University of New South Wales Law Research Series

STRENGTHENING DEVELOPMENT OF RULES AS CODE: SUBMISSION TO THE OECD'S OPSI ON CRACKING THE CODE

GRAHAM GREENLEAF, ANDREW MOWBRAY AND PHILIP CHUNG

[2021] *UNSWLRS* 4

UNSW Law
UNSW Sydney NSW 2052 Australia

E: unswlrs@unsw.edu.au
W: http://www.law.unsw.edu.au/research/faculty-publications
AustLII: http://www.austlii.edu.au/au/journals/UNSWLRS/
SSRN: http://www.ssrn.com/link/UNSW-LEG.html

Australasian Legal Information Institute
A joint facility of UTS and UNSW Faculties of Law
http://www.austlii.edu.au/

Strengthening development of Rules as Code:
Submission to the OECD's OPSI on Cracking the Code

Graham Greenleaf,* Andrew Mowbray & Philip Chung*****
Australasian Legal Information Institute (AustLII)
23 June 2020

Contents

1 Introduction **2**

2 Choices and challenges: Technologies for RaC **2**
2.1 Coding new or old rules? 2
2.2 Declarative or imperative programming languages? 3
2.3 Automated conversion from law to code? 3
2.4 Choice of software 4

3 Operationalising RaC: Participants, priorities and principles **5**
3.1 Who should produce RaC? 5
3.2 What area of law should be coded? 7
3.3 Principles for a successful approach 7

4 Conclusions: Summary of recommendations **10**

* *Professor of Law & Information Systems, UNSW Australia, Co-founder and Senior Researcher, AustLII.*
** *Professor of Law & Information Technology, UTS Australia, Co-Director, AustLII*
*** *Associate Professor, Faculty of Law, UNSW Australia, Executive Director, AustLII*

1 Introduction

The draft working paper, *Cracking the Code: Rulemaking for humans and machines* (OECD, May 2020), published by the OECD's Observatory of Public Sector Innovation (OPSI) within the Open and Innovative Government Division of the Public Governance Directorate, is intended to act as a resource for public servants across OECD Member States, to help them understand and engage with the concept of 'Rules as Code' (RAC) and its implications. OPSI seeks feedback on the draft.[1]

The 'ElectKB' knowledge-base and application developed using the DataLex software by the Australasian Legal Information Institute (AustLII) is cited by OPSI in its draft working paper as an exemplar of a RAC application developed outside of government.[2] AustLII is very pleased that the OECD finds its work valuable, and wishes to respond to the invitation to comment on the draft of *Cracking the Code*.

The Australasian Legal Information Institute (AustLII)[3] is a joint facility of two Australian law Faculties (University of New South Wales and University of Technology Sydney) which provides non-profit free public access to all significant types of Australasian legal information. It is the largest online provider of access to legal information in Australia, with over 700,000 page accesses per day, and is one of world's largest providers of free access and a founder of the international Free Access to Law Movement (FALM). AustLII relies in large part on its own technological developments (search engine, mark-up tools, citators, inferencing tools), including considerable use of AI techniques. AustLII commenced in 1995 and so has over a quarter-century of experience in legal informatics.

Our comments follow the order of presentation in *Cracking the Code*. Unless specified otherwise, page numbers in parentheses such as (pp. 20-21), and boxed paragraph references such as [8.1] refer to *Cracking the Code*.

2 Choices and challenges: Technologies for RaC

Chapter 6 of *Cracking the Code* sets out a number of choices that those setting out on RaC projects need to make.

2.1 Coding new or old rules?

We agree that there is a choice between (on the one hand) employing RaC to convert existing legislation into code, both for third party purposes of testing and ensuring regulatory compliance (a common meaning of RegTech) and government purposes of administering laws, and (on the other hand)

[1] OPSI 'Seeking your feedback on draft Rules as Code primer' 27 May 2020 https://oecd-opsi.org/seeking-your-feedback-on-draft-rules-as-code-primer/

[2] *Cracking the Code*, Box 4.3 and Figure 4.1, pp. 52-53; drawn from Mowbray, A, Chung, P. and G. Greenleaf (2019b), "Utilising AI in the legal assistance sector", *LegalAIIA Workshop*, Canada, 17 June 2019 (now published as Mowbray, Chung and Greenleaf 'Utilising AI in the legal assistance sector – Testing a role for legal information institutes' *Computer Law & Security Review* 38 (2020)).

[3] AustLII http://www.austlii.edu.au

changing the drafting process so that both conventional (human-oriented) and machine-processable versions of new legislation are produced together, by RaC processes (p. 81).

Our view is that both approaches should be pursued. Whichever approach is taken, the resources required will be significant, at least until RaC processes are far better developed and tested, so it is always going to be a question of where will be the priority for public funds to be expended in order to obtain the greatest public benefit.

2.2 Declarative or imperative programming languages?

While the primer is correct to emphasize the difference between declarative and imperative (or procedural) programming (p. 83), we don't think its conclusions are strong enough. For the reasons stated by Morris,[4] declarative programming has considerable advantages in relation to legislation (or other rules), including knowledge-bases that can more easily mirror the structure of the rules they model (isomorphism), and automated generation of explanations.

Our view is that any program which is useful for RaC development should allow both declarative and imperative programming, which is not difficult to achieve (DataLex knowledge-bases allow both). The most important thing to stress, however, is that it is essential to provide for a rich set of declarative programming capabilities,[5] and that it is not possible to rely on imperative (or procedural) programming alone.

2.3 Automated conversion from law to code?

As the primer correctly points out, there is as yet no 'technology solution that enables the automatic conversion of human readable (natural language) text into machine-consumable code' (p. 86). We do not know of any programs that will take a whole piece of legislation (Act or regulation), or a significant part thereof, and produce from the 'raw' legislation a piece of code which will run, with acceptable accuracy.

However, our view is that to make advances toward automating the conversion from legislation to code is essential for progress, particularly in relation to 'scaling up' applications from being small demonstrations to become 'real world' tools. This is another form of the 'knowledge acquisition bottleneck' which has always been one of the main impediments to progress in 'AI & law'. We suggest three things that could be valuable steps toward this goal:

(i) There may be some parts of legislation which have a reasonably consistent structure across different items of legislation, so that a

[4] Morris, Jason, 'Spreadsheets for Legal Reasoning: The Continued Promise of Declarative Logic Programming in Law' (Dissertation, April 15, 2020) https://ssrn.com/abstract=3577239

[5] See, for example, Mowbray, A and Greenleaf, G and Chung, P, AustLII's DataLex Developer's Manual (1st Edition, June 2019) http://austlii.community/wiki/DataLex/DataLexDeveloperSManual (wiki) or https://ssrn.com/abstract=3408555 (PDF), particularly Chapters 2-4

program could recognise them and convert them into components of a knowledge-base. An important example, which the DataLex Project is exploring, is definitions, where a very high percentage of definitions adhere to a small number of archetypes. The advantage of conversion of definitions is one defined term may occur in multiple sections.

(ii) Even where conversion of legislative texts directly to functioning code does not work, it may still be worthwhile to develop programs which can recognise certain elements of a section which can be converted into formal expressions in a knowledge-base, but falling short of full rules. It will then be left to human editors to complete the conversion of the section into rules. The program here could be seen as creating an incomplete first draft of a knowledge-base, for human expert completion.[6]

(iii) In other areas of AI, such as image recognition and NLP, there are annual competitions to successfully recognise or translate test images or texts. The OECD (or some other neutral organisation) could sponsor competitions for effectiveness of conversion of legislative texts to functioning code. Public comparative testing of tools on the market or under development would benefit all potential users of such tools. At present, public examples of anything that actually runs are few and far between.[7]

These are not tasks best suited for government agencies, or law firms, who are developing RaC examples, but are better suited for research organisations or universities. However, in our view the primer should not neglect to mention that this is a valuable part of the work that needs to be done to make RaC a reality.

2.4 Choice of software

Morris is no doubt correct that no one product currently 'has it all' (p. 86) for the purpose of converting legislation to code (let alone drafting code and legislation simultaneously). We are pleased to see that in his dissertation he concluded that the DataLex software had more virtues than any of the other programs being compared.[8]

The primer gives a few paragraphs of details of five programs that could be used for RaC projects, and in a few cases have been used (pp. 86-88). The DataLex software has not been included here, though it is mentioned elsewhere in the primer, and this unfortunately could give the impression that it is not suitable for RaC development. A summary is as follows:

[6] Data61's 'Parse IT' might do some of this: CSIRO's Data61 (2019), "Case study on CSIRO's Data61, Australia: Contribution to the OECD TIP Digital and Open Innovation project", p.31 https://www.innovationpolicyplatform.org/system/files/imce/Data61_Australia_TIPDigitalCaseStudy2019%20(2)/index.pdf

[7] AustLII's DataLex software can be tested by anyone who wishes to write and run their own test knowledge-base. Small completed examples (used mainly for teaching) are also provided for testing: DataLex Community pages http://austlii.community/foswiki/DataLex/WebHome

[8] Morris, Dissertation (cited above), Table p. 70.

> **DataLex**: The DataLex software, developed by the university-based Australasian Legal Information Institute (AustLII), primarily carries out rule-based reasoning, using backward-chaining and forward-chaining rules expressed in declarative form. These are supplemented by procedural (imperative) code, where procedural steps in reasoning are needed. Example-based (or 'case-based') reasoning may also be used where appropriate. A quasi-natural-language knowledge-base syntax (ie one resembling English) is used to declare rules (and examples). Knowledge-bases use propositional logic. There is no separate coding of questions, explanations and reports, because they are all generated automatically from the declared rules, in dialogues generated 'on the fly' when the system is in operation. This default operation can be customised where special circumstances require. Isomorphic (one-to-one) relationships between the knowledge-base and legislation is facilitated, and this assists in debugging and updating. DataLex aims to allow easier development, debugging and maintenance by domain experts (lawyers), without involvement by software experts or 'knowledge engineers'. Applications which operate within the AustLII Communities environment have automated hypertext links to Australian legislation and cases, from dialogues and reports, and these may also be customized. The software and all tools and manuals are freely accessible for developing test applications, use is free for non-commercial applications, and licensed for other purposes.[9] Extensive documentation is available.[10]

No doubt there is other software available which can also produce good outcomes. We can only speak for the advantages of the DataLex software.

3 Operationalising RaC: Participants, priorities and principles

Chapter 8 of *Cracking the Code* makes high-level recommendations to government. We generally agree with them, but we suggest improvements that can be made.

3.1 Who should produce RaC?

AustLII has previously (2018[11] and earlier papers cited therein) advocated principles for development of legislation-based apps, primarily from the

[9] DataLex Community web pages http://austlii.community/foswiki/DataLex/WebHome

[10] *AustLII's DataLex Developer's Manual* (1st Edition, June 2019) (wiki); 'Building sustainable free legal advisory systems: Experiences from the history of AI & law' (2018) 34(1) *Computer Law & Security Review*; 'Utilising AI in the legal assistance sector - Testing a role for Legal Information Institutes' (2019) *IAAI Workshop*, ICAIL Conference, Montreal, June 2019

[11] Greenleaf, G, Mowbray, A, and Chung, P, 'Building Sustainable Free Legal Advisory Systems: Experiences from the History of AI & Law' (2018) 34(1) *Computer Law & Security Review*; https://ssrn.com/abstract=3021452

perspective of providers of free legal advice, such as community legal centres, legal aid bodies, and the like. Our approach is that it is vital that such non-profit Non-Government Organisations are able to participate in the development of Rules as Code, to utilise it for the benefit of their clients, and to modify government-developed RaC to develop their own apps which meet the needs of their clients.

Cracking the Code makes a similar point when it notes that 'the government may not need to create an additional information service if there already exists a similar, well-known and accurate service that is privately provided. Making government rules available for consumption and use may therefore improve the accuracy and quality of information offered through private websites and applications'(p. 52). However, it was only referring to apps that 'administrators, local governments and businesses' might build, and we stress that the legal assistance sector must not be left out of this list.

In [8.1] the possibilities, advantages and risks, of 'private sector firms' contributing to the development of RaC is discussed, but this has the same limitation as above that only the profit-making private sector seems to be considered, with all the risks of lock-ins and monopolisation that involves. This ignores two important facts about the provision of legal information and legal information technology research in OECD Member States:

(i) In common law OECD countries free-access non-profit providers of access to legal information are among the largest providers of legal information to the public (including the legal profession, and the University sector), usually by organisations called 'legal information institutes' (LIIs), who are often but not always university-based. This significant free-access provision is found in Australia (AustLII), Canada (CanLII), the UK and Ireland (BAILII), New Zealand (NZLII), South Africa (SAFLII), all Pacific Island countries (PacLII), the USA (the LII (Cornell)), and Hong Kong (HKLII).

(ii) In civil law OECD countries the provision of free access legal information by LIIs has not developed, but similar bodies, mainly university-based legal informatics research institutes (who are also members of the Free Access to Law Movement[12]) have a very active role in developing legal technologies that are utilised by national governments in Europe and elsewhere, and by the European Commission, in their own provision of free and high quality public access to legal information.

We therefore recommend that these non-profit Civil Society bodies – the legal assistance sector, legal information institutes, and university research institutes – should be explicitly included in the bodies that the OECD envisages as being involved in RaC. These organisations may well be much more sympathetic to the principles such as open and transparent rules that *Cracking the Code* advocates (discussed below) than is the commercial sector.

[12] Free Access to Law Movement (FALM) website: http://www.falm.info/

3.2 What area of law should be coded?

Although [8.2] presents this as 'What rules should be coded?', that is not in fact the correct question, which is 'what area of law should be prioritised for coding?' All legislation and regulation comes with the possibility that, under some circumstances, it may require judicial or administrative interpretation before its meaning, and thus its application, is certain.

Cracking the Code impliedly accepts this, advocating the prioritising of coding prescriptive rules: 'Requiring little or no discretion, prescriptive rules leave little or no ambiguity about the course of action that must be taken.' This leads to the suggestion that civil law systems may be more amendable to RaC than common law systems (p. 106). We disagree, both because it exaggerates the difference between civil law and common law systems, and also because it is possible to build systems to assist user to make interpretative decisions in the relatively rare instances they are required. Most of the DataLex Project's approach concerns how to facilitate such 'decision support' systems, rather than 'robot lawyers'.[13]

This does not mean we disagree that 'Rules that would be valuable if codified… are rules that are likely used repeatedly and by multiple parties.' These pragmatic factors indicate which rules it is sensible to prioritise for codification. To such pragmatic factors, represented in Figure 8.2, we recommend addition of other factors such as aiming to code complexity, situations where numerous factors (often repetitively similar but not identical to each other) must be taken into account in order to reach a decision. Maxims such as 'complexity is the boring part of expertise', and 'code once, use often' help indicate why it is worth aiming to code complexity.

3.3 Principles for a successful approach

We endorse the six principles put forward in the draft *Cracking the Code* at [8.3], most of which are similar to the principles we have advocated previously. We make the following comments on these principles:

(i) **Transparency** – If RaC is to be 'transparent for end-users and citizens' this implies that that it must be understandable by those audiences, and not only by technical experts. The best way to achieve this essential goal is to take the approach used by the DataLex software, where the knowledge-base comprising the rules is written using a 'quasi-natural-language' knowledge representation, such as the rather formal, but understandable, English-like syntax in which knowledge-bases used by the DataLex software are written.

(ii) **Traceability** – 'Traceability', in our understanding, is somewhat more strict a requirement than is suggested. It is necessary to include in these principles that each coded representation of legal requirements can be traced back to the legal sources on which it is based (most commonly, legislative provisions, but also case law or policy decisions). We would

[13] Greenleaf, G, Mowbray, A, and Chung, P, 'Building Sustainable Free Legal Advisory Systems: Experiences from the History of AI & Law' (2018) 34(1) *Computer Law & Security Review*; https://ssrn.com/abstract=3021452, parts [3.9]-[3.14].

say that traceability *requires* that 'the coded rules isomorphically reflect the original rules'. This then links back to the concept of transparency, in that the main purpose of traceability is that the coded rules can be audited by third party experts (not the authors of the code), that they are able to be traced back to the 'original rules' (in legislation or elsewhere). Such 'auditability' could be considered as a separate essential principle , but it should at least be explained as part of 'traceability'. Our view is that a 'quasi-natural-language' knowledge representation (as discussed above) is the best way to ensure that auditors are able to undertake this.

(iii) **Accountability** – 'Authoritative' and 'trusted' are indeed related, but they are not the same thing. Trust in coded rules can also arise from the assurances of third party auditors, as discussed above under 'traceability', not only from government assurances, and it is important that this type of 'accountabilty' be included in a full description. However, it is also correct that it is desirable that an official publisher of coded rules should accept 'accountability' for them by making them 'authoritative', which can be by a legislative act, or by a publicly stated policy which has administrative law consequences. The final sentence is ambiguous and potentially misleading: 'standing by the rules if errors are made' is indeed required in the sense that officials responsible for coded rules must accept liability for the consequences of errors they have made. But it can be read as saying they should insist on adhering to coded rules that they have made authoritative, even once they know that they include errors which should be corrected (as recognised under 'appealability').

(iv) **Appealability** – The principle which is hear described as 'appealability' is, we suggest, a somewhat broader principle (like 'appropriate application'), of which 'appealability' is a key part. However, the onus of appealing is placed on the individual who is adversely affected, and who may not have the skills or resources required to exercise a right to appeal. In some situations, allowing automated decision-making at all can be disproportionate, potentially arbitrary and with too high a risk of error to be justifiable, which rights of appeal cannot overcome. For example, the Australian 'Robo-debt' fiasco could not be legitimated by provision of a right of appeal, because the automated decision-making involved was inherently erroneous and arbitrary, and this was obvious to anyone who thought about the method used. It exemplifies the dangers involved here because the insistence upon implemented an inappropriate automated decision mechanism will cost the Australian government an estimated AU$1 Billion in repayments, costs and compensation.[14] Because many countries do not even have guaranteed rights of appeal, this 'appropriate application' principle needs to be very strongly stated. Also as we understand it, GDPR Article 22(1)[15] does not forbid fully automated

[14]Rob Harris 'Robo-debt farce needs inquiry: Labor – ALP will campaign for a royal commission into government's controversial scheme' Sydney Morning Herald June 23, 2020 http://www.smh.com.au/federal-politics/political-news/labor-proposes-royal-commission-into-robo-debt-scheme-20200622-p554xh.html?btis

[15]GDPR 22(1): 'The data subject shall have the right not to be subject to a decision based

decision-making, but only requires *potential* human involvement (as a matter of right), which a right of appeal will often satisfy. But that is not enough.

(v) **Availability** and **Interoperability** – We agree that official coded rules should 'enable their consumption by third parties'. We also agree that this does not necessarily mean that they should be open source (for example, various Creative Commons licences might be more appropriate), and that they should be made available in formats which are open and interoperable. Once again, our view is that a 'quasi-natural-language' knowledge representation, such as used by DataLex, is the best way to ensure that auditors are able to undertake this.

(vi) **Security** – The need for governments to 'secure rules from cyber threats', once they become available to the public, will often be able to be achieved by ensuring that the rules are digitally signed, so that they can be readily checked that they have not been altered. This will be more difficult to achieve when the 'official' rules are incorporated in broader rule-sets by third parties.

We also wish to propose a number of further principles that are relevant to the development of Rule as Code, both by governments, and by third party developers such as legal assistance providers or law firms.

(vii) **Sustainability** - To assist in minimising the 'knowledge acquisition bottleneck', an important principle is to choose programs for knowledge-base development which maximise the ability of organisations to write, understand and update knowledge-bases from their own legal and administrative staff members, with minimal reliance on computing staff.[16] The ability of legal and policy staff to 'understand basic ideas associated with coding' (p. 110) depends to a great extent on the choice of software which allows development of knowledge-bases which are something close to natural languages. It is not just that 'legislative drafters and coders may need to work in tandem while writing new rules' (p. 111), but rather that a separate role for such 'knowledge engineers' (as they were once known) should and can be minimised.

(viii) **Access to legal sources** – In most legal applications, encoded rules cannot by themselves deal with every situation that may arise in practice, so interpretation by users of the terminology used in rules (or in questions they generate) is necessary. Where possible, rules should be linked to the legal sources on which the rules are based, to facilitate user interpretation.[17]

solely on automated processing, including profiling, which produces legal effects concerning him or her or similarly significantly affects him or her.'

[16] Graham Greenleaf, Andrew Mowbray and Philip Chung 'Building sustainable free legal advisory systems: Experiences from the history of AI & law' (2018) 34(1) *Computer Law & Security Review* 314, particularly at [3.6].

[17] See 'Building sustainable free legal advisory systems', above cited, particularly [3.11] – [3.14].

4 Conclusions: Summary of recommendations

In this submission, we are very supportive of the general approach taken in *Cracking the Code*.

We make the following suggestions for further strengthening of the recommendations made:

1. It is desirable to pursue both conversion of existing rules to code, and simultaneous production of new legislation in both conventional and machine-processable form. Pragmatic considerations of the priorities for expenditure of public funds will be determinative.
2. Any program which is useful for RaC development should allow both declarative and imperative programming, but it is essential to provide for a rich set of declarative programming capabilities.
3. Although no known technology yet achieves automated conversion of a whole piece of legislation (Act or regulation), or a significant part thereof, into code which will run, with acceptable accuracy, it is essential to make advances toward automating the conversion from legislation to code, particularly in relation to 'scaling up' applications from being small demonstrations to become 'real world' tools. This could be assisted by approaches such as (i) focusing on some parts of legislation with relatively uniform structure, such as definitions; or (ii) partial conversion of elements of legislative sections, falling short of functioning rules; or (iii) competitions such as in other areas of AI. Such research should be encouraged.
4. No software yet available has all the desirable features needed for RaC development. Although the DataLex software is said by Morris to have more desirable features than other software, it is not one of the programs detailed in *Cracking the Code*.
5. It should be recognised that non-profit Non-Government Organisations must be able to participate in the development of Rules as Code, to utilise it for the benefit of their clients, and to modify government-developed RaC to develop their own apps which meet the needs of their clients. The legal assistance sector, and not only private law firms, must be included. The roles of legal information institutes (mainly in common law countries), and legal informatics research institutes (mainly in civil law countries) should be recognised.
6. Rules which may sometimes require some discretion or interpretation in how they are implemented should not be avoided in RaC projects, but steps should be taken to assist users with the necessary interpretations if they arise.
7. The principle of *Transparency* is best achieved by knowledge-bases written in a 'quasi-natural-language' knowledge representation.
8. The principle of *Traceability* should *require* that 'the coded rules isomorphically reflect the original rules', particularly so that they can be audited by third party experts.
9. A principle of *Accountability* should require that, where coded rules are regarded as authoritative, the official body responsible for making them available should accept 'accountability' for them being correct, either

through legislation of enforceable policy.

10. A principle of *Appealabilty* seems to limited, and would be better as part of a broader principle called something like 'appropriate application'. Appeal rights are not sufficient when automated decision-making should not have been implemented at all.
11. Principles of *Availability* and *Interoperability* may be more easily achieved by a 'quasi-natural-language' knowledge representation.
12. A *Security* principle can be ensured with official coded rules by use of digital signatures, but this is more difficult when they are incorporated in broader rule-sets.
13. A further principle of *Sustainability* is recommended to assist in minimising the 'knowledge acquisition bottleneck', by eliminating specialist coders as far as possible.
14. A principle of *Access to legal sources* is also needed because in most legal applications, encoded rules cannot by themselves deal with every situation that may arise in practice, so rules should be linked to the legal sources on which the rules are based.