Parsable Law Marketplace

Zdenek Masek, LLB

November 5, 2017

Abstract

Current automated legal assessment languages are mostly one purpose, proprietary, tied to one expert system and do not enable a wider collaboration and easy programming nor are they based on a widely supported standard. This work proposes the Unified Modeling Language as the standard for the development of a publicly accessible library of computer parsable laws based on a mass collaboration and a hashgraph market principle.

Contents

Listings

List of Figures

1 Introduction

Legal expert systems have been used since the 1980's.[1] Most of them are based on proprietary internal logic programmed exclusively for their own usage, although some of them involve open source legal domain specific programming languages intended for widespread use in computerised expression of human laws.

Expert systems are being used in a wide range of jurisdictions and areas of law, helping a diverse spectrum of users from members of the public to judges in making relevant decisions.[2] However, there are so many laws that no current system covers them all. For illustration, the World Legal Information Institute project currently lists 1829 databases from 123 jurisdictions[3] and the European Union alone enacted over 100,000 legislative acts since 1957.[4] Not surprisingly the numbers have been ever growing: the EU has enacted 1,465 new legal acts since the beginning of 2017.[5] Similarly, the 93rd Congress enacted the total of 26,222 legislative acts and the current 115th has enacted 7,353 acts only since January 2017.[6] The growth of case law does not lag behind either.

2 The Features of an Expert System

An expert system should provide instant answers to legal situations in any jurisdiction or a combination of jurisdictions, lead through procedures and generate output such as legal documents or execute tasks, e. g. alert the authorities or instruct physical control mechanisms, while eliminating repetitive and time consuming human tasks. There is a huge potential to create numerous new jobs for legal technologists if its internal logic is easy to program.

It can work in a legal capacity or as a compliance application in both public and private sectors. In assessing a given situation it should consider all issues and possibilities, find pathways, and test multiple scenarios. Its inherent feature as a computer technology is that it does not forget or omit. It

[1] Pamela S. Katz, *Expert Robot: Using Artificial Intelligence to Assist Judges in Admitting Scientific Expert Testimony*, 24 ALB. L. J. SCI. & TECH. 1, 31 (2014).

[2] *Id.* 32, 33.

[3] WorldLII, *World Legal Information Institute*, WORLDLII, http://www.worldlii.org/ (last visited Oct. 29, 2017).

[4] Dimiter D. Toshkov, *55 Years of European Legislation*, DIMITER TOSHKOV, http://www.dimiter.eu/Eurlex.html (last visited Oct. 29, 2017).

[5] EUR-Lex, *Legal acts – statistics*, ACCESS TO EUROPEAN UNION LAW, http://eur-lex.europa.eu/statistics/legislative-acts-statistics.html?locale=en (last visited Oct. 29, 2017).

[6] Govtrack, *Statistics and Historical Comparison*, HISTORICAL STATISTICS ABOUT LEGISLATION IN THE U.S. CONGRESS, https://www.govtrack.us/congress/bills/statistics (last visited Oct. 29, 2017).

also does not fail to detect breaches and so it can unmistakably take relevant action and it can even enable preventive detection of loopholes in law.

It should enable constant improvements either through human input or self learning capability. Further it should provide high confidentiality and portability either as a self contained system running on a standalone device or a widely accessible service as a server based or distributed network application. It should be easy to integrate with input and output interfaces such as document content retrieval systems, speech recognition and synthesis, surveillance technologies and with artificial intelligence applications.

Its high market viability would be ideally based on the most widely supported open source technologies; in comparison with humans it is highly reliable and efficient as it works at the speed of its hardware and the cost of electricity.

3 Output Options

As a lawyer's work is expressed in spoken and written words, these are the primary output options for any legal software, with printed output as the simplest one to begin with. This does not pose much of a technical issue and basically consists of automated filling of court forms and generating court bundles, the former being PDF[7] forms and the latter any text format being capable of indexing, paginating and inserting evidential documents. PDF forms can be processed using the PDF Toolkit[8] and as regards bundles, they can be e. g. Latex[9] or LyX[10] documents exported into PDF.[11] Suitable legal citation systems are available on an open source basis, such as Bluebook[12] in the U.S. and Oscola[13] in the U.K., to name just the two largest legal markets. All the generating steps are taken in one process scripted in Python[14] or any other language of one's preference.

[7] International Organization for Standardization, *ISO 32000-1:2008 Document management – Portable document format – Part 1: PDF 1.7*, INTERNATIONAL ORGANIZATION FOR STANDARDIZATION, https://www.iso.org/standard/51502.html (last visited Oct. 7, 2017).

[8] Stewart Sid, *PDFtk - The PDF Toolkit*, PDF LABS, https://www.pdflabs.com/tools/pdftk-the-pdf-toolkit/ (last visited Oct. 7, 2017).

[9] The LaTeX Project, *LaTeX – A document preparation system*, THE LATEX PROJECT, https://www.latex-project.org/ (last visited Oct. 7, 2017).

[10] The LyX Team, *LyX – The Document Processor*, LYX, http://www.lyx.org/ (last visited Oct. 7, 2017).

[11] International Organization for Standardization, *supra* note 7, at the same address.

[12] Christopher DeCoro, *LaWTeX*, SOURCEFORGE, https://sourceforge.net/projects/lawtex/ (last visited Oct. 29, 2017).

[13] Paul Stanley, *Oscola – BibLaTeX Style for the Oxford Standard for the Citation of Legal Authorities*, CTAN: COMPREHENSIVE TEX ARCHIVE NETWORK, https://ctan.org/pkg/oscola (last visited Oct. 29, 2017).

[14] Python Software Foundation, *Python*, PYTHON, https://www.python.org/ (last visited Oct. 7, 2017).

Generating the output documents being the essentials, the next step is automated advanced processing of the very content of the text so produced. And that is when legal technologists' programming steps in: if the software can compose the chronology of events leading to litigation, make relevant legal conclusions itself and suggest the content of statements of case, then a lawyer user's focus can shift from repetitive text editing to legal matters at the highest possible level.

4 Approaches to Codifying the Law

There have been various approaches how to enable computerised processing of the law. Some of them focus on converting human readable laws into machine readable form, others concentrate on semantic analysis of legal texts. So, projects like the Estrella Project,[15] OWL2,[16] CEN MetaLex,[17] Oracle Policy Modeling[18] or the Hammurabi Project[19] have developed special programming languages that could serve as protocols for exchange of legal information. Projects using artificial intelligence such a IBM Watson,[20] IP Soft Amelia,[21] Ravn,[22] on the other hand, use natural language processing[23] methods to extract legal logic directly from sources of human laws. Systems already in practice are mostly based on legal rules hard coded in an existing programming language of their choice and update the embedded legal knowledge either by direct coding, for instance Stanford,[24] A2J

[15]Lawi Project, *Estrella Project*, LAWIN.ORG, http://lawin.org/ (last visited Oct. 8, 2017).

[16]World Wide Web Consortium, *OWL 2 Web Ontology Language Document Overview (Second Edition)*, W3C RECOMMENDATION, https://www.w3.org/TR/owl2-overview/ (last visited Oct. 8, 2017).

[17]European Committee for Standardization, *CEN MetaLex Open XML Interchange Format for Legal and Legislative Resources*, CEN METALEX, http://www.metalex.eu/ (last visited Oct. 8, 2017).

[18]Oracle, *Oracle Policy Automation on OTN*, ORACLE TECHNOLOGY NETWORK, http://www.oracle.com/technetwork/apps-tech/policy-automation/overview/index.html (last visited Oct. 29, 2017).

[19]Michael Poulshock, *Codifying U.S. Law in Machine-Executable Form*, THE HAMMURABI PROJECT, http://mpoulshock.github.io/hammurabi/ (last visited Oct. 8, 2017).

[20]IBM, Inc., *Watson*, IBM, https://www.ibm.com/watson/ (last visited Oct. 8, 2017).

[21]IPSoft, *Amelia the Most Human AI*, IPSOFT, http://www.ipsoft.com/amelia/ (last visited Oct. 8, 2017).

[22]iManage, *RAVN The Power of Understanding Your Documents*, IMANAGE, https://imanage.com/product/ravn/ (last visited Oct. 8, 2017).

[23]NLTK Project, *Natural Language Toolkit*, NLTK 3.2.5 DOCUMENTATION, http://www.nltk.org/ (last visited Oct. 8, 2017).

[24]Prof. Harry Surden, *Project CALC (Computer Assisted Legal Compliance)*, STANFORD LAW SCHOOL PROJECTS, https://law.stanford.edu/projects/project-calc-computer-assisted-legal-compliance-2/ (last visited Oct. 8, 2017).

Author,[25] and Modria[26], or via a graphical user interface so that no coding is required to amend existing legal rules or add new ones, which is the method used by Neota,[27] or by filling in web page forms as with Donotpay[28]. These are, however, tied to their own expert system. Overall, none of the implementations has managed to create a large library of parsable law, only specialised collections or incomplete libraries and so a general standard is still missing.

Garret Wilson pointed out both procedure oriented and object oriented modeling concepts as a common denominator between lawmaking and software design.[29] While software development is a rather recent discipline, in law the analogy has its roots in legal theories of ancient Antic philosophers.[30] And while the law has developed these concepts into an elaborate system contained in myriads of legal texts written in what is familiarly entitled as "legalese", the need for formal expression in computer science led to the development of the Unified Modeling Language[31]as a means of systematising the design of ever more complex computer programs.[32]

5 Connecting the Law and Computer Code

The UML describes systems and rules in a structured and formalised way. Programmers know how to implement systems and processes described by the UML in code, but when it comes to programming laws as expressed in legal parlance, lack of legal education presents a significant obstacle. And vice versa, most lawyers are not familiar with coding. This results in a gap between the two disciplines; it is proposed that the UML can bridge the gap: lawyers convert laws into the UML and programmers or conversion software further turn it into relevant applications.

[25] Center for Computer-Assisted Legal Instruction (CALI) and IIT Chicago-Kent College of Law, *Access to Justice Author*, A2J AUTHOR, https://www.a2jauthor.org/ (last visited Oct. 8, 2017).

[26] Tyler Technologies, *Modria: Expanding Access to Justice with Online Dispute Resolution*, TYLER TECHNOLOGIES, https://www.tylertech.com/solutions-products/modria (last visited Oct. 8, 2017).

[27] Neota Logic, *Neota Logic is re-imagining the way professionals provide their services with AI-powered applications that intelligently automate expertise, workflow, and documents.*, NEOTA LOGIC, https://www.neotalogic.com/ (last visited Oct. 8, 2017).

[28] Joshua Bowder, *Automatically sue Equifax for up to 25,000 Dollars (Depending on Your State)*, DONOTPAY, https://donotpay-search-master.herokuapp.com/ (last visited Oct. 8, 2017).

[29] Garret Wilson, *Refactoring the Law: Reformulating Legal Ontologies*, UNIVERSITY OF SAN FRANCISCO SCHOOL OF LAW JURIS DOCTOR WRITING REQUIREMENT 1 2006.

[30] *Id.*

[31] Object Management Group, Inc., *What is UML?*, UNIFIED MODELING LANGUAGE, http://www.uml.org/ (last visited Oct. 8, 2017).

[32] PAUL KIMMEL, UML DEMYSTIFIED McGraw-Hill, Inc. New York, NY, USA, 2006.

Most of UML software editors create diagrams by the ”drag and drop” method where the user is concerned with placing graphical UML symbols for model nodes and drawing lines representing model edges between them to depict the desired flow of logic. However, their files store positioning of individual nodes and edges rather than the logic underlying the diagrams. PlantUML,[33] on the contrary, generates UML diagrams automatically from a simple source code written by the user. The program takes care of graphical layout and the user can fully concentrate on the logic of the model in question. The PlantUML source code can be easily shared, split apart, joined together and exchanged between users so there is a significant potential for peer to peer collaboration in it. It is open source and widely supported as well. As it is graphically expressed, it is easy for legal technologists to learn and programmers can easily convert it into programming languages for further usage in automated legal assessment.

6 Expressing the Law in the UML

A statute defines which visa category is issued for the purposes of study:[34]

> (15) The term ”immigrant” means every alien except an alien who is within one of the following classes of nonimmigrant aliens . . .
>
> (F)(i) an alien having a residence in a foreign country which he has no intention of abandoning, who is a bona fide student qualified to pursue a full course of study and who seeks to enter the United States temporarily and solely for the purpose of pursuing such a course of study consistent with section 214(l) at an established college, university, seminary, conservatory, academic high school, elementary school, or other academic institution or in a language training program in the United States, particularly designated by him and approved by the Attorney General after consultation with the Secretary of Education, which institution or place of study shall have agreed to report to the Attorney General the termination of attendance of each nonimmigrant student, and if any such institution of learning or place of study fails to make reports promptly the approval shall be withdrawn, and (ii) the alien spouse and minor children of any such alien if accompanying him or following to join him;

[33] Arnaud Roques, *PlantUML*, PLANTUML, http://plantuml.com (last visited Oct. 13, 2017).

[34] 8 U.S.C. § 1101(a)(15)(F), Sec. 101(a)(15)(F) (2011).

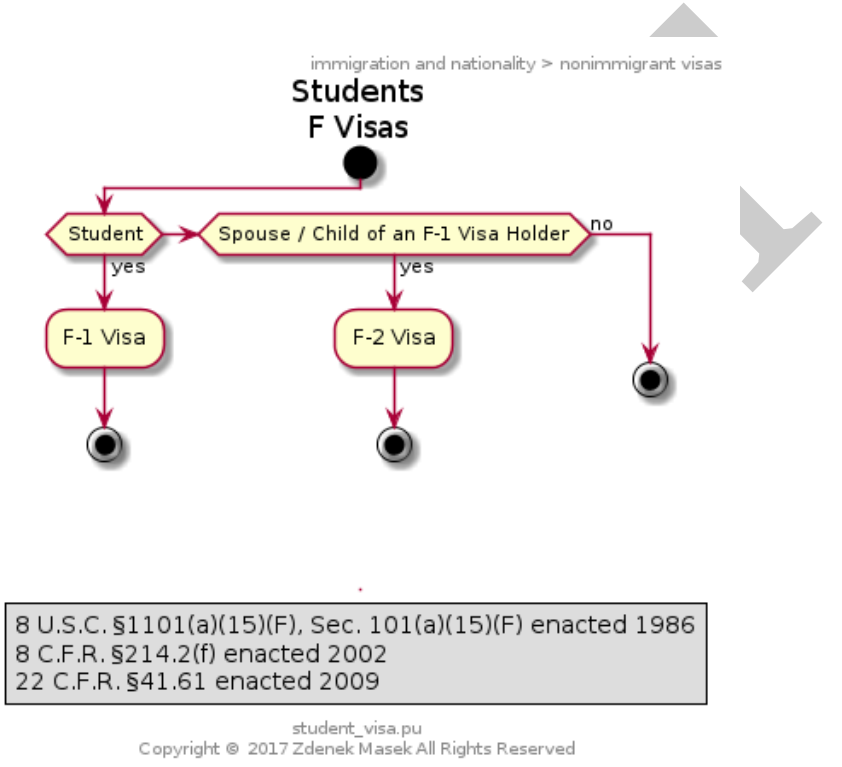

Figure 1: Student Visas. A UML activity diagram here serves as a legal decision making flowchart. It features enactment date so that applicability for a given period can be properly drawn by an expert system. The UML format is usable for any source of law, i.e. it can model a case's ratio decidendi in the same way as a statutory or non-statutory provision.

Thus, a UML activity diagram in Fig. 1 shows that, at the most simplified level, visa applicants who fulfill all the preconditions required for students fall within the F-1 visa category while their dependants within the F-2 category. The UML provides diagrammatic grammar to express legal provisions formulated in any declarative format including cases. It can as easily model the decision of the Court that an alien who is in breach of the conditions of her nonimmigrant student visa status is deportable.[35] Thus, the preconditions listed in the statute could be included in the diagrammed model so that after a conversion into an executable software module they would be evaluated within the same function or, if the preconditions were complex it might be more convenient to deal with them in a separate UML model, i. e. another code snippet. Accordingly, the resulting software function would be placed in its own module which would return a value corresponding with evaluation of relevant input information. The returned value would be then evaluated by the visa category decision making module so that they would form a pipeline of evaluations eventually leading to relevant conclusions. The pathway of the pipeline would be determined by the particular expert system in question rather than the UML module as the modules' only purpose is to provide law in machine readable format.

Any other legal area could be processed in the same way and the proposed framework enables processing of law at any level of detail, depending on whether snippets evaluating legal provisions at the lowest elementary level are available. As the code syntax is rather trivial, any lawyer can design new snippets as and when required and contribute them to the public collection, amend existing ones, or simply keep them for her own usage.

Listing 1 shows the model as expressed in PlantUML source code of that diagram.

Listing 1: Student Visas PlantUML Source Code

```
1  @startuml
2
3  title
4    Students
5    F Visas
6  end title
7
8  legend
9    8 U.S.C.  1101 (a)(15)(F), Sec. 101(a)(15)(F)
       enacted 1986
10   8 C.F.R.  214 .2(f) enacted 2002
11   22 C.F.R.  41 .61 enacted 2009
12 end legend
```

[35] Matter of Yazdani, 17 I. & N. Dec. 626 (BIA 1981).

```
13
14 header
15   immigration and nationality > nonimmigrant visas
16 end header
17
18 footer
19   student_visa.pu
20   Copyright    2017 Zdenek Masek All Rights Reserved
21 end footer
22
23 start
24
25 if (Student) then (yes)
26   :F-1 Visa;
27   stop
28 elseif (Spouse / Child of an F-1 Visa Holder) then (
       yes)
29   :F-2 Visa;
30   stop
31 else (no)
32   stop
33 endif
34
35
36 @enduml
```

The code snippet may contain a more or less elementary piece of law, be it any source of law or any rule outside the legal realm. A collection of snippets forms a library of legal knowledge for further conversion into programming languages.

For study purposes and usage outside regulated legal application the snippets can be open source and perhaps be attributed a credibility score based on public consensus. That credibility would be reflected in probability of accuracy of legal conclusions made by a system based on such snippets. Snippets intended for use in official legal expert systems,[36] however, would need to be endorsed by their issuer and carry a cryptographic signature to guarantee their integrity; so would need executable modules derived from them, to guarantee the authenticity of the legal provisions codified in them and also for the purposes of identifying them among a huge number of other similar snippets. Further, a suitable mechanism for their fast automated ”on

[36] John O. McGinnis and Russell G. Pearce, *The Great Disruption: How Machine Intelligence Will Transform the Role of Lawyers in the Delivery of Legal Services*, 82 FORDHAM L. REV. 3041, 3061 (2014).

the fly" verifying needs to be implemented. The distributed ledger technology appears to be an optimal solution here.

The snippets would enable the gargantuan task of gradual creation of a near complete library of the world's laws and updating it in a piecemeal manner. With a wider popular support, the UML would become a general standard in parsable law and expert system providers would be able to design more advanced products based on that library. Existing expert systems could import snippets in order to extend their own legal knowledge libraries.

As soon as artificial intelligence advances to the point that it can reliably extract legal principles from human readable sources of law, it will also be able to process the snippets as well. And in fact most likely even sooner as they embed structured legal knowledge in a contemporary computer readable form.

7 Converting a Legal Code Snippet Into a Programming Language

A UML model of a particular law provision is defined as a function in a Python module in Listing 2.

Listing 2: Student Visas Python Source Code. A PlantUML activity diagram converted into a Python module. The original PlantUML source code is preserved in comments followed by corresponding Python commands for easier monitoring of integrity with the underlying UML and relevant law as well as meta information from the UML snippet such as laws' identifiers and time validity. Modules should be able to re-generate UML diagrams they are derived from. They should also employ a logging facility to document the assessment processes they carry out.

```
1  #!/usr/bin/env python
2  # -*- coding: utf-8 -*-
3
4  def visa_category_f_students(x):
5    """
6    title
7      Students
8      F Visas
9    end title
10
11   legend
12     8 U.S.C.  1101 (a)(15)(F), Sec. 101(a)(15)(F)
          enacted 1986
13     8 C.F.R.  214 .2(f) enacted 2002
14     22 C.F.R.  41 .61 enacted 2009
```

```
15  end legend
16
17  header
18    immigration and nationality > nonimmigrant visas
19  end header
20
21  footer
22    student_visa.pu
23    Copyright    2017 Zdenek Masek All Rights Reserved
24  end footer
25  """
26
27  yf1 = ['student']
28  yf2 = ['spouse of an f-1 visa holder','child of an f
        -1 visa holder']
29
30  #.# if (Student) then (yes)
31  if x in yf1:
32    #.# :F-1 Visa;
33    return 'f-1'
34  #.# elseif (Spouse / Child of an F-1 Visa Holder)
        then (yes)
35  elif x in yf2:
36    #.# :F-2 Visa;
37    return 'f-2'
38  #.# else (no)
39  else:
40    #.# stop
41    return 'not eligible for category f visas'
42    #.# endif
```

The function uses default values for input variables where appropriate so that it does not stop executing if any input variable is missing. Alternatively, it calls another function that prompts the user for required information or otherwise obtains it, for example from a data storage or by a query from a distant source.[37] That enables to run automated legal assessment with an incomplete library of legal modules and to design a viable software product even with a minimum of modules.

The whole system forms a hierarchy of at least two types of modules: those that embed substantial legal knowledge and those that embed the processes of legal assessment. The former ones call the latter ones as and when needed according to relevant legal ontology.

[37] MIKE MCGRATH, PYTHON IN EASY STEPS 80 (In Easy Steps Ltd, 2015).

8 The Marketplace

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Etiam lobortis facilisis sem. Nullam nec mi et neque pharetra sollicitudin. Praesent imperdiet mi nec ante. Donec ullamcorper, felis non sodales commodo, lectus velit ultrices augue, a dignissim nibh lectus placerat pede. Vivamus nunc nunc, molestie ut, ultricies vel, semper in, velit. Ut porttitor. Praesent in sapien. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Duis fringilla tristique neque. Sed interdum libero ut metus. Pellentesque placerat. Nam rutrum augue a leo. Morbi sed elit sit amet ante lobortis sollicitudin. Praesent blandit blandit mauris. Praesent lectus tellus, aliquet aliquam, luctus a, egestas a, turpis. Mauris lacinia lorem sit amet ipsum. Nunc quis urna dictum turpis accumsan semper.

9 Conclusion

There is a potential and indeed a need for collaboration to form on a large scale. Not all lawyers write law books but every law student who is willing to learn a simple syntax may write UML snippets and even make an extra income from it. That will lead to the emergence of a whole new ecosystem of educational activities, materials and supporting services.

It is recognised that to ensure usability in official legal assessment the code will need to be endorsed by legal professionals and in order to achieve market viability it will need to be developed on a commercial basis, although not necessarily on a corporate one. The distributed ledger technology enables to form a hashgraph based market for a worldwide community of freelance legal technologists who will design and sell UML snippets of law in any specialisation of their choice and market demand. They will follow the evolution of law they will be interested in and update their code snippets accordingly to satisfy demand for up to date legal code.

10 Acknowledgments

The author specially thanks to Michael Poulshock for insightful comments on codifying the law,[38] Christopher DeCoro for kind technical support on his LawTeX automated Bluebook citation system,[39] and Paul Stanley for an informative feedback on his Oscola Biblatex citation system.[40]

[38] Poulshock, *supra* note 19, at the same address.
[39] DeCoro, *supra* note 12, at the same address.
[40] Stanley, *supra* note 13, at the same address.

11 Table of Authorities

CASES Page

CONSTITUTIONAL PROVISIONS, STATUTES AND RULES

OTHER AUTHORITIES

https://www.govtrack.us/congress/bills/statistics (last visited Oct. 29, 2017) 4

IBM, Inc., *Watson*, IBM, https://www.ibm.com/watson/ (last visited Oct. 8, 2017) 6

iManage, *RAVN The Power of Understanding Your Documents*, iMANAGE, https://imanage.com/product/ravn/ (last visited Oct. 8, 2017) 6

International Organization for Standardization, *ISO 32000-1:2008 Document management – Portable document format – Part 1: PDF 1.7*, INTERNATIONAL ORGANIZATION FOR STANDARDIZATION, https://www.iso.org/standard/51502.html (last visited Oct. 7, 2017) 5

IPSoft, *Amelia the Most Human AI*, IPSOFT, http://www.ipsoft.com/amelia/ (last visited Oct. 8, 2017) 6

John O. McGinnis and Russell G. Pearce, *The Great Disruption: How Machine Intelligence Will Transform the Role of Lawyers in the Delivery of Legal Services*, 82 FORDHAM L. REV. 3041 (2014) 11

Joshua Bowder, *Automatically sue Equifax for up to 25,000 Dollars (Depending on Your State)*, DONOTPAY, https://donotpay-search-master.herokuapp.com/ (last visited Oct. 8, 2017) 7

Lawi Project, *Estrella Project*, LAWIN.ORG, http://lawin.org/ (last visited Oct. 8, 2017) 6

Michael Poulshock, *Codifying U.S. Law in Machine-Executable Form*, THE HAMMURABI PROJECT, http://mpoulshock.github.io/hammurabi/ (last visited Oct. 8, 2017) 6, 15

Mike McGrath, *Python in Easy Steps* (In Easy Steps Ltd, 2015) 13

Neota Logic, *Neota Logic is re-imagining the way professionals provide their services with AI-powered applications that intelligently automate expertise, workflow, and documents.*, NEOTA LOGIC, https://www.neotalogic.com/ (last visited

Oct. 8, 2017) 7

NLTK Project, *Natural Language Toolkit*, NLTK 3.2.5 DOCUMENTATION, http://www.nltk.org/ (last visited Oct. 8, 2017) 6

Object Management Group, Inc., *What is UML?*, UNIFIED MODELING LANGUAGE, http://www.uml.org/ (last visited Oct. 8, 2017) 7

Oracle, *Oracle Policy Automation on OTN*, ORACLE TECHNOLOGY NETWORK, http://www.oracle.com/technetwork/apps-tech/policy-automation/overview/index.html (last visited Oct. 29, 2017) 6

Pamela S. Katz, *Expert Robot: Using Artificial Intelligence to Assist Judges in Admitting Scientific Expert Testimony*, 24 ALB. L. J. SCI. & TECH. 1 (2014) 4

Paul Kimmel, *UML Demystified* McGraw-Hill, Inc. New York, NY, USA, 2006 7

Paul Stanley, *Oscola 1–BibLaTeX Style for the Oxford Standard for the Citation of Legal Authorities*, CTAN: COMPREHENSIVE TEX ARCHIVE NETWORK, https://ctan.org/pkg/oscola (last visited Oct. 29, 2017) 5, 15

Prof. Harry Surden, *Project CALC (Computer Assisted Legal Compliance)*, STANFORD LAW SCHOOL PROJECTS, https://law.stanford.edu/projects/project-calc-computer-assisted-legal-compliance-2/ (last visited Oct. 8, 2017) 6

Python Software Foundation, *Python*, PYTHON, https://www.python.org/ (last visited Oct. 7, 2017) 5

Stewart Sid, *PDFtk - The PDF Toolkit*, PDF LABS, https://www.pdflabs.com/tools/pdftk-the-pdf-toolkit/ (last visited Oct. 7, 2017) 5

The LaTeX Project, *LaTeX 1–A document preparation system*, THE LATEX PROJECT, https://www.latex-project.org/ (last visited Oct. 7, 2017) 5

The LyX Team, *LyX 1–The Document Processor*, LYX, http://www.lyx.org/ (last visited Oct. 7, 2017) 5

Tyler Technologies, *Modria: Expanding Access to Justice with Online Dispute Resolution*, TYLER TECHNOLOGIES, https://www.tylertech.com/solutions-products/modria (last visited Oct. 8, 2017) 7

World Wide Web Consortium, *OWL 2 Web Ontology Language Document Overview (Second Edition)*, W3C RECOMMENDATION, https://www.w3.org/TR/owl2-overview/ (last visited Oct. 8, 2017) 6

WorldLII, *World Legal Information Institute*, WORLDLII, http://www.worldlii.org/ (last visited Oct. 29, 2017) 4