iterations of the code/data version will be run through an app/program that will illustrate the effects in user-friendly ways that help the policy officer & drafter (as well as the coder, and then later as well as the consultees, and then the legislators, etc) to ensure they are sharing the same understanding of the effects of what they are doing. For illustrations of how that could be useful see -
https://twitter.com/sminnee/status/1128927772401782784
https://twitter.com/BR3NDA/status/1128944827976962048
https://twitter.com/mengwong/status/1128966742619893760
If that is right then I can watch with interest while others who understand coding sort out between themselves what is the best format in which to have the coded/data version. There seem to be lots of people claiming theirs is the best, and that theirs can translate all the others - I will just hold their coats and wait.

2019-05-20

I could not have said it any better. Your vision accords with mine. I hope to present pretty much the full package exactly along the lines you have outlined (policy to code, tech included,
integrated process, code and English law etc!) in London.

Would love to talk to you about developments in Jersey. It seems to me that you guys have a very good handle on this stuff …. And I'm not saying that just because we align!

I think the integrated process is key to delivering real value to citizens.

Regards

2019-05-20

Spot on - I think you have articulated that perfectly.

You have provided a succinct description of the Better Rules approach, and the issue we are grappling with right now also.

The thing that the Better Rules approach provides that no technology solution can provide is "better rules" - ie, rules that have been worked out and developed to be as fit for purpose as possible. This has to be at the policy development point.

A harder question is how best to take the next step. As you note, the production in one language can't diminish the effectiveness of another language - that won't result in better rules.

Like you, we are now turning to the technologists to determine what the "missing link" is, which is the best process to use, or even if there is a missing link.

One option is that everyone continues to simply use the tools they use right now to produce the products they produce right now. That would certainly work for the legislative drafters amongst us. But having someone sit down and produce software code in a particular software format doesn't seem to get us all of the benefits that we could get. And it raises the questions you have identified - what is the best format to use, or does it not matter because there are translation tools that can go from one format to any other format?

We were discussing this very issue on Friday afternoon and are looking to continue it. So a timely summary.

Cheers

2019-05-20

I'd like to chime in (again) … In the end, from a technology perspective, I think transparency and isomorphism are the evaluative criteria. For me, because those criteria are HUMAN criteria, and can not be proven by an algorithm or a computer, it is the human process that become paramount. Tech is a tool to be integrated into the process. Of course a new technology (whatever it might be) will require a different process. We work differently in NZ now that we use xml. The same will be true with our processes to incorporate RaC into our practices.

I feel quitre confident that the tech is up to scratch. For me it is the integration of the tech that is the big question. It is also a real opportunity. Being able to deep integrate tech into the legislative process has such a myriad of potential benefits!

The specific issue for technologists is really the translation layer. That was identified at the better rules workshop!! It continues to be the focus of development in the tech world.

I have some perspectives on the translation layer ...'insights' slight_smile:

I like the idea of transparenly ''good enough' algorithms as an adjunct to clean modern precision drafting. To the extent that we can up our game in the drafting area to improve that, I think we will up the game in the translation layer area. Bad drafting is hard to translate!!

Anyway. as outlined the high level architecture pretty well. Developing an integrated, transparent, isomorphic process is the way to go.

2019-05-20

Also, 'translate" is perhaps not quite the right word .... I think parallelism is a better paradigm for RaC ...I also think the 'agile' methodology has benefits for the legislative process beyond coding ...

2019-05-23

Useful advice, from over a decade ago:

Will Business-Readable Domain-Specific Languages allow business people to write software rules without involving programmers?
https://www.martinfowler.com/bliki/BusinessReadableDSL.html
He wrote a book on this topic: https://martinfowler.com/books/dsl.html

Should I use a Rules Engine?
https://www.martinfowler.com/bliki/RulesEngine.html

2019-05-31

talks about production rules, but they are not the only kind of rules! See remarks from 4 years ago: http://blog.ruleml.org/post/32629706-the-sad-state-concerning-the-relationships-between-logic-rules-and-logic-programming

2019-06-01

Yes, its's a good comment. Following up on a number of papers et.al. it appears a couple of 'root' references (i.e. that many subsequence authors refer to) on basic rule types are:

Ten Theses on Logic Languages for the Semantic Web
https://www.researchgate.net/publication/220787259_Ten_Theses_on_Logic_Languages_for_the_Semantic_Web

Challenges for Rule Systems on the Web
http://www.cs.nccu.edu.tw/~jong/pub/rule-challenge(RuleML2009).pdf

The part that the free/libre Xalgorithms community is pursuing with generic specs and components that would enable any person or machine to put-find-get algorithms over the Internet holds to the Internet's core principle that complexity belongs at the edges, while the Internet should remain as simple as possible. That's to say, in practice we need to keep the IoR core as simple as possible without being 'simplistic', so that it can be efficiently complementary to any degree of complex rules automation of any type at the edges, where more advanced processors do their magic (Drools, OpenFisca, Machine Process Controllers, etc.) The IoR's Interlibr interpreter executes a limited set of basic functions for data manipulation, but only for the purpose of informing users of test results (so, for example, someone can find out about expected outcomes based on limited data exposed). The gathering of "in effect" and "applicable" rules is all accomplished with metadata and WHEN statements. But to deliver results back to users, we need to categorize those results coherently by rule type (amongst other criteria). So a widely acceptable rule typology is quite important for us.

2019-05-31

The Finnish Tax Administration was recently approached by a post-graduate scholar who intends to do some research in the context of applying a specific DSL.. "editor" to tax legislation in the Finnish language (which is considered to be a bit tricky due to it's "agglutinative" nature >
see http://www.lausti.com/articles/languages/finnishlanguage.htm).

Below some links to the...

- DSL-software itself > https://www.jetbrains.com/mps/
- a national tax legislation domain (Dutch) that has developed it's own DSL "RegelSpraak" (as I understand it) already ten years ago but is now testing the JetBrainsMPS approach in order to make the DSL more human/business readable > http://www.brcommunity.com/articles.php?id=b622 (personally I got aquainted with Blaze in the Netherlands already during the last decade when we in Finland were checking out RuleBurst (a.k.a OPA nowadays, I reckon))

Glad to hear some comments if these (or similar cases/tools) are familiar in any way. Our local research project would anyway not start until late 2020 since the researcher in question is planning on funding his work though the Academy of Finland who is notorious for it's slow funding decision process :-)

2019-09-20

We published a new report this week on an exploration run earlier in the year with a team new to the Better Rules concept.
You can read the introduction and download the pdf here (I have also attached it to this post):
https://serviceinnovationlab.github.io/projects/legislation-as-code/

RESEARCH QUESTIONS

2018-09-12

What questions need to be answered and what questions are people working on?

2018-09-25

How much do we need an ecosystem of private actors offering legislation as code as a service, or training on it; versus having a network of public actors that can maintain common tools and practices?

2018-10-05

Don't we need both? network of public actors agree on common tools and practices. This leads to a common output which can be used by private actors to develop services. Question is where is the hand-over point? Do the public actors stop at the rules and publish rules and private actors develop code from the rules. Or do public actors as far as producing the code?

2018-11-10

I'll try to be clearer: who else than public actors need legislation as code? Thus, why not let public actors create and maintain the tools to expose it? Just like most tools for publishing legislation as legalese that I know of are currently homegrown by governments. The main distinction between code and legalese is the necessary capital investment, and I'm wondering if this capital should come as human capital shared across governments, or as financial capital provided by private actors (and indirectly by public actors paying them for it).

Now, in any case I don't think the solution where “a network of public actors agree on common tools and practices which] can be used by private actors to develop services” is viable. Tech (web, hardware, interop) standards teach us that standardisation makes sense when implementers have a say. Public actors agreeing on stuff which then has to be implemented by private actors without an explicit feedback loop means disconnection between theory and reality and increasing discrepancies between governments.

2018-11-07

Do we have any jurisprudence nerds who are interested in the nature of legal rules? And in particular the extent to which it is desirable for rules to leave uncertainty in outcome to allow for future applicability? I have a feeling that an examination of when this might be desirable will help us to understand better which rules (and which types of rules) are most suited to algorithmic drafting in the legislation.
I'd be grateful to hear from anyone interested in this point.

2018-11-08

Its a good question for sure. There are many types of Rules other than calculation ones. Inference, restriction, behavioural to name a few. We also come across the discussion about Principle based regulations. This is where using something like visual concept models helps to understand where we are purposefully creating the rule types you describe. The question though is how would you make them logical enough for a human to understand and be able to have say an AI based system pick it up. I will talk to some of our more legal minded people about this though

2018-11-09

RE: "the nature of legal rules"
There are several rule taxonomies, which differ in terms of what they are distinguishing. In collaboration with, our lead technical designer/developer on the team to create an Internet of Rules, we have drawn upon various rule taxonomies to group rules for automation purposes by the following functional types, based on the required computational design:

Event/Condition/Action (ECA): Information about an event is provided, and if certain conditions are met, a specific action is taken.

Action-by-Inference/Classification: Information provided is classified, and a pattern-matching algorithm builds a network of related inferences;

Workflow: Information provided is processed according to a sequential procedure chosen from a library of procedures;

Transcription Map: Information provided is transcribed to another representation chosen from a library of data mapping tables.

RE: "In particular the extent to which it is desirable for rules to leave uncertainty in outcome to allow for future applicability?"

In our design we address that in terms of what the computers are permitted to perform, versus what results will the computers provide to people to perform. When it comes to many domains of ruie application, it seems best to empower people with comupters, rather than pre-empt or substitute for people. Humans are pretty good at nuance.

2018-11-09

I very much agree on the last point - isolate what needs humans and build the computerisation round it, without seeing the human input as a failure of the computerisation. I also suspect that, if that is accepted (and the policy developer & legislative drafter of the future will work out which bits should be human), then the original question melts away, to be replaced with approach about categorising by how they should be computerised rather than by their legal status or an assumed rule about what should be left to humans.

2018-11-09

Thanks. Another way to put it: "Let's keep all the interesting stuff for us humans, and give all the boring stuff to the computers. They don't mind. (Really!)

2018-11-08

I am trying to get other legislative drafters interested in contributing to these discussions. There are academic lawyers interested in jurisprudence as legal philosophy, but they are more commonly focused on what counts as a legal rule (and what the effect is on that of leaving gaps), rather than whether & when it is good policy to leave gaps. But legislative drafters are lawyers with practical experience of dealing with different ways of crafting legal rules and handling uncertainty. Sometimes our policy clients want the uncertainty and sometimes they don't. To some degree there will always be an element of uncertainty in any legal rule - someone has to decide how the particular facts of an actual event fit the categories in the rule - even with legislation-as-code somebody presumably has to enter the data to work out the result for the particular case, and it is hard to see how that could ever completely remove the need for a human using natural language to decide "is this one a something or a something-else". To a drafter it is not necessarily so much about leaving "uncertainty in outcome to allow for future applicability", as about where you place that decision function. The kinds of legislation so far taken as most suitable for computation have been those about taxes & benefits, where you plug in data and produce an amount of money owed/due. In that work people have talked about the difficulties of defining income (gross, net, etc) - but at a deeper level the drafter will be looking at who has the say on the result of their applying that definition to the particular facts. If you leave it objective/unstated in legislation then that means it is ultimately down to a court (who will listen to an expert in that factual field, but not be bound by their evidence). There are several other ways of doing it though - you can say it is whatever the employed person says it is (where an honesty system works), or whatever the tax/benefits authorities have on their records for last year's assessment/application (which they then control), or whatever the tax/benefit authorities reasonably believe it is. That last one can be pitched in different ways to attract different levels on which a court can intervene to query that belief - not at all; only if there is a clear legal error; only if the belief is so unreasonable that no reasonable authority could have held it; only on the basis of the contributing facts as put in evidence before the decision; or in other ways down to - as a full objective re-hearing of all the original evidence plus any new evidence that anyone has, and deciding on the merits (in which case it is really just the objective, down-to-the-court test, with an initial stage at which the authority makes their best guess at what the court would say, and nobody asks the court as long as nobody is unhappy enough to challenge it - which is really the same as what happens if you leave it objective when it depends in practice on the authority deciding to act). I suspect a key area for development will be how far we can identify the computable and the uncomputable elements (without regarding them as a sign of failure), and work out ways to structure the computable elements around the uncomputables in ways that mean we can be happy with both. I don't know if that is the sort of thing you were looking for, but I hope it is useful.

2018-11-09

I am looking at making a ML application for calculating duty based on these rules, https://www.customs.govt.nz/personal/duty-and-gst/duty-and-allowances/

The problem I discovered when looking at these rules is that any letter would receive GST because of the line, "any international transport and insurance costs". There seems to be no exceptions list and information would be considered goods, even if the information was free. I wonder how many other cases where there are these kinds of rules exist with similar errors.

So I was looking at ML task to do but I figured that a Duty calculator helper would be easier to do as it's an 'if then' conditional app.
It would test if the goods are over $400 and then show the duty/gst as well as links to getting a customs number so the duty/gst can be paid.
It would also have tick boxes, for things like tobacco and alcohol and other exemption calculations. This would be a practice for a legal hack even like the next https://legalhackers.nz/betterruleshack/#about

2018-11-09

I don't know the NZ customs legislation, but do you mean you were looking at the rules on the linked webpage, or in the actual legislation? The link is to an explanatory web page that has been abbreviated to be used by humans exercising human common sense and to cover the majority of questions that most users would ask. I suspect that the actual legislation will be much more complex than this webpage, because it will go into much greater detail. I would hope the drafter of the legislation has not made the error you mention, but I have no idea from looking at this webpage. I support webpages simplifying legislation for users, but I do think they should include a link to the published legislation, so that someone with a more detailed question can at least see that there are more detailed actual rules behind the explanation.

But even if you look at the legislation itself, and find the error, it might be more apparent than real, as the answer might be in an interpretation provision elsewhere (in that legislation, its parent legislation, case-law on that legislation, an Interpretation Act, that country's case-law on statutory interpretation, and so on). Legislative drafters' daily work involves balancing the need for detail and certainty against the need for clarity, brevity and ease of use, and all of that against the realities of what the particular legislative project can and cannot be expected to improve in the existing legal background that the new legislation will be fitted in to.

We need to mesh coding, policy development and legislative drafting into the production of new laws, with an understanding of each contributed by specialists in all these fields. That is partly so that errors can be spotted earlier, but it seems to me the much more significant opportunities for improvement are not about errors, and are instead about taking in the fresh view of the coding community, coupled with the policy drive to produce more digital-friendly policy & legislation. I would hope those might mean that we can make the improvements in standardisation and interpretation that have eluded us so far for decades (since the English-Welsh Law Commission was set up in the 1960s) because of historical and political obstacles (but still without treating legitimate needs for human involvement as an obstacle).

2018-11-09

True, I will have to dive into that, thanks :D

2018-11-09

Thank you for the discussion above. We have done a short practical experiment led by NZ lab where they built a demonstrator rule engine that gets one input and shows the results in 3 countries - NZ, Uruguay and Israel. It was done by OpenFisca tools.
Regarding the discussion above, I can relate to some aspects.
First - hierarchy. Laws start from constitution (or "Primary Rules in Israel since we have not constitution); laws; regulations; practices. So it can be one way to look at the "legal picture";
Second - "re-use" of pieces of legislation, by a reference to them in other laws. Here we can use a "where-used" sort of function to be effective and to dispose of law-redundancy;
Third - Change Management which is based on the above;
Fourth - actual use-cases, practiced in court, tested against the "norm".
We have just started this journey and try to involve legal advisers from the government, the Ministry of Justice and the Parliament, to get them familiar with these tools and hear their needs and concerns
I have had two specific experience that I was involved in creating a legal act. One was on border control, entry and exit from the country` and the second one the Biometric database related to issuance of eID cards. In both cases, I prepared a very detailed process analysis including charts in a graphical form, first reviewed and confirmed by the business oriented people, that later were presented to the legal people so they could understand the different use cases. So the first part of the process should be definitely a clear vision of the objectives and the businesses case, a clear deployment of it to the legal people, and a flexible tool that could reflect the legal input and change the model according to legal needs, policy needs and political agreements. It is important that at the end of the process, the business model, the legal model and the IT model will all be aligned to each other, for the continuous future progress.
Regarding uncertainty - I think it relates also to the nature of the rule itself and to its place in the hierarchy. Naturally, the higher you go you must leave more space and flexibility. The lower you go you can be more precise and definitive.
I believe that the introduction of AI algorithms will mandate the use of a computerized tool to assess new decisions that are to be handed over to machines and robots, so Legislation as a code will be essential for any community pushing on the "AI-way".
Hope this contributes to the discussion and to the question raised.

2018-11-29

I'm wondering if people have any views on *why* we should be rendering laws and legal rules as decision trees/code/programming language. I acknowledge that the transition is underway and essentially inevitable, but in our rush to keep pace with the technological developments driving this change I think it's important that we don't forget to interrogate/explore the principles that legitimise the change in the first place.
Is it purely a matter of economics and the money we can save money by automating decisions that are made by slower and more costly humans, or are some things worth spending money on? Do people think there is a cost to re-couching laws in terms that are easier for a machine to resolve but eliminate the 'human dimension' from legal and regulatory decision making, or can we make systems that are complex enough to account for the exceptions to 'the rules'?

2018-11-30

Our work in the Better for Business programme is about making it easier for businesses to deal with government. Areas of interest for us are tax, licensing/permitting, importing/exporting, employment etc. In short, a

regulatory and compliance environment. Businesses in dealing with government are looking for certainty, transparency and fairness. In general they don't like ambiguity, want to deal with government regulations as fast and easy as possible and focus on running their business. In this context the Better Rules concept makes a lot of sense. It is even not driven by technology but emerging technology is an opportunity to make it easier to deal with complex regulation. In NZ, and most other countries, the majority of businesses are SMEs - small - and they struggle to keep up with new regulation. A business doesn't want to go to court, a tribunal or consultant to discuss how to interpret the law. We are conscious of the need for flexibility in policies - but this shouldn't result in a interaction with government that is difficult or impractical.
Secondly, the interaction between businesses and government is indirect and their are a lot of intermediaries like tax agents, accountants, export agents and consultants. More and more their business systems are linked and integrated. Government can not stay behind.

2018-12-03

Because making legislation as code is what enabled https://mes-aides.gouv.fr to help 12k individuals a day get access to benefits they were entitled to but did not know about.

I don't care about tech. I care about helping people activate their rights. And I believe legislation as code is an enabler for that, by massively simplifying the creation of software that can act as highly qualified advisers.

2018-12-03

there are two parts to this discussion. Human and Machine. I prefer to understand this from a Human aspect. The Regulations, Legislation, Policy that we write (We have control over all we produce) have to be implemented, understood etc. Improving how we make these join up where they need to, not create conflicting statements and become citizen centric rather than legislator/producer centric is the future we need to continue to strive towards. People in trying to navigate this information are less likely to read through pages and pages of information, either a search or with say Expert systems (AI) they are having the answers delivered to them. Add in the transition to voice based navigation, creating some "standards" or improving the logic of the law will end up in a better experience for those who it was written for.

2018-11-29

I see it as a tool that would aid the law making in that new laws would be tested against an algorithm to see if it is in conflict with existing laws. A way to find issues before they are stamped into the legislation.

2018-11-30

I am currently working on the issue of computing salary for employees in a specific government-type environment. I can see the value of having a model of the salary for different groups, that once defined and tested, could be integrated into many different salary software modules or packages so that they will all compute the salary according to the same rules and could be updated in an easier way.