A Formal Logic Perspective on Legal Theory and Information Technology in a Historical Context

Notes by

Luigi Logrippo
luigi@uqo.ca
http://w3.uqo.ca/luigi/
http://www.site.uottawa.ca/~luigi/

Work in progress!

Started: 2009-08; Last updated: 2013-11-05

This work is dedicated to the memory of Fritz Paradies, a lawyer and scholar from Frankfurt and Amsterdam. In his self-published paper "Enthält Cobol eine juristische Logik?", *written in the mid-1960s, he expressed some of the ideas I mention below. But do we have a basic philosophical bug here? Law is based on the idea that humans have* free will, *something that computers are not supposed to have! I still don't have a good answer to this, but you might wish to read on … and get lost in the similarities, as happened to Fritz and me.*

Contents

1. Introduction

1.1 Motivation and precedents

For many years I have been interested in issues of legal logic, its history, and corresponding developments in Information Technology (IT). Since I have studied law, then I have moved to IT and now again I am interested in law, this document will include reflections taken from the points of view of both areas.

The main purpose of this document is '*notes to myself*', however I am making it available to the world in case it might interest others. I am interested in discussion, in hearing other views. Notable interventions are being added at the end.

So this document is work in progress, and I will keep updating it as I find sources, ideas and time. Expect conjectures, incompleteness, inconsistency, rough text, lack of bibliographic references, etc.

The terms *norm* and *normative system* will be used often to refer collectively to the different types of rules, laws, computer-based policies, etc., that will be considered here. In a well-known 1972 book Alchourròn and Bulygin loosely define norms as statements that relate cases to solutions. The authors did not intend to extend the scope of their definition beyond the social sciences and law but clearly according to their definition norms and normative systems exist in IT.

In 1993, Jones and Sergot wrote:

> *"The general position which we here develop and illustrate is that---at the appropriate level of abstraction--- law, computer systems, and many other kinds of organisational structure may be viewed as instances of normative systems. We use the term to refer to any set of interacting agents whose behaviour can usefully be regarded as governed by norms. Norms prescribe how the agents ought to behave, and specify how they are permitted to behave and what their rights are. Agents may be human individuals or collections of human individuals, or computer systems or collections of computer systems. Normative systems include systems of law, abstract models of computer systems, and hybrid systems consisting of human and computer agents in interaction."*

I subscribe to this view, with two exceptions. First of all, are normative systems sets of interacting agents (legal institutions), or sets of norms? This question, whether institutions or laws have come first, has been extensively debated in philosophy of law and therefore it should be avoided if possible. Isn't it similar to the 'chicken and egg' problem? In these notes, I am mostly interested in sets of norms. Second, this view characterizes norms in terms of the deontic concepts of obligation ('ought to') and permission. This is a common view, endorsed by the best authorities. However normative reasoning is possible without deontic concepts.

So the main emphasis of these notes is not on legal logic and legal reasoning alone. It is on finding traces of actual use of logic reasoning in applied legal contexts, and note correspondences with methods in Computer Science, Software design and IT applications.

According to [Alchourròn, Bulygin 1972] '*The rules of inference have rarely, if ever, been discussed by legal writers, who often are not aware of their existence.*' But this is consistent with the role of logic in all disciplines that use it, including mathematics and philosophy. Texts in disciplines that use logic will very seldom include explicit logic derivations, but faults in the use of logic will be pointed out and often will be considered to invalidate the whole argument. So logic tends to be invisible as long as it is correct. Legal thinking is dominated by political, ethical, sociological and economic concerns. These other disciplines, in relation with law, posit assumptions or *domain axioms* that are far more visible than the inference rules of logic that are used to derive conclusions from them. In other words, legal thinking is dominated by assumptions coming from these disciplines, as well as from law and precedent. The coverage of these assumptions is extended by using analogical thinking. After this, the application of law to the specific case is often a simple deduction (e.g. a syllogism) of which we are often not even aware.

One will not find in legal texts phrases such as: '*All taxpayers who make between 100K$ and 150K$ are in the 30% tax bracket; Alice is a taxpayer who makes between 100K$ and 150K$; hence Alice is in the 30% tax bracket*'. Statements formed in this way (called syllogisms) are found in logic manuals and seldom anywhere else. But statements such as: '*Alice is in the 30% tax bracket because she makes between 100K$ and 150K$*' are essentially equivalent, can be represented in formal logic and can be used for logical inference. No legal system, in fact no science, is possible without this type of reasoning.

Occasionally however, fairly complex logical reasoning can be found in judicial texts. Many such examples have been published in the literature. Several are analyzed in plain language in [Tammelo 1978], and here is one (page 100, please refer to the book for some background that is necessary in order to fully understand this text):

> *For every* m*: if* m *commits larceny then for some* n*:* m *takes* n *and* m *acts feloniously.*
> *For every* m*: exactly if for some* n*:* m *takes* n *and* m *acts feloniously then* m *commits a trespass.*
> *For every* m*: if for some* n*:* m *takes* n *and* m *acts innocently and* m *fraudulently converts* n *to his use subsequent to the original taking and neither* m *is a servant and* m *commits embezzlement or* m *is an agent and* m *commits misappropriation nor* m *commits a bailee's larceny then it is not that* m *commits larceny.*
> *Ashwell takes the souvereign and Ashwell is innocent.*
> *Ashwell fraudulently converts the souvereign to his own use subsequent to the original taking.*
> *Therefore, it is not that Ashwell is guilty of larceny.*

[Stelmach, Brozek 2006] identify four historically established methods used by legal practitioners and theoreticians: logic, analysis, argumentation and hermeneutics. They claim that they are all useful and used, none being subordinate to the others. The boundaries among these methods are not sharp.

So I will follow [Fraunce 1588]: “I sought for Logike in our Law, and found it as I thought”. Logic reasoning is identified by precise and complete definitions and clear logic inferences, as well as an explicit effort to maintain consistency (avoidance of contradictions).

The proponents of the use of formal logic in the legal process have often pointed out that such use helps towards predictability in the process, which is required for assuring the principle of *certainty of law*, proposed by Max Weber among others as necessary condition for the achievement of economic goals. In other words, the results of the legal process are more predictable and uniform if the law is logically clear and consistent and the decisions are reached by formal logical inference from the law and the established facts. Today, a technological argument for the use of formal logic in the legal process is provided by the fact that information systems are increasingly entrusted roles of legal relevance and the most obvious mechanism for computers to draw legal conclusions is logical deduction. In IT terms, the laws are the policies and the established facts are the context. Examples are in e-commerce and privacy protection systems, among others. Multi-agent systems are very similar to social systems with their policies, which essentially have the function of laws, but are inferred and enforced automatically.

With the availability of efficient Boolean satisfaction (SAT) algorithms [Malik, Zhang 2009], many application possibilities are now open.

There is no lack of theoretical works on legal logic, going back to the 16th Century. If I have read well, the ideas of logic formalization of legal thinking and even of automation of legal thinking go back to Leibniz (1646-1716), but I haven’t been able to research this point yet.

[Lorini 2003] and [Kalinowski 1983] cite quite a number of publications on the subject, accelerating towards the 18th, 19th and 20th C. I have seen very few of these treatises, however most probably they take the word ‘logic’ rather loosely, with emphasis on argumentation. But this is OK, good argumentation must be based on good logic, although this connection may not be obvious. Starting in the 1920s, there has been further acceleration of interest and papers and books have appeared, with different orientations. Formal logic approaches to legal interpretation and deduction were developed. Very significant research in formal logic of law was done in the 1950s, see http://www.jstor.org/stable/2269771?seq=1 (consulted October 2009). This link contains reviews of books and papers by Layman E. Allen and William Halberstadt. Layman Allen may have been one of the first to apply modern formal logic for the interpretation of a real law, the USA Internal Revenue Service (IRS) Code (what better candidate?)

Lucien Mehl, a French jurist and legal documentation expert, was perhaps the first person to articulate the view of automation in the legal world, in a paper of 1959 [Mehl 1959]. His ideas were of course simple by today’s standards, based on binary encoding of logic concepts. He viewed his ‘juridical machine’ as an aid for the jurist and judge, rather than a substitute. However in a paper published some decades later [Mehl 1995] he recognized that in some cases legal decisions can be entirely automated.

From 1959, there was a newsletter on logic and law in the USA, it was called M.U.L.L. (Modern Uses of Logic in Law) and was edited by Layman E. Allen (copies can now be found in the WWW). This newsletter was soon swamped with papers on jurimetrics and then changed name to ‘Jurimetrics Journal’. [Klug 1951][Tammelo 1969] [Alchourròn, Bulygin 1972] are some of the first major monographs that have proposed and illustrated the use of formal logic methods for the analysis of legal reasoning. [Narayanan, Bennun 1998] contains several articles presenting the view of the field at the end of the last century.

The philosophical basis of this direction in the study of law may perhaps be traced to legal positivism, which had different but partly converging developments in Austria, Germany, UK and USA. [Haack 2007] provides ample background on this. [Sartor, 2005, Chapters 5 and 6] are interesting chapters on Law and Logic.

Nowadays, there are several international conferences or workshops which deal with research on logic and law:

CLIMA (Computational Logic in Multi-Agent Systems)
DEON (Deontic Logic in Computer Science)
ICAIL (Intern. Conf. on AI and Law)
JURIX (Intern. Conf. on Legal Knowledge and Information Systems)
NorMAS (Normative Multi-Agent Systems)
ReLaw (Requirements Engineering and Law)

Some of the views presented here were published in [Logrippo 2007 and 2011].

1.2 Preliminary concepts

1.2.1 Ontologies

The term *ontology* has a history in philosophy. It has become a technical word in Computer Science, with a somewhat different meaning, and it is in its CS meaning that I will use it. An ontology in this sense is the definition of a set of concepts together with their relationships. Various ways of representing ontologies are: sets of logical axioms involving constants, data types, diagrams (e.g. UML diagrams), conceptual taxonomies, etc. Many different, and very complex, ontologies can be present in a legal system. Some can be explicitly defined in the law, others can be considered to be understood, or 'inherited' from areas of knowledge that are technical or common knowledge. For example, inheritance law involves (at least) a family ontology, an ontology describing rights that the deceased may hold, an ontology describing the objects on which rights can be held, and an ontology describing the structure of testaments.

The role of ontologies in legal thinking is debated [Sartor 2009]. [Breuker, Valente, Winkels 2004, 2005] present their views on the same subject, including the conclusions of an extensive application study carried out by their group.

Where should ontologies come from? They could come from legal theory, however only informal, partial and often messy ontologies are found in legal textbooks. Ontologies are also different from textbook to textbook. Or they could come from a 'grounded' approach, e.g. from processing the legal texts and deriving relationships in the terminology. In the first case, we risk using irrelevant ontologies. In the second case, we risk not seeing what the concepts really mean because there is a lot of assumed knowledge in legal texts. It seems that an intermediate approach is necessary [Fernández Barrera 2011].

A main problem with ontologies is that, although they are needed to understand the law, often they are not part of the law. In other words, the interpretation of the law can be heavily determined by external concepts. Precision in the law itself won't help if the necessary ontologies are not well defined, but often to make these precise would involve including segments of many sciences, as well also of many concepts that are not defined scientifically. This problem pervades much of legal thinking.

[Winkels 2010] is a web page dedicated to information on legal ontologies. [Fernández Barrera 2009] is a presentation on legal ontologies, showing some historical ontologies in graphic format. See also [Valente 1995] [Valente 2005][Van Engers 2008][Sartor 2011].

1.2.2 Different types of norms, and different ways of representing them

Legal norms can be stated in many different ways, using the expressive power of natural language. As is normal for natural language statements, usually there are several ways of expressing norms in logic.

The analysis of [Dworkin 1978] given in [Verheij 1998] distinguishes between *rules* and *principles* in the following way:

Rules: *if the condition of a rule is satisfied, the rule is applied and its conclusion follows directly*

Principles: *a principle only gives rise to a reason for its conclusion if it applies. Moreover, there can be other applying principles that give rise to both reasons for and reasons against the same conclusion. A conclusion then only follows by weighing the pros and cons.*

In this view, reasoning with principles leads to models of argumentation and types of logic that support the weighing of arguments.

Personally, I have been inspired by the concepts of implementations and requirements, current in software engineering [Logrippo 2007]. Taking this view, a principle is a requirement, which can be used to generate rules and to which rules can conform or not. Conflicting principles are due either to poor legislative practices, or to the desire of the legislator to allow space for different implementations. In this latter situation, eliminating the conflict is indeed a matter of interpretation, which is not a strictly logical process and can be done in several ways, including argumentation.

A *rule*, or 'implementation" type of norm can be expressed as Event-Condition-Action (ECA):

if Event and Condition then Action

meaning that if an event occurs, in the presence of conditions that must be true on the context, then an action should follow. If a nurse requires access to the X-Ray department, and she is working in the Emergency Ward, and it's between 22h and 06hrs, then permission will be given.

Whether a syntactic occurrence should be listed as action or as part of the context condition, this depends on the architecture and ontology that has been defined for the system.

The ECA style is widely used in IT in event-driven architectures, policy-directed systems and rule engines. The ECA style can be immediately translated in Horn-clause style, which is at the basis of the semantics of the programming language Prolog and of its derivates.

Principles are simple logical statements, e.g. "nurses can be given access to the X-ray department only in exceptional circumstances". Note that there is a difference between a logical statement like A ^ B → C and a rule such as 'If A ^ B then C'. The former is a logical statement, an implication; the latter is an *operational* norm; it defines a state transition, with a side effect, by reaching a new state where the postcondition of C is true. The logical statement form is suitable for expressing principles, the state transition form is suitable for expressing rules. ECA style is operational, but the ECA norm makes the implication true.

ECA style allows *forward chaining*, since in the new state new conditions are true and new actions can lead to other actions. In the example above, when the nurse has been given access to the X-Ray department, then new rules may come into play, by which perhaps she can take X-Rays herself or she can ask a technician to take them for her. ECA style allows also *backward chaining*, by which one can ask what are the possible actions and conditions that can lead to a given state. E.g. what are the actions and conditions that can lead to a nurse taking X-rays. Forward and backward chaining can become very interesting in the presence of a rich ontology. If we have a ECA rule If A ^ B then C and also C → D is asserted in the ontology, then all the consequences of D become possible after the state transition. If the X-Ray department includes other equipment, then the rules for the use of that equipment may become applicable.

A.J.I. Jones, R. Kowalski, M.J. Sergot and their collaborators have been the recognized pioneers in this line of thinking.

Normative conditionals and their analysis are discussed in detail in [Sartor 2005, Chapter 20, 21].

It might be considered important to decide which part of a legal system is ontology, and which part is 'something else'. For example, an assertion such as: *Land transfer acts are civil law acts* could be considered ontological, a classifying statement. But what about: *Land transfer acts must be registered with the land registry*. At first, this could be considered a deontic statement, involving the concept of obligation. But it could also be seen as a simple implication, part of an ontology, just like the other assertion.

We should note before we go on that there has been considerable discussion in philosophy of law about the logical nature of norms [Lorini 2003, Kalinowski 1972]. According to some, norms cannot be true or false as logical assertions can, and as example statements such as 'do not drive through a red light!' are brought forward. But is this a norm? According to what I have just said, I consider two kinds of norms. By way of example:

'*no one should drive through a red light*', '*debts must be repaid*', these are statements that can be true in some legal systems, false in others. These seem to be *principles*.

'*If someone does not pay debt, her properties shall be sold and the proceeds distributed to her creditor*s' can be read in two different ways. One is as a logical implication, part of a legal ontology, which can be true or false, and so it is a *principle*. The second is an ECA rule or program, which establishes a postcondition on the basis of a precondition. In this second case, the pre- and post-conditions can be true or false at different states in the system, but the program does not have a truth value. This is an operational *rule*.

However the statement: '*ECA rule X is a valid rule in a given legal system*' can indeed be true or false, but it is not a norm, most likely it should be taken as a meta-norm.

So there are several possible interpretations for a legal statement, which one(s) should be preferred depends on the preferred type of reasoning. Philosophers may dwell on such distinctions for a long time but I am inclined to think that these distinctions don't matter in principle. Simpler is better whenever computers are involved. A typical logic analyzer or theorem prover won't make any difference regarding the provenance or the nature of different logical assertions.

Needless to say, legal texts contain many different types of norms; in addition, laws can be classified in different ways according to the purpose of the classification. Our purpose is logical structure. So the discussion above doesn't even scratch the surface of the subject.

An important fact about legal norms is that they can be decomposed into norms of different types, for example a norm directed to the subject, and another directed to the enforcer. I call the first the Moses norm, and the second the Hammurabi norm (in ECA-atyle). Example: "Parking on Rideau Street is punished by an amend of $10". The Moses norm here is: 'Thou shall not park on Rideau Street". The Hammurabi norm (for the police officer or the judge) is: "If anyone parks on Rideau Street, he or she will be punished with an amend of $10". Both norms can be violated, but the first can be violated by the subject, while the second can be violated by the enforcer. Clearly, each type of violation can have cascading effects with the activation of other norms. Interesting, if a subject is also an enforcer, he can, of course, violate both norms.

1.2.3. The difference between norms and programming language statements

I haven't yet found any reference on this topic, so I will try my best to get in hot water.

Intrinsic in the concept of norm is that it can be violated. Hence the semantic of 'norm' should include a non-deterministic choice (the 'free will') which, in the case of violation, leads to consequences. These can be as simple as flagging the fact that a violation has occurred. This flag can trigger other norms.

In traditional programming languages, the semantic of statements does not include this possibility. Statements are executed in sequence, and choices are deterministic.

Several modern programming languages include constructs that specify nondeterminism or exceptions (e.g. in Java: try, catch, finally). However these must be specifically programmed.

The best programming model for norms can perhaps be found in logic programming languages, where statements can succeed or fail according to the truth or falsehood of preconditions, independently of the order in which they are written.

2. Sumerian and Babylonian codes (about 2000 BC)

Sumerians and Babylonians are well-known for their scientific knowledge. But also their legal codes are written precisely in a rigorously uniform style. The code of Ur-Nammu is said to precede Hammurabi's by some 300 year and is presumed to be the earliest legal text extant. However since Hammurabi's code is more complete, we will concentrate on this one, which is written essentially in the same style.

Here is one of Hammurabi's almost 300 rules (from http://www.wsu.edu/~dee/MESO/CODE.HTM, consulted September 2009):

> *If any one agree with another to tend his field, give him seed, entrust a yoke of oxen to him, and bind him to cultivate the field, if he steal the corn or plants, and take them for himself, his hands shall be hewn off.*

This rule, like many other rules in this code, is written in ECA style:

- The Event here is: *If any one steals the corn or plants, and take them for himself*,
- The Condition is: *If that person has agreed with another to tend his field, give him seed, entrust a yoke of oxen to him, and bind him to cultivate the field*
- The Action is: *His hands shall be hewn off*

Not only this, but in many rules the event itself consists of three elements:

- Subject: in this case *anyone*

- Verb(s): in this case *steal* and *take for himself*
- Object(s): *corn* or *plants*

As well, the Action may contain simple ‘algorithms’ to decide the penalty, this is seen in other rules.

It would be possible to dwell further on the logical structure of these rules. In the case of the rule given above, it can be seen that the Condition defines a ‘precondition’ for the rule to be applied. It can be further analyzed, in this case it describes an existing legal relationship between the main legal Subject (any one) and another legal Subject (another), as well as an object, the field. In its turn, the field contains ‘corn or plants’, The Verb is the ‘trigger’ , the event that causes the rule to become applicable in what otherwise is a normal legal situation.

It has been very interesting for me to see that a legal system can exist at an elementary structural level that is also well known in IT.

Many later codes do not show such uniformity. They show mixtures of rules in ECA style with other rules that are written in various other styles, notably often not explicitly mentioning the consequences of actions, such as violations.

On the negative side, there is in this code a lack of conceptualization. Each rule considers a specific case only, without attempts of generalization. There are no explicit ontologies. There are no subject headings in the code, although rules concerning similar cases are often grouped together (e.g. articles 215-225 group rules applying to physicians). This leads to a multiplication of rules to consider individual cases. The only legal concept to be found in the rule above is the concept of theft, which depends on the concept of property, and neither is defined. Several other rules make reference to an implicit ontology that reflects the Babylonian social structure, from priests to slaves, etc. So forward chaining is very limited, although there may be some: e.g., trivially, if someone is executed for an offence, then inheritance rules come into consideration.

It would be an interesting project to study this code in order to complete this analysis, and better identify the ontologies and the basic structural principles it is based on, to the extent to which they can be identified.

The aim of ancient legislators using the ECA style was probably to strictly control the behavior of their delegates, the judges. It has been speculated that the articles of these codes were decisions actually taken by the King, and it can be questioned whether they were set down as precedents to be followed strictly or as examples. Surely other situations would present themselves that did not exactly match the given patterns. What would the judges do in these cases? Probably infer by analogy, as judges still do today. In any case, modern codes explicitly leave space to judges’ discretion to determine circumstances and dose decisions. This may not be appropriate in simple cases where it is desired to obtain automatic judgments [Logrippo 2011].

We shall see that the ECA style has been often used in the history of legislation, as an effective and simple legislative style.

2.1 Related Systems in IT

2.1.1 Firewalls

There is a striking similarity between the style of the Hammurabi code and the style in which firewall rules are written. A firewall is a part of a computer system or network that is designed to block unauthorized access while permitting outward communication. Essentially, firewalls monitor communication lines and block or authorize entry to data packets showing certain characteristics, such as address of origin or port of destination.

Firewalls are programmed by sets of rules written in ECA style. Most rules are of the type: if *a packet having such characteristics is received, then it should be blocked at the firewall*, or *it should be forwarded to the destination*. Just as it may have been done in the code of Hammurabi, firewall rules are added by the system administrator when the need presents itself. Each rule is on its own and there are no general principles or definitions in a firewall system.

But in sets of firewall rules the order is important: at the arrival of a data packet, the set of rules is scanned top-down, the first applicable rule is executed, and then the next packet is taken into consideration. Other applicable rules later on are not used, so specific rules must appear before more general ones, more important rules before less important ones. Instead in codes of law all rules are equally important and if two rules are applicable for a given situation, both may apply or some conflict-resolution strategy will have to be applied, usually by a judicial authority.

In both firewalls and systems of legal rules, there is a ‘closure norm’. In the case of firewalls, the closure norm is often: if *no rule applies for a given incoming packet, then reject it*. In the case of legal systems, the closure norm is often: *the action is not legally relevant, nothing to do* (e.g. the legal maxim: *nullum crimen sine lege*, no crime without law).

2.1.2 Access control systems (ACS)

ACS are also characterized by ECA rules. Typically, these are used to protect or limit access to resources, such as computer files or also physical resources, such as rooms or equipment. A typical rule in an ACS could state: *nurses can access only the files of patients in their wards, and only during their work hours*; or: in *the army, only officers can access documents classified as ‘top secret’*.

As in the Hammurabi code, ACS rules are often written in the fixed style:

(subject, verb, object, condition, action)

For example: (if a <clerk> <requires access> to the <bank safe><from 0:00h to 7:00><deny>).

A type of ontology that is used in some ACS is the role ontology, and the best known conceptualization of this is provided in the theory underlying RBAC (Role-Based Access Control). Here, roles are essentially functional positions in organizations, to which users can be associated (e.g. Director of Marketing, Chief Surgeon, emergency doctor, etc.). Access rights are associated to roles, and then users are associates to roles. Role hierarchies with access right inheritance can be defined.

A standard computer language has been defined for ACS, by the standards organization OASIS, is XACML. It is related to the access control method ABAC, Attribute-Based Access Control.

The XACML standard defines mechanisms for deciding access and enforcing access, comparable to a judicial system. Two main components of the XACML system are the 'Policy Decision Point' and the 'Policy Enforcement Point'. Exchange messages are defined between these points.

XACML allows the user to set meta-rules to resolve potential conflicts between rules: Deny-Overrides, Permit-Overrides, First-Applicable, Only-One-Aplicable, Lower-Role-Overrides.

[Barker 2012] develops a logic framework for core concepts in access control. He also states: "*In future work, we intend to consider the type of more general authorization model requirements that Jones and Sergot have considered and their representation of rich forms of non-standard access control models within our framework.*" He cites among others [Jones, Sergot 1992]. He died in 2012, unfortunately.

2.2 Discussion

A lesson that may be drawn from these analogies is that, just as legal systems have evolved from the simplest ECA structures into complex structures involving ontologies, many types of legal rules, etc. , firewalls and access control systems will also evolve in similar ways. Legal systems have been developed for a very long time, and so they are very sophisticated. However comparable IT systems are more precisely defined, tuned for automatic decision-taking. Precision and complexity may head towards convergence in time. Convergence will likely be first achieved for legal systems that are meant for automatic implementation. Privacy protection systems may be among the first to witness such convergence.

3. Moses Code (about 1400 BC)

With respect to Sumerian and Babylonian codes, the code that according to the Bible was given by God to Moses shows two important characteristics: use of deontic concepts and higher level of abstraction.

Thou shalt not steal…
Remember the Sabbath day…

are typical rules of this code. Deontically, the first can be written as: it is prohibited to steal; and the second: it is obligatory to remember the Sabbath day. These are the only two deontic modalities used in this code, namely permission is not used. However an implicit understanding of permission seems to be present: that obligatory actions are permitted, that forbidden actions are not permitted.

Opposite to the Hammurabi code, this code ignores the judicial authority and speaks directly to the individual. However in order to enforce it, a judicial authority may be necessary, with corresponding norms.

To see the consideration of large classes of behaviors, consider again the norm: *Thou shall not steal*. There are quite a few specific scattered norms corresponding to this one in the Hammurabi code: e.g. 6, 8, 14, 22, 23, 24… However in each case the Hammurabi code dictates the consequences of each specific type of theft, which are left unspecified in the Moses code. In engineering terms, one could say that a rule written in the 'Moses style' is a *requirement*, to be implemented in the 'Hammurabi style'.

I am getting into things I don't know now, but perhaps one could view the Talmud as an implementation or refinement of God's code.

Another interesting legal concept that is related to Moses is delegation:

Exodus 18:
17: And Moses' father in law said unto him, The thing that thou doest is not good.
18: Thou wilt surely wear away, both thou, and this people that is with thee: for this thing is too heavy for thee; thou art not able to perform it thyself alone.
19: Hearken now unto my voice, I will give thee counsel, and God shall be with thee: Be thou for the people to Godward, that thou mayest bring the causes unto God:
20: And thou shalt teach them ordinances and laws, and shalt shew them the way wherein they must walk, and the work that they must do.
21: Moreover thou shalt provide out of all the people able men, such as fear God, men of truth, hating covetousness; and place such over them, to be rulers of thousands, and rulers of hundreds, rulers of fifties, and rulers of tens:
22: And let them judge the people at all seasons: and it shall be, that every great matter they shall bring unto thee, but every small matter they shall judge: so shall it be easier for thyself, and they shall bear the burden with thee.

A couple of interesting legal concepts can be found here: one delegates powers that one already has; and one delegates to lower-rank subjects who have the ability to take the responsibility. There is no mention of another constraint: that in order to delegate a responsibility, one should have the power to perform such delegation.

Delegation in law is discussed in detail in [Sartor 2005, Chapter 5].

4. Legal Logic in Roman Law, and Greek influences

People who think that legal logic is irrelevant need to look no further than Rome for an argument in their favour. Roman legal thinking has dominated legal thinking in much of Western Europe for millennia, up to our days. Roman lawyers invented many of the legal concepts still used today, and in many cases their solutions to private legal issues were the same as the ones of today. Yet pragmatic Romans were quite uninterested in logic, in fact scornful towards it, as they were towards much of philosophy and pure sciences. Philosophers were often banned from Rome, and had either to teach out of town, or present themselves as teachers of something else, such as rhetoric. However there was at least one philosophical persuasion that had a deep and lasting impact on Roman society, this was Stoicism. Roman scholars of law were exposed to Stoicism, including its logic component, probably through their schools of rhetoric, and this will have the consequences that we will note below. The role of Stoic philosophy in the development of Roman law is discussed in [Wright 1983], but also see [Watson 1995] for an opposite opinion. It can be debated to what extent this philosophy had influence on the spirit of the law and on the logic.

I would say that the contributions of Roman law to legal logic were two: the development of analogical thinking and the development of the first legal ontologies.

Equity and the *intent of the law,* based on analogy*,* were the main principles of interpretation in Roman law, rather than the letter of the law [Bruncken 1917]. Many Roman legal texts can be read in this way: here is an interesting case (legal scenario, *quaestio*) to solve. Here are similar (analogous) cases. There is a common pattern, and an equitable solution for all these cases is so-and-so. The solution may come from a law or another known principle that applies to some of the cases in the similarity set. Similarity of cases does not mean real-life similarity; it means that the cases involve similar rights and obligations. The method of analogical thinking is still very commonly used in legal reasoning.

Opposite to the Romans, the Greeks were the inventors of formal logic in the Western world, and held philosophy in great esteem. But they are not known to have developed a memorable legal theory of their own.

It is believed by some that early Roman legal codes were influenced by pre-existing Greek codes. What is left of the Law of the XII Tables of the 5th C. BC are modern reconstructions based on citations written centuries after the tables had disappeared. On this basis, this does not seem to be a remarkable document from the logical point of view. It is well-organized, each of the 12 tables dealing with a different subject heading, but it shows a low level of conceptualization. There is no uniformity of styles and while in some cases consequences of violations are specified, in others are not. It uses indifferently ECA style or deontic statements.

A much later source is the Institutes of Gaius, a law manual, see http://faculty.cua.edu/pennington/law508/roman%20law/GaiusInstitutesEnglish.htm. It was written in the 2nd Century AD, probably on the basis of earlier manuals (Gaius=Caius was a very common praenomen in Rome but this is the only famous Roman that is known to us with only this name: so it has been questioned whether he was a real person, rather than the traditional name of a compilation). Somehow this manual had a very long life, since it was re-worked by Justinian in the 6th Century AD, and by others in legal texts of the early Middle Ages. In spite of its practical nature, it shows considerable conceptualization and it is probably the first legal source that is based on ontologies, in fact it starts with a single ontology that is supposed to include all legal concepts. At the highest level, Gaius presents a tripartite division of private law (*ius*) into persons (personal status), things (property, succession and obligations) and actions (forms of action and procedure). Notice the echo of the concepts of subject, object and verb already noted in Sumerian law, and which is still used today in access control theory, as mentioned. There are many progressive subdivisions that work towards elementary legal concepts. For example, persons can be free men or slaves. Free men can be born free or freed. The latter can be Roman citizens, Latins, or ‘dediticii’. And so on. Things can be of human law or sacred, etc. Although the ontology developed by Gaius no longer applies today, his method of creating tree-structured legal ontologies is still used. This method has its foundations in the diairesis (=conceptual distinction, classification of concepts) of Greek philosophers. Diairesis starts with general concepts and obtains more specific ones by distinctions. Concepts are defined by their position in the tree. Diairesis was known to the Stoics, and it is an interesting coincidence that Marcus Aurelius, the Stoic emperor, was a rough contemporary of Gaius. Graphical representations of parts of Gaius’ ontology, as well as of more modern legal ontologies, are given in [Fernández Barrera 2009] [Fernández Barrera 2010].

It is interesting to note that Gaius presents rights as incorporeal things. In fact, they can be held and transferred like corporeal things. How can this be connected with modern Hohfeldian concepts of duty, privilege, and the different ‘actions’, this I haven’t figured out yet.

Tree-structured ontologies of this type have implicit the concept known in computing as *inheritance*: if a concept A subdivides in A1, A2 … An then clearly everything valid for A is also valid for each of A1… An. If a legal question comes up concerning a freeborn citizen and the question can be answered in more general terms for all free men, then this latter answer applies.

An interesting section of the Institutes is the one where Gaius explains the Roman system of judicial ‘formulae’: [Poste 1875] discusses how formulae can be understood in terms of syllogisms. I won’t explain how the formula system worked, since there are explanations on this in the web (a good source may be the Institutes themselves, see Book Four, starting in paragraph 39). Essentially, for each type of litigation there were pre-set formulae consisting of several parts where the main elements of the litigation were expressed in precise, stylized language. Formulae could include, in fixed sections: the subject of litigation, uncontested elements, elements to be proven, possible results of the litigation, etc. The formula was set up by a magistrate and then the judge (essentially an arbitrator) had to decide on specific predetermined questions and among specific predetermined outcomes. His mandate was often limited to determine the facts and then choose the corresponding outcome according to the formula. Today, stylized formulae are used in legal documents such as land transfer acts, but not normally in judicial procedures. It seems that if we want to think of automated tools to speed up the judicial process, such as in the e-commerce context, this would be a good method to study [Logrippo 2011].

The formula process was preceded by another, called ‘lege agere’. This ancient and not well documented type of process was characterized by the fact that specific gestures had to be performed and specific words pronounced. Gaius mentions that this type of process became unpopular for this formalism and rigidity and also for the limited number of possibilities it proposed. A minimal error, apparently even an error of pronunciation, could lead to the loss of a case. Does this sound familiar to computer people?

The main legal legacy of Rome is the *Corpus Iuris Civilis*. The part that interests us is the *Digesta* or *Pandectae*. This is a very large (a volume of 1 000 large, tightly printed pages) compilation of excerpts of writings of jurists, covering civil law, compiled by order of Justinian in the 6th Century AD. The excerpts are from writings of the 3rd Century, but edited to reflect the changed thinking. This document is well organized, with major titles divided in subtitles, etc., and many of these titles are similar to the ones that we find in modern texts of civil law.

Examples of explicit logic reasoning or inference are pretty well impossible to find in the Digesta. Logical operators such as *if*, *and*, *or, forall*, *exists* are much more difficult to find in these texts than they are in modern legal texts.

The legal excerpts center on the solution on legal cases, or *quaestiones*. Unlike the cases of modern common law, these were often manufactured for the sake of discussion, in fact many of them seem to be artificial. Perhaps they should be called 'legal scenarios'. Analogical thinking is paramount. This often consists in finding other similar legal scenarios and then coming up with a solution that takes care equitably of all the similar scenarios.

The legal system documented in the Digesta is a compromise between conceptualization and practical considerations. [Watson 1995] points to the paradox of 'high conceptualization and small extent of systematization' for there was prestige in finding intuitively convincing solutions for specific problems but none in systematizing. He cites

> *Digesta 50.17.202. Javolenus, Epistles, Book XL. Every definition in the civil law is subject to modification, for it is rare that it may not be overthrown.*

So definitions are rare in the Digesta and when they are present they are expressed in rather awkward terms. Watson further cites Digesta 26.1.1 and a text by Aulus Gellius (Noctes Acticae 4.1.17, not part of the Digesta) to illustrate this point. Concepts were useful to the extent where they could establish predictability in the system. Much of he conceptual systematization of Roman law was really done in the 19th C.!

The majority of legal opinions in the Digesta are written in an informal ECA style. Sketchy explanations are often introduced by the word '*quia*'= since, because. Here is an example (some translations in this section are from http://www.constitution.org/sps/sps.htm).

> *29.7. 8. Pomponius, On Sabinus, Book II. If a testator, after having bequeathed a tract of land, should dispose of a part of the same, it is held that only the remaining portion is due to the party to whom it was left; because even if an addition was made to said land the legatee would profit by the increase.*

I selected this citation almost at random, but then by examining it I found it to be a good example of the occasional shortcomings of legal thinking in the Digesta. Pomponious has posed a quaestio, found a similar scenario, and posited an equitable solution for both scenarios. I have asked the question of two Ottawa lawyers [1], and promptly they both came up with a much better reason for the same conclusion. The main point is that the author of a testament retains the full right of disposing of any properties until death, hence the legatee can find herself with a part, more or nothing at all. This reasoning is based on clearer legal concepts and has much wider consequences than the reasoning proposed by Pomponius. Apparently brilliant, but really ill-conceived statements such as this may be the price that Roman lawyers paid for not being interested in conceptualization.

Here is another example:

> *7.1.12. Ulpianus, On Sabinus, Book XVII. Julianus presents the following question in the Thirty-fifth Book of the Digest. If a thief plucks, or cuts off ripe fruit which is hanging upon a tree, who will be entitled to a suit against him for its recovery; the owner of the land, or the usufructuary? And he thinks that as fruit does not belong to the usufructuary unless it has been gathered by him, and not when it was separated from the land by another person, the owner has the better right to bring an action for its recovery; but the usufructuary has a right to an action for theft, for it was to his interest that the fruit should not have been removed.*

Perhaps I should not have used this text, because the concept of usufruct will be mysterious for some of my readers (usufruct gives the usufructuary the right to enjoy the fruits of a property, most normally land, while the property remains with someone else). Also there are several concepts at work that are discussed in previous and following sections. According to Julian, the property of the fruits separated by someone who is not the usufructuary is of the owner of the land. We note that the last part of the statement is justified (by Julian) in economic terms, by the *interest* of the usufructuary (an equity-related concept), rather than in legal terms. In the next several lines (not shown above) Ulpian tries to answer the question by discussing situations that in the view of him and other lawyers are similar (but not obviously so). His conclusion remains unclear to me, after having read it in the original Latin and in English and French translations: both the ownership and the right of action remain in suspense (*in pendenti*). It seems that this was considered to be a difficult case.

Julian's opinion demonstrates a fairly usual reasoning process used in jurisprudence: an economic concept (interest) that has legal consequences. I have read somewhere that the right of property of the North American natives on their land was legally questioned because of the fact that their economic organization (primarily hunting-gathering) did not make an efficient use of the land.

There are lots of concepts in Roman juridical thinking, fuzzy as they often are. Forward and backward chaining are possible, but difficult to use because the definitions were so shifting. [Watson 1995] notes that Roman jurists used the distinction between genus and species (see above the mention of diairesis in Gaius): "*Quintus Mucius distinguished five genera of tutela and different genera of possession, and Servius three genera of tutela and four genera of furtum*". The notion of type inheritance is implied: there are rules that apply to *tutela* in general and which also apply to its five genera, unless there are particular reasons to the contrary.

Book 50, Title 16 of the Corpus is particularly interesting from the logical point of view, it is entitled*: On the meaning of words* and it includes many rules and definitions, some of logical interest.

Here is a well-known passage that elucidates the three possible meanings of the word 'aut' = 'or' [2]

> *50.16.124 Proculus libro secundo epistularum*
> *The following words, "So-and-So or So-and-So," are not only disjunctive, but subdisjunctive in their signification. They are disjunctive; for example, when we say, "It is either day or night," for having suggested one of two things, the other is necessarily impossible, since to suppose one disposes of the other. Therefore, by a similar form of words, an expression can be subdisjunctive. There are, however, two kinds of subdisjunctives; one where in a proposition both things cannot be true, and neither of them may be; as, for*

instance, when we say, "He is either sitting or walking," for as no one can do both these things at the same time, neither of them may be true, for example, if the person should be lying down. The other kind of disjunctive occurs in a statement where of two things neither may be true, but both of them can happen to be; for instance, when we say "Every animal either acts or suffers," for there is no animal which neither acts nor suffers, but an animal may act and suffer at the same time.

Modern propositional logic of course knows about these three meanings. The disjunctive operator is the exclusive OR or XOR operator; the first subdisjunctive is the NAND operator, better known as the negation of the conjunction; and the best known is the second subdisjunctive, which nowadays is denoted by the ubiquitous logical operator 'v'.

A close analysis of this explanation shows other interesting points, e.g. the writer alludes at the rule ((A|B) ^ A) → ¬B, where | is the exclusive or operator. This is hardly a significant logical discovery, however. The same law holds for the first subdisjunctive NAND, but Proculus does not mention it. This may be the reason why the NAND is considered as a case of disjunction, in fact it is the disjunction of the negation of the two operands.

So there doesn't seem to be much that could be recognized as predicate calculus or syllogism in the Digesta, but here we have an explicit explanation of propositional logic operators. Aristotelian syllogistic logic, involving predicates and quantifiers, appears to be hardly known in ancient Rome. Apparently at more than one point in history Peripatetic philosophers teaching such doctrines, or even all philosophers, were considered dangerous and banished from the city... But I have mentioned the importance of Stoic philosophy [Wright 1983] and part of Stoic philosophy was what we call today propositional logic. In the text above, Proculus probably cites from some rhetoric textbook inspired by the Stoics. The fact that this was standard knowledge is confirmed by the fact that the term 'subdisjunctive' was also known in medieval times, but only in the last of the meanings described above, which is the one of 'v'.

Watson (1995) reaches essentially the same conclusions and rightly notes that this passage is presented in a vacuum: no applications of the logical laws cited by Proculus are given, neither in the context of the passage itself nor probably in any other part of the Digesta.

Copi (1979) explains how Latins used *vel* for the modern operator 'v', *aut* for XOR, and similar views have been expressed by others, even in Latin texts. This is wrong according to Proculus.

This book also cites a number of legal rules that show the beginning of a trend towards the legal conceptualization that will see developed hundreds of years later. Many of these rules became *brocardi* in the later middle ages. One can recognize fragments of deontic and other modal logic in statements such as the following:

Title 17: *Different rules of ancient law*.

50.17.55. Gaius, On Wills Relating to the Urban Edict, Book II. No one is considered to commit a fraud who does what he has a right to do.

50.17.151. Paulus, On the Edict, Book LXIV. No one commits a wrong against another unless he does something which he has no right to do.

50.17.185. Celsus, Digest, Book VII. No obligation is binding which is impossible.

Concerning delegation, the following text is of interest:

50.17.54. Ulpianus, On the Edict, Book XLVI. No one can transfer to another a right which he himself does not possess.

There are also rules to solve inconsistencies between laws, such as:

50.17.80. Papinianus, Questions, Book XXXIII. In all legal matters, the species takes precedence of the genus, and whatever has reference to it is considered of the most importance.

This rule acquired great importance in the many centuries when the Digesta had the authority of Common Law but could be derogated by specific laws of local authorities: *in toto iure generi per speciem derogatur*, see the discussion on consistency.

50.17.100. Gaius, Rules, Book I. Any obligation contracted under one law is annulled by a contrary law.
Presumably the contrary law would be a following law, in time. This seems to be *lex posterior derogat legi priori,* discussed elsewhere.

It would be interesting to dissect the arguments of the Roman jurists from the logical point of view: formalizing the ontologies they are based on, as well as the detailed logical reasoning.

If any historians of Roman laws happen to read this text, I would like to ask them: was there a progression of legal thinking in Rome that went from formalism at the beginning, towards freer interpretation based on equity later? I seem to recognize such evolution from some of the points discussed above: formalism in the 'lege agere' process; formalism still (although possibly progressively less pronounced) in the formula process; formal ontologies such as Gaius', which could date from much earlier times. But then later legal thinking as documented in much of the Digesta avoids definitions and strict deduction in favor of analogy and equity.

5. Early Medieval European Law: the Example of Salic Law

The dissolution of the Roman Empire led to a multiplication of laws, called *leges barbarorum*, which contained a mixture of elementary Roman law concepts with the concepts of the traditional laws of the invaders, most of them of Germanic origin. They were usually written in a form of Latin. Understandably, the conceptual level was elementary.

Much of the medieval Frankish Salic Law (about 500 CE) is written in ECA style, however the implied ontology is complex, inherited from both Roman law and Germanic laws. The rules are separated in chapters: there is a chapter about murder, one about property etc.

6. Talmudic Law

The Talmud is a collection of Jewish texts of various ages, compiled around 200-500 CE. It contains legal texts of considerable logical sophistication. There are web resources and publications on this; a standard reference is [Jacobs 2006].

7. Islamic Law

I have heard about studies on the logic of Islamic law, but unfortunately my ignorance on this is very deep. Here are some rules found in the Koran [3]. They appear to be written in variants of the ECA style

God recommends that a son receive twice as much as a daughter.
If it were more than two women then they receive 2/3rds of inheritance;
if it was an only female she receives half and to his parents 1/6 each;
if he has no children then his mother takes 1/3;
if he has siblings then his mother gets 1/6;
these rules apply to moneys left after satisfying the will and debts to man and god
to men you have 1/2 what your wives leave if they don't you have children; if she had children then you get 1/4 of what is left after the will or debt.
Your women get 1/4 of what you leave if you don't have any children after satisfying your will and your debts to God and man.
If man or woman passes away and they have 1 sibling female or male they receive 1/6; if there was more than one sibling they split the third of the inheritance

Medieval Arabic and Persian philosophers were very familiar with Aristotelian logic. We all know that much Arabic mathematics was developed in order to solve the fractions given above in complex cases. It would be interesting to know whether logic conceptualization or logic deduction was used by medieval Muslim jurists. However by the little I have read I seem to understand that analogical thinking was paramount, as in the case of Roman jurisprudence. [Hallaq 1997] seems to be quite interesting but I haven't read it.

[Sowa, Majumdar 2003] discuss analogical reasoning in Islamic law and especially the contribution of Taqi al-Din Ibn Taymiyya.

[Joerden 2010] presents several diagrams to illustrate the relationships between concepts of Islamic law.

8. Chinese law: the example of the T'ang code

Citing from: *http://www.tobenot.com/students/061/guides/04_cosmopolitan/03_culture_society/resources/Ebrey_116-131.pdf* (viewed Nov. 2009):

'The earliest of these [codes] to survive intact is the T'ang code, issued in 653. This code contains laws on criminal matters like theft and murder, civil matters like inheritance and ownership of property, and bureaucratic procedures like transmittal of documents.'

This law is remarkable for its clear style and the intricate decisional procedures it describes. Essentially it is ECA, with few legal concepts. But in terms of computer science, one can recognize well-known concepts such as function calls with parameters, loops with arithmetic, if statements, case statements etc. These are found especially in the Action part, where there are rules to calculate the penalties.

Here are two articles:

In cases in which someone at first hit a person for some other reason, and then snatched his goods, calculate the value of the stolen goods to apply the law on robbery by force. When death resulted, the sentence is exile with labor. When he took the goods by stealth, use the law on robbery by stealth, but increase the penalties one degree. When killing or injuring resulted, apply the laws on intentional battery.

Those who plant public or private land they do not have rights to are liable to a beating of thirty strokes for the first mu *or less, increasing one degree for each five* mu*. After the penalty reaches one hundred strokes, it increases a degree for every ten* mu*. The maximum penalty is one and a half years penal servitude. The penalty is reduced one degree if the land had been uncultivated. If force was used, the penalty is increased one degree. The crops belong to the government or the owner.*

9. Common law

9.1 Fraunce's legal argumentation model

The Normans who conquered England in the 11th Century used Frankish and other Germanic laws that were influenced by Roman law. England was ruled by an Anglo-Saxon legal system that had evolved under other influences, but was also essentially Germanic and Roman. After the conquest, a new legal system was started in England that was based on custom and precedent, with limited conceptual background. This is what we know as the Common Law system, distinguished from the Civil Law system of Roman descent. However in the 16th C. the Common Law system came under the influence of Roman jurisprudence and it acquired many concepts from it. In many matters of civil law, the decision reached under the Common Law system or under the Roman Law system are very similar today.

It seems to me that from a logical point of view there is little difference between interpreting a precedent and interpreting a legal opinion found in the Digesta.

A very special position is occupied by the work of Abraham Fraunce [Fraunce 1588]. Aspects of this work are briefly discussed in [Gray, Mann 2003]. These authors say: '*Fraunce was a forerunner of legal knowledge engineering. [His] case diagrams are not unlike the extensive structure of the Latent Damage Law legal advice tree constructed by Capper and Susskind four hundred years later. '* This is a good paper to read for people wanting to see an example of logic-based legal argumentation. As expected, Fraunce's argument is based on powerful assumptions, which are used to reach conclusions by simple syllogisms.

9.2 Logic aspects

There is much literature on the logical analysis of judgments and precedents, but I won't be able to dwell on this subject at this point.

Much could also be said about common law systems seen as learning systems. I'll mention this later.

10. Late Medieval European Law and Scholastic Logic

Here is another huge hole to be filled. In the late middle ages, jurists and cultivated people in general were well informed in Scholastic logic, including the Aristotelian syllogism that was adopted and developed by the Scholastics. Initially, such doctrines came to the West through Arabic translations of Greek texts. Did this knowledge have any influence on legal thinking? This would be interesting to see.

[Bruncken 1917] mentions that late Medieval and early Renaissance jurists considered that their work consisted in finding for every situation an applicable principle in the Digesta, which completely denatured the way Roman jurists considered the same matter.

These jurists tried to conceptualize what they read in the Digesta, without understanding the culture that created the text. This led to complications in some cases, by which they were sometimes vilified (namely by Rabelais, obviously not interested in complex reasoning).

Medieval jurists emphasized the need of conceptualization by distilling and using many short mnemonic statements called brocardi, which essentially were taken as axioms from which conclusions could be drawn. An example is:

Semel heres, semper heres: once heir, forever heir

meaning that once one has accepted an inheritance, this cannot be changed – or otherwise, one cannot be heir for a limited period. This can be represented in terms of the 'henceforth' temporal logic operator:

heir → □ heir

Some such statements were already present in Roman texts, and many of them are still in use, even in countries outside of the immediate influence of Roman law. Most of the short Latin texts that I have included in this text were known as brocardi.

11. Leibniz

Leibniz (1646-1716), a philosopher, mathematician and jurist, was perhaps the first person who thought of formalizing the rules of legal thinking. Apparently he believed that an algebra or logic could be developed to this end.

His contribution to deontic logic is briefly mentioned in the appropriate section below.

I understand that the ideas of Leibniz on this subject are scattered among several of his works. So far I haven't had the time of finding a good summary of these ideas. If someone can help (in almost any west-European language) thanks!

12. 19th Century legal theory in the West

In the 19th C there was a great effort on systematization of legal theory in continental Europe, comparable to advances in sciences and engineering in the same period. Scattered laws were unified in single codes of law, the Napoleonic Code being the main example. Roman law remained the base, however many concepts that are said to come from Roman law were clearly formulated for the first time in this period.

In fact, it is striking to compare a modern textbook on Roman law with an ancient textbook such as the Institutes of Gaius. They describe the same system, but the modern texts are more conceptual and less case-driven. Modern texts also try to describe the historical development, rather than a fixed system.

Some lawyers promoted syllogistic reasoning at this time, others abhorred it.

Mathematical logic started to be developed at the same time, but was essentially ignored by legal philosophy until the 1950s.

Since these notes are being written in Canada, I should mention the work of Pierre-Stanislas Bédard (1762-1829) a Québec politician, journalist, lawyer and judge. He is a well-known personality in the history of Québec and general information on him is available on the web. He is believed to be the author of a manuscript "*Traité du droit naturel démontré par des formules algébriques*", which unfortunately has not reached us. Note that this work was done in complete isolation, and Bédard's dates make him an absolute pioneer in this area. A short report on my examination of a notebook by Bédard can be found here.

13. Hohfeld's ontology of legal concepts

Much of modern western legal theory is constructed in terms of legal concepts that have been developed over the centuries, mainly on the basis of Roman law – although other legal system also knew them.

The American jurist Wesley Newcomb Hohfeld developed a well-known ontology of these concepts [Hohfeld 1913]. I follow the presentation of [Sartor 2006], which has given a brilliant interpretation of this ontology (in fact, the model I will now describe should probably be called Hohfeld-Sartor). The concepts are presented in two squares: the *obligative square* is as follows:

```
Right,Claim-----correlative----------Duty
     |                               |
     |                               |
   opposite                       opposite
     |                               |

     |                               |
No-right, no claim---correlative---- Privilege,Liberty
```

The *potestative square* is as follows:

```
  Power-----correlative----Subjection, liability
     |                        |
     |                        |
  opposite                 opposite
     |                        |
     |                        |
  Disability-correlative---Immunity
```

The connection between the two squares is given by the fact that the rights, duties ... in the obligative set generate powers, subjections... in the potestative set that can change rights, duties, etc. One subject's *right* may be protected through that subject's *power* to activate a sanction against another subject. This is turn can create other *rights, duties*, etc.

Example (from Hohfeld): The correlative of X's *right* that Y shall not enter on the land is Y's *duty* not to enter; but the correlative of X's *privilege* of entering himself is Y's "*no-right*" that X shall not enter. (My addition): If Y violates the duty, X has the *power* to take Y to court.

Example (from Hohfeld): X commits an assault on Y by putting the latter in fear of bodily harm; this creates in Y the *privilege* of self-defense, that is, the privilege of using sufficient force to repel X's attack; or, correlatively, the otherwise existing *duty* of Y to refrain from the application of force to the person of X is, by virtue of the special operative facts, immediately terminated or extinguished.

Example (from Hohfeld): X, a landowner, has *power* to alienate to Y or to any other ordinary party. On the other hand, X has also various *immunities* against Y, and all other ordinary parties. For Y is under a *disability* (i. e., has *no power*) so far as shifting the legal interest either to himself or to a third party is concerned.

Example: If a subject A has a *right* of accessing a data base, then the provider B of the data base has a *duty* to make it available, and A has the *power* of asking access, to which B is *subject*. However if the database is provided by B on a purely voluntary basis, A has *no right* to which corresponds the *privilege* of B to make it available. A is *not able* to ask for access, and B is *immune* from A's claims.

These concepts can be defined in terms of deontic concepts of obligation and permission [Sartor 2005 Ch. 19 and 22, 2006, 2008]. See also [Weinar 2011, Saunders 1989].

Many important legal concepts are based on the concepts just mentioned. Hence, the precise formal expression of Hohfeld's ontology continues to be the subject of interesting research.

[Joerden 2010] presents various networks of legal concept, not related to Hohfeld's and mostly related to criminal law.

However these conceptualizations are not well-known in legal practice, nor in legal theory. They have been criticized by some [Corbin 1921] (often without proposing other possibilities). They are well-known among legal logic researchers, who are interested in legal concepts that can be precisely formulated.

14. From E-Commerce to E-Laws, E-Courts, E-judgments and Computable contracts

The court system could be considered as a human-directed system for analog-to-digital conversion. It converts from continuous reality made of infinite nuances into discrete judicial decisions stating who is right and who is wrong, who gets what rights, imposing discrete penalties, etc.

The need for rapid decision of litigation in contexts of E-commerce or privacy protection may lead to E-laws, to be enforced automatically by E-courts. Suppose for example that a web query tries to access an external data base, but the database access control system denies access on grounds of privacy protection. The requesting agent may have been programmed to appeal this decision by automatically sending a query to an electronic system set up by a body such as a Privacy Commissioner. The latter, after considering the privacy status of the requesting agent and of the data being requested, may prescribe that access should be provided. This e-judgment would be sent to the data base access control system, which would immediately allow access. Today, this is not realistic because it depends on much relevant information being electronically available: the status of the requesting agent, the status of the information requested, and the data access laws. Further, the e-judgment mechanism would have to be programmed. However, all these things are possible and in fact necessary, so it can be foreseen that one day they will be in place. This point is developed in [Logrippo 2011].

It is interesting that some referees of [Logrippo 2011] or of its variants have thrown themselves in eloquent, involved and cultivated disquisitions about the complexity and possible unfeasibility of e-judgments and e-courts, from the social and legal point of view. I hope in some future to be able to present the elements of this discussion in this blog. My point of view however is simple. It starts with the obvious view that e-judgments, based on electronically established facts and clear, unambiguous law, seem to be possible now already. For example, the tax office already issues automatic assessments that have the force of judgments, although it is not recognized as that (solve et repete). My idea is that this concept can slowly (perhaps very slowly) be extended to many other fields, for which examples are given in the paper. Such rapid e-judgments will always be appealable to a human court, of course. This evolutionary view does not pretend to solve at this point complex problems that arise in the general case: the desirability in general of e-courts, the mechanics of e-judgments, etc. In my view, solutions to these problems will grow slowly, as the applications of the concepts of e-courts and e-judgments will grow. Probably e-courts and e-judgments initially will follow very straight logical procedures, more similar to what is now done by the tax office than to what many researchers involved in legal logic are envisioning.

I have touched a raw nerve here: the fear of societies regulated by computers, while it should be clear that the path to such evolution will be regulated and controlled by the society.

[Narayanan, Bennun 1998] contains some articles presenting views on the use of AI methods for automatic judgments.

[Benyekhlef, Gélinas 2005] is a detailed study of application and consequences of on-line conflict resolution.

[Surden 2011] is a very interesting, very detailed legal theory article that attempts to precisely identify the characteristics of the areas of law where automatic judgments may be possible. It is worthwhile to cite from its conclusions:

> *Within the legal literature, this Article pushes back against the view that automation of legal analysis is not possible in any area of law, by providing a means to identify relatively determinate portions even amidst a background of indeterminacy. One observation is that although the task of the lawyer in performing legal analysis mostly involves professional judgment, there is some small subset of legal analysis that is relatively mechanical. A rough heuristic is that where the task of the lawyer is approximately mechanical, it is more likely to be (eventually) automatable via computers.*

The article does not take a position on whether automation is desirable or possible, but it notes possible gains in efficiency. It includes detailed discussion of the existing literature on the subject, and is therefore a good starting point for other reading.

A similar position was taken fifteen years earlier by one of the pioneers of the idea of automation in the legal process, Lucien Mehl [Mehl 1995], a jurist. He mentioned then the existence in the legal world of acts where the decision-taker is essentially bound, and he says that in such cases it doesn't matter if the decision is taken by man or machine. He further mentions the fact that automatic decisions can always be documented by explaining their reasoning, and can be appealed to human authorities.

In IT, distributed resolution mechanisms exist, see for example [Crespo 2007].

[Surden 2011b, Surden 2012] is developing a theory of 'Computable contracts'. "A computable contract is a contractual obligation that has been formulated such that a computer system can both interpret and determine whether the obligation has been complied with".

A similar concept of contracts exists in the framework of the Open Distributed Processing Reference Model, ODP-RM [Putman 2001]. An interesting paper with application to e-commerce is [Merz 1998].

15. Logical aspects

15.1 Proof process

In its general outline, the proof process used in law is the same as the proof process used in mathematics and logic. Basic principles are postulated, some will come from the law, others are established facts and others yet will come from common knowledge, common sense, ethical principles, etc. The proposition to be proven is broken into simpler propositions, and these are broken into simpler ones recursively until one gets to propositions that can be considered to be self-evident (the axioms). All the nodes of the proof tree must be taken care of. Any weak links can be challenged. This process is discussed in [Sartor 2005, Ch. 27].

15.2 Consistency

15.2.1 Consistency in logic and normative systems

In classical logic, a system is consistent if there is no statement A for which it is possible to prove both A and not A. Different rules can cover the same cases with contradictory effects. One rule allows access, another prohibits it. One rule prescribes a minimum penalty of life, the other allows immediate release. Inconsistencies can be immediate between laws, or can be the result of logical derivations. This situation of course cannot be confused with the similar case in which two rules can be applied, but their results are compatible (e.g. one norm stipulates a repayment, another stipulates a fine).

Inconsistency is the only real limit of logic. A system of assertions (legal or otherwise) is consistent if and only if a logical model can be found for it.

Should legal system be consistent? This question may seem absurd to logically-oriented readers. If so, they better read [Haack 2007] for discussion and bibliography on this subject. The question and its answers are not as simple as they are in purely logical systems. However one can ask what is the effect of inconsistency on the principle of *certainty of law*.

As mentioned in [Logrippo 2007], there can be several types and levels of norms in a normative system. Some norms may specify general requirements, others may specify actions and consequences. Some norms may be at the meta-level. Of course, inconsistencies can exist between such levels.

In classical logic, in an inconsistent system anything can be derived, because an inconsistency is false and from false anything can be derived (*ex falso quodlibet*). Therefore, any inconsistency has global implications.

However this conclusion is insignificant in practice. If an enterprise database contains an inconsistency, users normally still believe the rest, although the more inconsistencies are found, the more confidence will decrease. If an inconsistency is found in rules for a complex game, players will still play the game according to the remaining rules. Therefore the users of an inconsistent system tend more to isolate the inconsistent part, than to say that since the system is inconsistent then any rule applies. And in those cases that are not flagrant, users try to iron out inconsistencies by means of interpretation, i.e. by trying to show that norms that appear to be mutually inconsistent apply to different cases. In other words, it may be possible to interpret the norms in such a way that the inconsistency disappears. This is a main occupation for judges and lawyers. In Computer Science there is a literature on the topic of inconsistency in databases and inconsistent data sources.

[Breuker, Valente, Winkels 2005] show, with interesting examples,

15.2.2 Resolution of inconsistencies in normative systems

Some normative systems have built-in conflict resolution strategies. For example, we have seen that in firewalls the first applicable rule is executed, and the others are ignored.

In access control systems, similar to legal systems, all rules in the rule set must be considered when deciding access. This raises the possibility of inconsistencies, if two or more rules lead to different conclusions. XACML allows the system administrator to set these meta-rules. Five such meta-rules are defined, and they are known as: Deny overrides, ordered deny overrides, permit overrides, ordered permit overrides, and first applicable. The last one is essentially the same that is used in firewalls.

Legal systems also have rules for resolving inconsistencies, e.g. in Western jurisprudence, some overriding principles of a quite different nature have been known for centuries, of which the best known are the chronological rule, the specialization rule, the hierarchical rule, and the rule of competence.

The chronological principle (*lex posterior derogat priori*) states that among two laws at the same level, the most recent prevails on the previous one. In fact, it is normally assumed that the successive rule is abrogated by the previous rule. Note the difference between the case where a law is abrogated and the case where the law is declared invalid. In the former case, the law can still have had valid effects during the time of its validity. In IT, the posterior norm will have to be implemented. In an access control system this norm may still be void if it is less restrictive than the previous one, since such systems implement deny override independently of the time in which the regulations came into effect. So if Alice is denied access to certain files, and then later she is allowed, the previous rule will stay in effect unless it is removed. Semantic checks at the time of the introduction of the second rule may point out the existence of the previous rule to the Security Administrator, who will have to remove one of the two rules.

Annulment of a norm can be often impractical in IT systems, because it implies that all the effects of the annulled norm will have to be rescinded. This means rolling back the whole system to the state where it was when the norm started to be implemented, but still keeping the effects of all other norms [Governatori, Rotolo 2008].

The specialization principle (*lex specialis derogat generali*) states that a law covering specific cases takes precedence over a law covering the general case. Now, this seems to be clear if the general and specific law are in the same text, or if the special law has followed the general law. However this principle has sometimes been strengthened to claim that a specific rule takes precedence over a following general rule (*lex posterior generalis non derogat priori speciali*). This principle seems to contradict chronological principle, but such are the discussions of lawyers… To see how this could work in IT, suppose that Alice, a nurse, was given access to the financial records of her hospital's patients. A subsequent hospital regulation states: nurses do not have access to such information. This regulation can be implemented immediately if the hospital has an access control system based on RBAC. Normally the access control system will start denying access to Alice. However a semantic check at the time when the second rule is enacted may bring the previous rule to the attention of the Administrator, with a question: do you wish to keep Alice as an exception?

The hierarchical principle assumes a hierarchy of norms (*lex superior derogat inferiori*). For example, in Canada the Charter of Rights and Freedoms takes precedence over other laws, whether precedent or successive. In IT, the priority of the higher law will not be automatic. Suppose that in a consulting company Allan has access to both Chrysler's and GM's files. At a later stage, the Chinese Wall policy is enacted, by which consultants who work in behalf of a company cannot access files related to a competing company. This policy will have to be implemented in terms of detailed access control rule, otherwise it will have no effect in the access control system.

The competence principle may limit the validity of laws with respect to certain territorial domains or spheres of competence. For example, in the province of Quebec it is allowed for cars to turn right on a red light, but in the city of Montreal, which is in Quebec, this is not allowed. The traffic laws of Iowa are not applicable in Pennsylvania, etc. According to some this principle can be assimilated to the specialization or the hierarchical principle.

There is no priority among these principles in legal systems and so in some cases more than one of them can be applied. So there can be meta-principles that regulate the priority between these principles. It could be decided that hierarchy takes precedence over chronology; this would be a case where a previous higher law takes precedence over a successive lower law. Or we saw above the case where a hierarchically higher law takes priority over a successive specialized law. These are the things lawyers get their money for.

Apart from these principles, the main method for resolving apparent inconsistency in law is to make distinctions. This method takes advantage of the fact that legal definitions are often imprecise and will say that mutually inconsistent rules apply to different cases. To separate the cases, reference is often made to the intention of the law. Similarly, if an inconsistency is found in IT policies, it can be resolved by separating cases.

In all cases, there is question of whether the application of the chosen resolution algorithm or principle may betray the intention of the author of the norms, who may not fully understand all existing conflicts and their possible solutions. So inconsistencies should be seen and understood.

There is an important difference between resolution rules for IT systems and resolution rules for legal systems. In IT, the rules are applied automatically by a program. In legal practice, they are guidance for jurists, lawyers and ultimately the judges who apply them.

An interesting discussion of the use of ‘preferences’ to resolve inconsistencies in law and ethics is presented in [Sartor 2005, Ch. 7].

15.3 Completeness and closure norm

What does it mean for a legal system to be complete? Much has been written on this subject by philosophers of law, and I should remember the extensive monograph [Conte 1962].

Some very easy cases can be resolved by combinatorial analysis. For example, assume an access control system in which rules are of the type (S,V,O) → P, for subject, verb, object, permission. A typical rule in this system could be: if subject A requires to read file X, then this should be permitted. If there are k subjects, m verbs, n objects, then such as system is complete if and only if there is a rule for each of the k x m x n possible combinations. Such systems usually include rules stating that all accesses that are not explicitly permitted will be blocked, This is a closure norm.

Closure norms can be implicit or explicit, in fact most often they are implicit. Consider for example the description of a game such as chess. It includes the description of all possible moves, plus an implicit closure norm stating that no other moves are allowed. Or consider a criminal law system. In the Western world, it is usually assumed to include a norm stating that all behaviors not explicitly disallowed are allowed. No crime without law, *nullum crimen sine lege*. Closure norms make the system complete. However the Judaic or Christian legal systems do not seem to consider Moses law ‘closed’, since they seem to prohibit many behaviors that are not obviously considered in Moses’ law (but see the discussion on arguments *a fortiori*, etc). Another IT example of closure norm is in firewalls that won’t let packets get through unless there is an explicit permission rule for the specific packet type. But in Linux routers the implicit closure rule is positive (let pass through unless explicitly dropped) because the main function of a router is to forward, not to block.

Can the closure norm be derived? It seems that when the rule system contains only positive rules, then the closure norm is negative, and vice-versa. This may have to do with the purpose of the system, whether the purpose is to permit or to forbid. I am not sure what could be said about systems that contain both positive and negative rules.

In some systems there can be closure rules at different levels. For example, in a company there could be a closure rule that applies to executives, e.g. all information access is allowed to executives even if no specific rule is present, however for everyone else the closure rule is to deny access. Closure rules can be different by department. I have seen examples more complicated than this.

However, whenever there are closure norms, there may be question of whether those who specified policies have forgotten something. So this question always makes sense: does the system, including the closure norms, correspond to the *intentions* of the system designer, e.g. the legislator, the system administrator?

Considering all combinations can be difficult or impossible when conditions are added to rules. There can be complex conditions involving intervals where infinitely many values are possible: quantities, temperatures, etc. E.g. file X can be accessed only when the authorization of the requestor is more than $10,000, and only if the requestor sends the request from certain GPS coordinates.

So another way to look at completeness is to ask whether the system has all the rules that are required by the security officer’s or the legislator’s *intention*. Intention is a well-established concept in software engineering and jurisprudence.

Here is an example from law: suppose that Moses wants to find out whether Hammurabi has in fact condemned all types of theft. Then Moses should develop a classification (an ontology) of all behaviors that for him are theft, then he should go through Hammurabi’s law to see whether they are all considered.

Here is an example from IT: suppose that in an enterprise, the Bell-La Padula information protection method must be implemented by means of an access control system. To verify this, it has to be checked whether all cases prevented by Bell-La Padula are covered by access control rules. For example, it has to be checked whether there is an access control rule to prevent Alice, a private, from reading ‘top secret’ files.

A way to ensure that a system of rules is complete with respect to intentions can be to provide a language to specify directly intentions, providing an ontology to define all cases to consider, and then automatically compiling the detailed rules, if this is possible.

Since legal reasoning is informal, incompleteness can be difficult to distinguish from inconsistency. This is because when there is no immediately applicable rule all sorts of other rules can be brought in the discussion.

Legal relevance is a related concept. Blowing your nose may be irrelevant from the point of view of the law of your country or baseball rules. It will be relevant if you are participating in a etiquette context.

Completeness and closure are considered in detail in [Sartor 2005, Ch. 18].[Breuker, Valente, Winkels 2005] present an interesting discussion showing that it can be quite difficult to determine completeness and consistency in real systems of norms, since the underlying ontologies are not given.

15.4 Defeasible logic

We have seen that in some systems there are implicit meta-rules by which some rules take the priority. I.e. in firewalls, the rules that come first take the priority. This is not justifiable in logic terms, because order has no importance in logic. Similarly, in legal system all norms are equally valid unless otherwise said.

Defeasible logic is a logic well-known in AI and to researchers on legal logic. It can be seen as a logical mechanism to manage inconsistency and exceptions. It is a non-monotonic logic first proposed Donald Nute. It involves three types of propositions:

- Hard rules: these specify that a fact is always a consequence of another: all packets from spammers.com must be refused.
- Defeasible rules: specify that a fact is typically a consequence of another: all packets from nuisance.com

must be refused

- Defeaters: specify exceptions to defeasible rules: packets from luigi@nuisance.com must be accepted.

Therefore, before applying a defeasible rule, it must be checked for defeaters. Defeasible logic provides a framework for specifying exceptions and priorities. Interestingly, it is difficult to find hard rules in nowadays' legal systems, a fact that creates a lot of work for lawyers and judges…

The closure norm can be seen as a defeasible norm [4]. It exists in the system, but can be defeated by any other norm. It applies only if no other norm applies. If defeasible logic is not used, the closure norm can be constructed as the norm that applies when the conjunction of the negation of the premises of all other norms is true, and this conjunction may be very lengthy indeed. However, I have mentioned elsewhere that some normative systems can have different closure norms for different roles, levels or departments. I am not sure how to construct such mechanisms by using defeasible logic.

Interesting methodological observations on the relation between defeasible logic and classical logic can be found in [Soeteman 2003].

[Sartor 2005] presents a history of this concept tracing it back to Aristotle. To justify the need of defeasibility in law, Aristotle starts from the fact that the law must be universal, but this must be tempered by the need to consider special cases. But in pragmatic terms, defeasible reasoning finds its justification in the practicality of grouping cases. If a group that covers a majority of situations can be found, the other situations can be described as exceptions. This fact is well-known in programming: many modern programming languages have the concept of exception. It helps making programs clearer and more efficient.

15.5 Deontic logic and deontic concepts

We mentioned that the Moses code uses the concepts of obligation and prohibition, but this is not a use of deontic logic. Deontic logic deals with the relationships among the logical modalities 'obligatory', 'permissible', 'forbidden' etc. There are several variations of deontic logic, however most of them state the duality between permission and obligation, i.e. a behavior x is obligatory if and only if the negation of x is not permitted:

$$O(x) = \neg P\neg(x) \text{ or } \neg O\neg(x) = P(x)$$

Another commonly stated law is that permission is the negation of prohibition, written F here:

$$P(x) = \neg F(x) \text{ or } O\neg(x) = F(x)$$

There are good articles on deontic logic in the Stanford Encyclopaedia of Philosophy and in Wikipedia. The latter article mentions the following about its history (this has been found on August 31, 2009 and of course it may change):

> *Philosophers from the Indian Mimamsa school to those of Ancient Greece have remarked on the formal logical relations of deontic concepts and philosophers from the late Middle Ages compared deontic concepts with alethic ones. In his Elementa juris naturalis, Leibniz notes the logical relations between the licitum, illicitum, debitum, and indifferens are equivalent to those between the possible, impossible, necessarium, and contingens respectively.*

The mention of these insights by Leibniz is very interesting, because part of Leibniz enormous culture were law and ethics, in addition to logic, mathematics, etc. See also [Kalinowski 1972][Joerden 2010].

In Hilpinen (1981) there are essays by von Wright and by Knuttila, where there is discussion on the history of deontic logic. The second author analyzes some sources going back to the 14th century, and claims that the basic laws of modern deontic logic where known at that time. However the authors cited were philosophers with interests in theology and ethics, not law.

The beginnings of the modern history of deontic logic are dominated by the thinking of Mally and von Wright. A discussion of early work in this area is in [Kalinowski 1972].

A belaboured problem in deontic logic is the matter of paradoxes, i.e counterintuitive propositions. It seems that almost none of the many systems of deontic logic invented so far is free of them. *The Web Stanford Encyclopaedia of Philosophy* has a good article on deontic logic, which includes a discussion of paradoxes. Its final conclusion is *: Clearly, there is a lot of work to be done.* This phrase was still there in the revision dated April 2010*.*

[Sartor 2005, Chapters 17 ff] present a pragmatic-legally oriented view of deontic logic.

Deontic concepts have different meanings, depending on whether they relate to machine behavior or human behavior. If I tell a machine that it is obligatory to do something, e.g 'all incoming messages must be acknowledged' the machine will do it unless it is broken or badly implemented. However if software requirements say that some software behavior is obligatory, this is an obligation for the implementer. She may implement it wrong, perhaps entailing consequences for her.

Similarly in law obligation implies necessity if this can be achieved entirely within the legal system, without human intervention. For example, if polygamy is legally prohibited in a country, then it is legally impossible. If a legal act, such as purchase, has certain obligatory legal consequences, such as transfer of property, then the latter will happen. But if a law says that a certain behavior is obligatory, then this or may not be executed by an individual, perhaps entailing consequences.

In other words, obligation of human action is usually implemented as: do it, or else. It can be broken. Obligation not depending on human action means necessity. The concepts are so different that different terms should perhaps be used; a possibility would be to talk about implication or casual dependency and not obligation in cases where obligation means necessity [Boella, van der Torre 2004].

For example, if a specification says: upon reception of message X, message Y must be sent: this requirement seems to be expressed better as a simple implication or as a temporal logic statement than as a deontic logic statement.

Deontic logic is based on a 'square of oppositions', described in the *The Web Stanford Encyclopaedia of*

Philosophy; see
[Beller 2008] for experimental validation of this square. This square is homomorphic to the well-known square of propositional functions and quantifiers, see [Copi 1979]. [Joerden 2010] shows that more complex deontic models can be built as hexagons, octagons, etc. In fact, he shows that such complex deontic models are used in Islamic legal theory.

Some adventurous designers of a Mitel telephony system tried ad one point to design a system based on obligations that could be broken with penalties, then the choice of the action to be taken would be determined by a comparison of penalties. Apart from this, I am unaware of applications of deontic logic in practical IT, although current research aims at applying it to express policies in multi-agent systems. I am unaware of automated tools specifically designed for deontic logic, although these can be obtained by adding axioms to standard logic tools.

I respect the work done with deontic logic, and I am also convinced that deontic logic laws such as the ones mentioned at the beginning of this section are basic for legal and ethical thinking in many legal cultures. Deontic logic can be used to advantage to formalize normative texts that use deontic concepts. However, many normative texts do not use them. I also wonder whether higher order logics wouldn't be more flexible and powerful. Constant higher order predicates can be used to express deontic concepts. This would avoid us all the peculiarities and the varieties of modal logics and would make tools for higher order logics immediately useful for research on legal reasoning.

There will be some discussion on this further in this document.

15.6 Contrary to duty obligations, Chisholm's example

There is a considerable literature on using combinations of deontic logic and defeasible logic to express norms. The discussion is sometimes based on examples such as the following one [Prakken, Sergot 1969]:

> *There must be no fence (i.e. it is forbidden to have fences)*
> *If there is a fence, then it must be white*
> *There is a fence*

The obligation on the second line is often called a 'contrary to duty' obligation, which means a secondary obligation that becomes active after another obligation is violated. So the assignment of a penalty would be a contrary to duty obligation. This example is based on a more complicated example presented in [Chisholm 1963]. Several papers have discussed different formalizations of examples of this type.

I understand that this reasoning can be useful for examples such as the one above, but many laws and regulations are expressed in (or can be translated to) simpler terms. Consider for example the following article from the Hammurabi code:

> *If anyone agrees with another to tend his field, give him seed, entrust a yoke of oxen to him, and bind him to cultivate the field, if he steal the corn or plants, and take them for himself, his hands shall be hewn off.*

Would it be reasonable to translate it to the following:

> *If anyone agrees with another to tend his field, give him seed, entrust a yoke of oxen to him, and bind him to cultivate the field, he must not steal the corn or plants, and take them for himself. If he does this, his hands shall we hewn off.*

Logically, this is more complicated and so there should be a good reason to do this. This reason might be to achieve uniformity in contexts where some other rules are already expressed by deontic statements.

Unfortunately, one finds several authors who claim that there are 'right' and 'wrong' formalisms to solve these problems, a type of discussion that I cannot accept, given that conceptual frameworks have their advantages and disadvantages, can do certain things but have limitations. In addition, the applicability of different frameworks depends on how the problems are stated initially, and what type of conclusions one is interested to draw.

Much discussion in this area is based on contrived examples, which are not clearly related to legal texts or legal reasoning.

15.7 Logic of action and agency

In recent decades, there has been very considerable work on logics of action and agency, and some of the concepts have been applied to express legal concepts. A good reference is: 'The Logic of Action' in the on-line Stanford Encyclopedia of Philosophy. I find the situation confusing because there are several different proposals.

15.8 Machine learning, theory revision and common law

Common law is defined in various ways as a system of law developed on the basis of precedents established by judges. Common law is thus the result of an ongoing process. In Artificial Intelligence, the areas of *machine learning* and *theory revision* seem to provide the tools necessary to study and model this process. In intuitive terms, at some system state, there is a certain set of established rules, which can be expressed in some form of logic. A new judicial decision causes a state transition, into a new state defined by a new set of rules, including the rule just introduced. We now have three cases. The new rule can be independent of the previous ones: in this case, it can be simply added. Or it can be implied by the existing ones: in this case, the new rule is unnecessary, redundant. It is also possible that the new rule is inconsistent with the existing ones. To avoid this inconsistency, the system of rules will have to be adjusted: this is theory revision.

In 2012 the USA Supreme Court stated that the "*The Affordable Care Act's requirement that certain individuals pay a financial penalty for not obtaining health insurance may reasonably be characterized as a tax*". So now we know a new fact, and the USA tax ontology has acquired a new component, with all the logical consequences. All legal concepts and rules that apply to taxes in general now apply to this new tax.

The same ideas apply to other systems of law which can be influenced in some way by judicial decisions. In fact, even the legislative process can introduce inconsistencies, causing the need of theory revision.

I don't know much on this, if some of my readers can find references, I would be pleased to cite them and learn more.

15.9 Feature Interaction

In IT, much study has been generated by a particular type of inconsistency, called Feature Interaction (FI). This subject attracted the attention of designers of telephony features, when they realized that the combination of several features led in some cases to problems, because one feature could disrupt the *intended effect* of another. This could occur on one end of the system, or, worse, between end points.

An example of distributed FI between end points is the following: suppose that Alice does not want Bill to be called from her phone, so it puts Bill's number on her Outgoing Call Screening list. Don can contradict the purpose of Alice by putting Connie's phone on Call Forward to Bill. Now Don can call Bill from Alice's phone by calling Connie, and these calls will not be blocked. FI! Note that there is no error here: each feature worked as expected but the goal of one feature (or intention of a user) was defeated. Also note that this is not easy to fix, because of the distributed nature of the system. Two possible solutions are equally awkward. One would require Alice's phone to communicate to Connie's phone her list of blocked phone numbers at the time of call; but surely Alice doesn't want this to happen! A second would be for Connie's phone to ask Alice's phone for permission to forward. But does Connie want this to happen?

So the concept of *intent* is just as relevant here as it is in law..

FIs have been identified in many systems that are not telecom. Examples exist in security features, home electronics features, etc. They also exist in enterprises and law, in fact it is easy to find legal examples similar to the telecom example above. Such examples occur in cases of delegation, among others. I like this example [5]: Alice does not want her credit card number to be divulged to third parties. It does business with a company who promises confidentiality. However this company subcontracts Connie, who, possibly unaware of the confidentiality issue, may leak the information. This problem is easier to fix, since a good law would force all companies to be responsible about transferring private information to reliable parties only.

A more general case of this problem can be stated in the following way. In a component-based system, some components may have mutually inconsistent requirements. How can this be detected, can they still be combined?

Two types of detection have been studied: in static detection, the feature interaction is detected by formally analyzing the specifications, or policies. This is similar to detecting inconsistency in a normative system. Static detection can be followed by static resolution, a process where inconsistencies are ironed out (akin to performing revisions in legal systems).

Dynamic detection is done at execution time, if mechanisms have been put in place to issue an alarm when a FI is about to occur. Some sort of judicial process can then be executed, where the processes involved in the FI submit their needs to an arbitrating entity, who decides who can go ahead. This arbitrating entity must be provided with all necessary information, and this can be a difficult thing to do in a telecom system.

A series of workshops and conferences has been dedicated to this topic, they started in 1992 (proceedings by IOS Press). [Calder et al. 2003] is a good starting point to study the topic.

Law was invented of course in order to avoid and solve conflicts and so methods to do this are well-known in the legal domain [Bench-Capon, Prakken 2008]. Unfortunately some such methods are impractical for IT, where response times must be very fast and systems are very distributed, without central authorities. The two examples above show this.

16. Concepts common to Computer Science, Software Engineering and Law

Software engineering includes concepts that are also known in law, and for a very long time. Some of these have already been discussed implicitly. Here are some more. Can we conclude that computer science and software engineering are the exact sciences that are most related to jurisprudence?

16.1 Laws as programs

I have suggested that certain types of legal norms can be read as programs. The next question is: what is the entity that should execute these programs? In certain cases (e.g. Hammurabi code or T'ang code, see below) it is clear that the norms are programs for the judicial authority. In others it is clear that they are programs for individuals who find themselves in certain circumstances, or want to reach certain results. But these two aspects are like the two sides of a medal. In practice, many legal statements are written in a generic style, and both aspects are present at once. Suppose that a country wishes to eliminate theft. The principle is stated as: theft is prohibited. This is implemented (refined) in a law, perhaps stating that theft is punished by jail. This law can be seen as a program for judges, to send thieves to jail. However individuals wishing to go to jail see the same law as a program that enables them to achieve their goal.

This situation is mirrored in Software Engineering. For example, the description of an access control mechanism can be used to program the mechanism itself, or the procedure to gain access.

One of the referees of one of my papers has stressed "the necessary inequivalence between laws and computer code". In some sense this is obvious, but there are also strong relationships.

16.2 Refinement from requirements level to operational level

How does one refine Moses norms to Hammurabi norms? How does one translate a requirement into a rule? How can it be recognized that a proposed refinement is in fact correct? Please refer to the section on deontic logic for discussion on implementation of deontic concepts.

16.3 Meta-theory

The concept of meta-theory (which is a theory about another theory) was invented by logicians and mathematicians at the beginning of the 20th C. This concept is well known in software theory. Isn't the constitution of a country a meta-law? It talks about how laws are created, about the validity of laws, etc. It is a law about laws.

Another example is the case of laws that state conditions that company regulations must satisfy.

The concept of meta-law has been known for a long time, but I haven't found any solid reference on the subject.

One difficulty in the formalization of legal concepts is that they often traverse several of these layers.

16.4 Conformance

The concept of *conformance* and *compliance* is well-known in software engineering, especially in relation to software testing and protocol testing. A specification document may identify compulsory and optional features of a product. The product will have to satisfy these requirements, as well it might have additional characteristics.

Similarly, a company regulation will have to satisfy law requirements. We have done some research on this, and we have treated conformance checks as consistency checks between different levels of law. But surely this subject needs more research.

16.5 Layered models

Layering of protocols is a fundamental concept in telecommunications protocol engineering. Similar concepts are used in law. I'll take an example from international affairs, where it is customary that a head of state communicates with another head of state by sending a message via an ambassador.

So a head of state wants to send a message to her peer entity, another head of state. For her, this is direct communication however in reality this has to go through numerous underlying steps. Her office will submit the message to the Department of External Affairs, including instructions. The Department of External Affairs is a service provider for the head of state. It provides a well-defined service, which is transmittal of messages to other heads of state. But this is not done directly. External has inside itself another service provider, the External Relations Office. This office performs a well-defined service, which is transmittal of messages to embassies around the world. Although the office thinks in terms of direct communication to the ambassadors abroad, this is not done directly, it is done through an underlying service provider, and so on up to a lowest level that actually transmits the information. So initially the message goes down a hierarchy up to the lowest level. At each level, a layer consisting of an addressed envelope and instructions to the peer entity abroad is added to the message. When the package gets to the other country, it will start going up towards the other head of state. Each level will strip the envelope directed to it and read the instructions directed to it from its peer in the other country. Eventually the message will get to the ambassador, who will strip the envelope and read the instructions directed to her. She now knows that she has a message to be delivered to the other head of state. What will be left of the package at this point will be the envelope directed from the first head of state to the other head of state, containing the initial message. This can now be delivered, the other head of state will tear the initial (which is also the final) envelope and read the message.

(Note that in this structure the two Departments of External Affairs do not communicate, since ambassadors can visit heads of state directly.)

Telecom protocols work just like this.

16.6 Event projections and aspect-oriented design

Here is an example of projections of actions on different normative dimensions: a group of business people play bridge while discussing deals and signing contracts. Some of their actions project on the normative dimension of bridge rules, others project on the normative dimension of civil or common law, others may be relevant on the ethical dimension. Some of their actions may project on two or more dimensions.

Aspect-oriented design is an application of similar thinking.

16.7 Normative systems for e-societies

Societies are ruled by laws and customs. Those who do not abide by them are punished or emarginated. Electronic societies can span the world and these enforcement methods may not be effective, as we all know by our constant fight against spam and viruses. Similarly, we can get into what appears to be a bona fide electronic agreement with a party, and then we are helpless when we see that the party does something we don't believe was agreed.

A model for peer-to-peer agreements and electronic societies may be provided by international law, which is constituted mainly of customs and multilateral conventions.

Collaboration of distributed systems can only be achieved if they all use certain common mechanisms. For example, interprocess communication in a set of distributed processes depends on all the processes using *common* synchronized methods. Such tacit agreements constitute customs.

In law and society, many customs exist that are respected by all who want to be considered reliable citizen. For example, if Alice lends a book to Bob, Bob is supposed to check back with Alice if she wants to lend it to Carl. However in telephony Bob can automatically forward to Carl a call from Alice without checking with her. So Alice may find herself talking to Carl, although she may have Carl on her incoming call screening list. This is a well-know example of interaction of telephony features that is possible because of violation of a rule that is well understood in society, but not so in telephony.

Unequal agreements with network entities such as Skype, who dictates terms of operation, may be similar to protectorates. Consortia such as Apache, where participants can collaborate in the evolution of the system, are more similar to alliances. In computing systems we can have multi-faceted situations where a user can be simultaneously in many such agreements, again creating the possibility of inconsistencies. People routinely click the ‘Accept all conditions’ box, happily no one compares all clauses that have been accepted.

Network entities will associate with other entities they trust, and this will establish societies of mutual trust. Concepts and modes of operations will be created in these societies, some of which will slowly gain acceptance, thus enlarging the societies. The concept of Web of Trust is an application of this idea.

16.8 Laws, standards and the political process

The political process that produces both laws and IT standards may have an effect on logical clarity. Each party may have a clear logical idea of what the law or the standard should say, but in order to reach consensus it may be necessary to express the law in a way that allows several interpretations. It is then to implementers (judges or programmers) to decide, and this may lead to a long period of incertitude.

17. Argumentation models, Artificial Intelligence, and automated legal decision systems

[Klug 1951] discusses the following models of argumentation used in the legal process: ‘*argumenta a simile* (= analogical)*, e contrario, a majore ad minus, a minore ad majus, a fortiori, ad absurdum*’, all known with Latin names because of their very long history not only in the legal world, but also in philosophy and ethics. Klug cites different definitions given by different authors for each of them, along with legal examples, and attempts to provide different interpretations in logic. [Haack 2007] discusses the role of *abduction* in legal reasoning. These thinking models are also common in science, however in modern science they are used as the source of conjectures, to be corroborated by experimentation or more precise analysis. Argumentation models, as well as the relation between argumentation and inference, are discussed in [Sartor 2005 Ch. 26 ff.].

Here are some simple examples [6].

In the Qu’ran, the use of wine is forbidden because of its intoxicating effects. Islamic tradition then forbids the use of intoxicating drugs. According to Klug’s definitions, this is an application of the argument ‘a minore ad maius’ (from minor to major), or possibly of the similar argument ‘a fortiori’ (for stronger reasons). This reasoning can be modeled in logic with the help of an ontology, which in this case is a partial order between intoxicating media, including the fact: wine < drugs. Then we need an axiom, e.g:

$x< y \rightarrow (\mathrm{Forbidden}(x) \rightarrow \mathrm{Forbidden}(y))$

If we wish to model the fact that performing a more serious offence involves a more serious penalty, then we need an ontology for penalties, with a partial order among them, and a corresponding axiom.

Similar reasoning can be used in IT. For example, suppose that in an enterprise Confidential files are password-protected, however Top-Secret files are not. The enterprise’s security policies can be considered incomplete by this reasoning. This can be implemented in a completeness checking program by using the same idea: partial orders between document classifications and protection schemes, as well as an axiom.

Other argumentation schemes mentioned by Klug can be modeled in similar ways, at least in their simpler uses. In each case, the main task is to create satisfactory ontologies, for example an ontology of similarities in the given domain in the case of reasoning by analogy.

Note that the use of these arguments in criminal law requires to nuance in some way the principle ‘no crime without law’. In principle, almost every behavior can end up being considered to be legislated by analogy or similarity. This also can make it impossible for law to be incomplete [Conte 1962].

The argument ‘ad absurdum’ corresponds to the proof by contradiction and can be modeled in logic by showing that if the contrary of the desired conclusion is assumed, then a contradiction is derived, or no logical model exists for the system.

Often in the legal literature one can find argumentation models that are based on complex intuitions that are difficult to formalize. It is also common for several of the mentioned argumentation models to be used at once, packed together in short phrases, so a detailed analysis may be necessary to tentatively recognize reasoning patterns.

This is where Artificial Intelligence methods have their place. For example, one can try to automate analogical thinking by calculating degrees of similarity of legal situations. An AI-oriented view of legal reasoning is presented in the book of [Sartor 2005]. This book analyzes many types of legal reasoning with an AI approach.

I have found the most convincing explanation of analogical thinking in computer science terms in [Sowa, Majumdar 2003]. This paper claims that deductive, inductive and adbuctive thinking are applications of analogical thinking, which is explained in terms of conceptual graphs. It cites the 14th-C Islamic jurist and legal scholar Taqi al-Din Ibn Taymiyya for his analysis of analogical thinking in law, and for his comparison of the latter with Aristotelian syllogism.

While classical deductive logic produces certitudes, argumentation models produce possibility and probability. This observation establishes their place in the legal process. Since the latter must produce a decision, argumentation models must be enclosed in deductive processes expressed in classical logic [Soeteman 2003]. This situation is similar to the one found in automated control systems. For example, consider a nuclear reactor shutdown system: it takes a number of continuous parameters, which indicate the likelihood of a meltdown: pressure, temperature, and possibly a database of dangerous scenarios; it must produce a binary decision, shutdown or not.

Legal decision systems are discussed in [Yannopoulos, Reed 1997][Oskamp, Tragter 1997]. The construction of such systems is possible in limited areas of law, where definitions are precise and rules are clear. I briefly mention the role of AI in legal decision-making in [Logrippo 2011].

A comprehensive survey of the role of logic in computational models of legal arguments, with many citations, can be found in [Prakken 2001]. This paper develops a view based on four layers: a logical layer (constructing an argument); a dialectical layer (comparing and assessing conflicting arguments); a procedural layer (regulating the process of argumentation); and a strategic, or heuristic layer (arguing persuasively).

The possible use of AI methods to model the common law system was mentioned earlier.

It appears that the field of legal logic, which has ancient roots in legal practice, philosophy and legal theory, has now come under the protectorate of Artificial Intelligence, see [Bench-Capon et al. 2012].

18. From Law to Software

Some laws must be implemented in software. In other words, software requirements have to be derived from laws and implemented in software. This process is particularly important in areas such as: fiscal law, e-commerce law, privacy law.

Essentially, the law has to be translated into some formalism for expressing software requirements in some software development methodology, e,g, UML. This is called 'requirement extraction'. When this is done, the normal software development process can take over. So the research in this topic focuses on the requirement extraction phase [Hassan, Logrippo 2009].

The yearly RELAW workshop, an affiliated workshop of the IEEE Requirement Engineering conference, is dedicated to this topic, and its proceedings can be found in the IEEE database.

19. Tools

This section will discuss automated deduction tools. The tools I will consider fall in the following categories:

19.1 Logic programming and constraint logic programming (e.g. Prolog, Constraint-Prolog)

19.2 Logic checkers and satisfaction algorithms (e.g. Alloy)

19.3 State exploration (e.g. SPIN)

19.4 Theorem provers (e.g. Coq)

20. What else is there *(too much…)*

I have already mentioned that legal thinking is dominated by political, ethical, social, economic concerns. In the western culture, equity and the intent of the law remain two main interpretation criteria. Argumentation models, which lead through 'discourses', or chains of arguments, that are plausible, intuitively 'logical' but have no formal logical basis, dominate the legal process, see [Haack 2007] (a remarkable philosophical paper with many references), also [Kalinowski 1983][Sartor 2005]. In principle argumentation models can be expressed in formal logic with the help of suitable ontologies, although complexity and consistency may be challenges. As we have already mentioned, argumentations are dominated by complex assumptions or domain axioms (the ontologies), connected by simple logic, which may all be left unspecified, possibly because they are too complex to specify. This type of legal thinking has been modeled by using artificial intelligence methods. However one might wonder to what extent mechanising traditional legal thinking will perpetuate its faults, among which are occasional obscurity, unpredictability and inconsistency. Although human judgment is necessary in many areas of law, in others clarity, predictability and consistency, as well as simplicity, are just as necessary. I have already mentioned the growing areas of e-commerce and privacy protection, where decisions of legal relevance must be reached and enforced impersonally and at electronic speed. It seems to me that clear formal logic methods should be used in these areas.

If one could map out various methods of reasoning in various areas of the legal process, surely one could find areas where formal logic is prevalent. The believer might detect an extremely slow process of increased conceptualization in legal thinking over the millennia, conceptualization which is a necessary premise for the use of logical inference methods. Already for some time now, in the fiscal legal systems of many countries we have very clear conceptualization and inference rules, so that initial assessments, fines, etc. are calculated by computer.

'Classical', two-valued logic seems to be prevalent: either one owns something, or one doesn't. Either one is guilty, or one isn't. However probabilistic thinking can also be used in the chain of reasoning.

Probably the most comprehensive book ever written on legal reasoning is [Sartor 2005], an 844 pages volume. It features chapters analyzing many types of legal reasoning. Happily, it is also very well written and pleasant to read, with good examples on almost every page. I could have cited it much more. A good 'motivational' article for this general research area, with many suggestions for research and many references, is [Bench-Capon, Prakken 2008]. A more recent and very useful survey paper is [Bench-Capon et al. 2012]. Other comprehensive recent books, covering many subjects discussed above, are [Haage 2005] [Stelmach, Brozek 2006] [Joerden 2010]. This flurry of substantial publications witnesses the increasing attention that our field is receiving.

I can't resist concluding all this without attempting my own philosophy, which I will base on social and scientific communication concepts (one of many possible explanations). Many processes of social communication are based on reproducibility. Natural sciences and engineering are based on reproducibility and predictability. However a person's reasoning is based on chaotic processes that depend on many things that go on in the person's mind. So a person must state her reasoning in a way that she and others can understand and verify, reproduce, remember. Logic and mathematics are built on concepts and processes that many people are able to understand and reproduce, so they are good means to communicate in certain environments and situations. Predictable and reproducible processes, such as scientific and engineering processes, can be based on logic and mathematics. Other ways to communicate exist, for example some are based on feeling and sentiment, and they can also produce understanding and predictable, reproducible results; this is the area of social sciences and other areas such as argumentation. Legal thinking stands at the intersection of several of these reasoning and communication processes.

Appendix 1. Tammelo's "Manifesto of legal logic"

Ilmar Tammelo (Narva, Estonia, 1917 –Sydney, Australia, 1982) wrote some of the first books on the use of modern formal logic for the analysis of legal reasoning. [Tammelo 1978] contains a number of interesting examples taken from real-life judgments. The book concludes with the following text which is worth citing in full:

> *The nature and role of legal logic can be epitomized in the form of the following manifesto:*
> *(1) Legal logic is indispensable for any rational treatment of legal problems.*
> *(2) Legal logic is complementary to other disciplines of fundamental legal thought.*
> *(3) Legal logic is not a source of the material contents of law but an instrument of legal thought.*
> *(4) Legal logic is a prerequisite of utilisation of modern technology in the field of law.*
> *(5) Legal logic is indispensable for promoting expediency, efficiency, and integrity in legal reasoning.*

References

(Note: This list is a personal jumble, it includes publications that I have found useful, as well as others that I would like to know better. I have privileged sources available in the web; but I have also cited certain sources that are difficult to find. Needless to say, it is quite possible that better sources exist in many subjects and if so I would like to know from my readers).

Carlos Alchourròn, Eugenio Bulygin (1972): Normative Systems. Springer.

Steven Barker (2012): Logical Approaches to Authorization Policies. In: Logic Programs, Norms and Action LNCS 7360, 2012, 349-373.

Trevor J. M. Bench-Capon, Michal Araszkiewicz, Kevin D. Ashley, Katie Atkinson, Floris Bex, Filipe Borges, Danièle Bourcier, Paul Bourgine, Jack G. Conrad, Enrico Francesconi, Thomas F. Gordon, Guido Governatori, Jochen L. Leidner, David D. Lewis, Ronald Prescott Loui, L. Thorne McCarty, Henry Prakken, Frank Schilder, Erich Schweighofer, Paul Thompson, Alex Tyrrell, Bart Verheij, Douglas N. Walton, Adam Zachary Wyner (2012): A history of AI and Law in 50 papers: 25 years of the international conference on AI and Law. Artif. Intell. Law 20(3): 215-319

Trevor J.M. Bench-Capon, Henry Prakken (2008): Introducing the Logic and Law corner. Journal of logic and computation, 18(1), 1-12

Sieghard Beller (2008): Deontic reasoning squared. In B. C. Love, K. McRae, V. M. Sloutsky (Eds.), Proc. 30th Annual Conf. of the Cognitive Science Society. Cognitive Science Society, 2103-2108.

Karim Benyekhlef, Fabien Gélinas (2005): Online dispute resolution. Lex Electronica (10), 2, 1-129.

Guido Boella and Leendert van der Torre (2004): Fulfilling or violating obligations in multiagent systems. IEEE/WIC/ACM International Conference on Intelligent Agent Technology (IAT 2004) 483-486.

Joost Breuker, André Valente, Radboud Winkels (2004): Legal Ontologies in Knowledge Engineering and Information Management. Artificial Intelligence and Law 12: 241-277.

Joost Breuker, André Valente, Radboud Winkels (2005): Use and Reuse of Legal Ontologies in Knowledge Engineering and Information Management. In: V.R. Benjamins et al (Eds): Law and the Semantic Web, LNAI 3369, 36-64.

Ernest Bruncken (1917): Science Of Legal Method, Boston Book Company. Chapter IV, Sections 3 and 4

Muffy Calder, Mario Kolberg, Evan H. Magill, Stefan Reiff-Marganiec (2003): Feature Interaction: A Critical Review and Considered Forecast. Computer Networks, 41(1) 115-141.

CBC - Canadian Broadcasting Corporation (broadcast by Paul Kennedy) (2011): http://podcast.cbc.ca/mp3/podcasts/ideas_20111019_45513.mp3

Gaetano Carcaterra (1996): Corso di filosofia del diritto, Bulzoni Ed.

Roderick M. Chisholm (1963): Contrary to duty imperatives. Analysis, 24(2), 33-36.

Amedeo G. Conte (1962): Saggio sulla completezza degli ordinamenti giuridici. Giappichelli.

Arthur Corbin (1921): Jural Relations and Their Classification. Faculty Scholarship Series. Paper 2873. http://digitalcommons.law.yale.edu/fss_papers/2873

Rui G. Crespo, Miguel Carvalho, Luigi Logrippo (2007): Distributed Resolution of Feature Interactions for Internet Applications. Computer Networks 51 (2) 382-397.

Irving M. Copi (1979): Symbolic Logic. Prentice-Hall

Ronald Dworkin (1978): Taking rights seriously. New impression with a reply to critics. Duckworth, London.

Meritxell Fernández Barrera (2009): Legal ontologies - Historical origins and state of the art. http://lexsummerschool.files.wordpress.com/2009/09/sate-art-legal-ontologies.pdf (Consulted Jan 2011).

Meritxell Fernández Barrera, Giovanni Sartor (2010): Classifications and the Law: Doctrinal Classifications vs. Computational Ontologies. EUI Working Papers LAW No. 2010/10. July 2010. http://papers.ssrn.com/sol3/papers.cfm?abstract_id=1698686 (consulted Jan 2011).

Abraham Fraunce (1588): The Lawyer's Logic, reprinted by The Scolar Press Limited, Menston, 1969. Original title: Lawiers Logike.

Guido Governatori, Antonino Rotolo (2008): Changing legal systems: Abrogation and Annulment. in: R. van der Meyden, Leendert van der Torre (Eds.). Deontic Logic in Computer Science, DEON 2008, Proc. of the 9th International Conference, DEON 2008. LNAI5076, Springer, 3-18.

Pamela N. Gray, Scott Mann, S. (2003). The Fraunce (1588) model of case-based reasoning. In Proceedings of the 9th international Conference on Artificial intelligence and Law (Scotland, United Kingdom, June 24 - 28, 2003). ICAIL '03. ACM, New York, NY, 89-90.

Stefan Gruner (2010): Software Engineering Between Technics and Science: Recent Discussions about the Foundations and the Scientificness of a Rising Discipline. J.l for General Philos. of Science / Zeitschrift für allgemeine Wissenschaftstheorie 41/1, 237-260, Springer Verlag, June 2010.

Susan Haack (2007): On Logic in the Law: "Something, but not All". Ratio Juris, 20 (1), 1-31.

Jaap Hage (2002). What to expect from legal logic?. J. Breuker e.a. (eds.), *Legal Knowledge and Information Systems. Jurix 2000: The Fourteenth Annual Conference*. IOS-Press, Amsterdam, 77-87

Jaap Hage (2006): Studies in Legal Logic. Law and Philosophy Library, Springer, 2005.

Wael B. Hallaq (1997): A history of Islamic legal theories: an introduction to Sunni usul al-fiqh. Cambridge University Press (available in Google Books)

Waël Hassan, Luigi Logrippo: A Governance Requirements Extraction Model for Legal Compliance Validation. In Proc. IEEE 17th International Requirements Engineering Conference (RE'09): RELAW Workshop. Atlanta, GA. Sep. 2009. (Electronic proceedings, 6 pages)

Risto Hilpinen, (Ed.) (1981): New Studies in Deontic Logic – Norms, Actions, and the foundation of Ethics. University of Turku, Turku, Finland: D. Reidel Publishing Company.

Wesley Newcomb Hohfeld (1913): Some Fundamental Legal Conceptions as Applied in Judicial Reasoning. Yale Law Journal 16. This classic paper is available in a number of contemporary re-printings.

Louis Jacobs (2006) : Studies in Talmudic Logic and Methodology, Vallentine Mitchell Publishers.

Anthony J.I, Jones, Marek J. Sergot (1992): Formal Specification of Security Requirements Using the Theory of Normative Positions. In: Deswarte, Y., Quisquater, J.-J., Eizenberg, G. (eds.) ESORICS 1992. LNCS, vol. 648, pp. 103–121. Springer, Heidelberg (1992)

Anthony J.I. Jones, Marek J. Sergot (1993): On the characterisation of law and computer systems: The normative systems perspective. In: Deontic Logic in Computer Science: Normative System Specification, J.-J.C. Meyer and R.J. Wieringa (Eds), Wiley.

Jan C. Joerden (2010) : Logik im Recht, 2te Aufl., Springer.

Georges Kalinowski (1972): La logique des normes. Presses Universitaires de France.

Georges Kalinowski (1983) : La logique juridique et son histoire. In Simposio de Historia de la Logica. Anuario Filosofico de la Universidad de Navarra Pamplona, 1983, vol. 16, n^{o}1, pp. 331-350. http://dspace.unav.es/dspace/retrieve/4798/license.txt (consulted December 2009).

Ulrich Klug (1951): Juristische Logik, Springer-Verlag.

Luigi Logrippo (2007): Normative Systems: the Meeting Point between Jurisprudence and Information Technology? In: H. Fujita, D. Pisanelli (Eds.): New Trends in Software Methodologies, Tools and Techniques – Proc. of the 6th SoMeT 07. IOS Press, 2007, 343-354.

Luigi Logrippo (2011): From e-business to e-laws and e-judgments: 4,000 years of experience. CYBERLAWS 2011, Proc. of the Second International Conference on Technical and Legal Aspects of the e-Society, Guadeloupe, Feb 2011, 22-28.

Giuseppe Lorini (2003) : Il valore logico delle norme. Atlantica Editrice.

Lucien Mehl (1959) Automation in the legal world: from the machine processing of legal information to the "Law Machine". National Physical Laboratory Symposium No 10, Mechanisms of Thought processing (2 vols.). London: HMSO.

Lucien Mehl (1995) À quelles consitions une décision juridictionnelle, administrative ou privée, peut-elle être totalement ou partiellement automatisée? Publication de l'association québecoise pour le développement de l'informatique juridique.

Sharad Malik, Lintao Zhang (2009) : Boolean Satisfiability – From Theoretical Hardness to Practical Success. Comm. ACM 57 (8), 76-82.

Michael Merz, Frank Griffel, M. Tuan Tu, Stefan Müller-Wilken, Harald Weinreich, Marko Boger, Winfried Lamersdorf (1998): Supporting Electronic Commerce Transactions with Contracting Services. Int. J. Cooperative Inf. Syst., 249-274.

Ajit Narayanan, Marvyn Bennun : Law, Computer Science and Artificial Intelligence. Intellect, 1998.

Anja Oskamp, Maaike W. Tragter (1997): Automated Legal Decision Systems in Practice: The Mirror of Reality. Journal of Artificial Intelligence and Law. 5 (4), 291-322.

Edward Poste (1875): Elements of Roman law by Gaius. Clarendon Press. Pages 522-530.

Henry Prakken, Marek Sergot (1996): Contrary-to-Duty Obligations. Studia Logica, 57, 91-115.

Henry Prakken, Giovanni Sartor (2001) : The role of logic in computational models of legal argument – a critical survey. In: A. Kakas, F. Sadri (eds.): Computational logic from logic programming into the future. Springer-Verlag 2001.

Janis R. Putman (2001): Architecting with RM-ODP. Prentice-Hall

Jean Ray (1926) : Essai sur la structure logique du code civil français. Alcan (Paris).

Giovanni Sartor (2005): Legal Reasoning: A Cognitive Approach to the Law. Published as Chapter 5 in: A treatise of legal philosophy and general jurisprudence, Springer (available on-line on the Springer site).

Giovanni Sartor (2006): Fundamental legal concepts: a formal and teleological characterization. Artificial Intelligence and Law 14 (1). See also a related presentation: http://deon2008.uni.lu/Sartor.pdf

Giovanni Sartor (2009): Legal concepts as inferential nodes and ontological categories. Artificial intelligence and Law, 17: pp. 217–51.

Giovanni Sartor, Pompeu Casanovas, Mariangela Biasiotti and Meritxell Fernández-Barrera (2011): Approaches to Legal Ontologies – Theories, Domains, Methodologies. Springer.

Kevin W. Saunders (1989): A Formal Analysis of Hohfeldian Relations, 23 Akron L. Rev. 465.

Marek J. Sergot, Fariba Sadri, Robert A. Kowalski, Frank R. Kriwaczek, P. Hammond, H. T. Cory (1986): The British Nationality Act as a Logic Program, in Comm. ACM, 29 (5), 370–386.

Jerzy Stelmach, Bartosz Brozek (2006): Methods of Legal Reasoning. Springer.

John F. Sowa, Arun K. Majumdar (2003). Analogical Reasoning. In: A. Aldo, W. Lex, & B. Ganter, eds. Conceptual Structures for Knowledge Creation and Communication, LNAI 2746, Springer-Verlag, pp. 16-36

Arend Soeteman (2003): Legal Logic? Or can we do without? Artificial Intelligence and Law 11, 197-210.

Harry Surden (2011a) The Variable Determinacy Thesis, 12 Columbia Science and Technology Law Review 1, (91 pages), (April 2011)

Harry Surden (2011b): Computable Contracts (Presentation) http://www.youtube.com/watch?v=KLAE_SKMeAY

Harry Surden (2012): Computable Contracts. UC Davis Law Review, Vol. 46, No. 629, 629-700.

Ilmar Tammelo (1969): Outlines of Modern Legal Logic. Franz Steiner Verlag, Wiesbaden 1969.

Ilmar Tammelo (1978): Modern Logic in the Service of Law. Springer-Verlag.

André Valente (1995): Legal knowledge engineering - A modelling approach. IOS Press.

André Valente (2005): Types and Roles of Legal Ontologies. In: V.R. Benjamins et al. (Eds.): Law and the Semantic Web, LNAI 3369, 65-76.

Bart Verheij, Jaap C. Hage; H. Jaap Van Den Herik (1998): An integrated view on rules and principles. Artificial Intelligence and Law, 6 (1), Mar. 1998 , 3-26.

Alan Watson (1995): The Spirit of Roman Law. University of Georgia Press. Chapter 12: A Central Indefiniteness; and 13: Legal Isolationism III.

Radboud G.F. Winkels (2010): Legal ontologies. http://www.lri.jur.uva.nl/~winkels/LegalOntologies.html (Consulted Jan 2011).

Tom Van Engers, Alexander Boer, Joost Breuker, André Valente (2008): Ontologies in the Legal Domain. Chapter 13 of: *Digital Government*. Springer.

Leif Weinar (2011): Rights. In Stanford Encyclopaedia of Philosophy. http://plato.stanford.edu/entries/rights/ (Consulted Aug. 2011).

George Wright (1983) : Stoic midwives at the birth of jurisprudence. Am. J. Juris. 169 1983, 169-188.

Georgios N. Yannopoulos, Chris Reed: Modeling the Legal Decision Process for Information Technology Applications in Law. Kluwer Law International, 1997.

Footnotes

1. Thanks to Ottawa lawyers Chris Arnold and Julie Audet for corresponding with me
2. Mirella Capozzi and Carlo Cellucci of the University of Rome helped me with the interpretation of this text
3. Thanks to Waël Hassan for the following citations
4. Thanks to Guido Governatori for this insight
5. Thanks to Waël Hassan for this example
6. Thanks to Sofiene Boulares for the two following examples

Feedback, interventions and discussion

This section includes feedback I have received from identified people and from anonymous referees. Such feedback contains valuable points of discussion and so it should not remain hidden in my personal files.

1. Discussion with Peter Denning (January 2010)

http://cs.gmu.edu/cne/denning/

Peter Denning wrote:

In skimming the introduction, the following thoughts occurred to me:

(1) US law now has algorithms written in to it. The IRS code,for example, contains algorithms for computing the deductions Congress has decided to allow. As far as I can tell, they embed the algorithm because in their conference committees they negotiated using spreadsheets ("Well, I'll support your amendment if you'll reduce the tax loss according to our spreadsheets.") They play with the numbers on the spreadsheet until the bottom line comes out right. By that time there is no longer a "principle" involved in arriving at the amount of the deduction, so they instead just state the algorithm.

(2) There is an attitude among lawmakers that they are writing algorithms to control (govern) society. They try very hard to anticipate all the little ways people might try to circumvent the intent of the law and write in clauses to block those actions. They literally think they are "programming society". But of course there is a huge difference between programming a computer (a deterministic machine) and governing a society (a network of conversations and practices that can take unpredicted directions at any time). There is a huge danger in failing to make the distinction and a hubris in thinking that programming society is possible. Laws seem to work better when they state their intent and give their principles, than when they try to prescribe or block behaviors.

L.L.'s answer:

Here are some tentative answers to your insightful remarks:

Your remark (1): In software, we know how to keep track of principles. They are expressed as invariants whose truth must be maintained, or as goals that must be achieved. So something can be done for tax legislation, as well, if the legislators can agree on what the principles are. A model checker can try to see whether the principles have been respected in the algorithm. But what happens in bad law mirrors what happens in bad software: it gets complicated, details take over, and no one can figure out what the invariants are, or even what the program is really supposed to do. In OSI protocols, every protocol had to have a service specification. But same layers were poorly designed, and there was little difference between the protocol specification and the service specification. Similarly for Internet protocols: if there are any service specifications at all, they are usually encumbered by protocol details. Poor design IMOMO. Many areas of engineering are more advanced than IT (or law) in this respect, and this would be a long discussion…

Your remark (2). Lawmakers *are* writing algorithms to control certain behaviors in society. Now we get into politics, but almost every political philosophy asserts that certain behaviors must be controlled, programmed. Already the ancient lawmakers I talk about tried to do this, and you can read the little programs in the Hammurabi code, or in the Chinese T'ang code. Of course, this can get intrusive and the society as a whole must decide where this should stop.

You say that there is a difference between programming a deterministic machine and programming social behavior. This is very true, and I touch on this when I talk about deontic concepts. Humans have free will, machines are not supposed to have it. A malfunctioning human must be corrected; a malfunctioning machine can be stopped. This distinction has many consequences. It is a point that must be retained. But, I claim, there are *certain areas* of law where determinism, efficiency and certitude are important. Commerce law may be one, tax law and privacy law may be two others. It's for us to decide. Human mechanisms may be set up to review doubtful outcomes, to see whether the human concept of what should happen is reflected in certain outcomes. The 'necessary distinctions' you talk about must be made, but according to me they should be inside law, they should separate one type of law from another. Technology offers possibilities, but distinctions must be made to decide where and how they should be used. A car can be used to drive to your cottage, or to drive into a crowd.

2. Referee reports on my paper "From e-business to e-laws, e-courts and e-judgments: Can computers be entrusted with legal judgments?" (October 2011).

This paper has undergone several revisions, and was submitted in two versions to a couple of venues (one version was published in CYBERLAWS 2011, but the version on my web site is enhanced with respect to conference version).

Some of the referee comments are quite interesting and worth considering. One important point seems to be the following: Your (my) view is either trivial or too ambitious. On one hand, e-judgments are already a reality (see traffic tickets, tax assessments). If we go beyond that, we run into insurmountable difficulties such as:

- *The inevitable ambiguity of the law, which supports the law's capability to evolve and be reinterpreted in various ways; hence it is very rare that law can be specified precisely.* So it remains to be seen whether e-commerce laws (for example) can ever be specified in precise ways.
- *Difficulties to interpret all sorts of evidence: precise, unambiguous evidence is very rare.* Is it? I think I have some good examples of precise evidence in the paper, but this is to be discussed further.
- *The examples in the paper refer to easy cases.* Quite right. I don't advocate using e-judgments in difficult cases. Some of my legal contacts tell me that much of law is easy drudgery.
- *"Legal scholars have for over a decade recognized that there are reasons to think that the automation of legal reasoning will result in the denial of due process, the overenforcement of unfair laws, and profound injustice."* This seems to be an overreaction, since in my view e-judgments would only be first-degree judgments, immediately appealable. Abuse of concept is always possible but since most technologies have good and bad uses, technologists keep developing them.
- *"It is naive to assume that the proposal will be taken as such. As the traffic speeding example shows,*

digital legal practice may rather evolve stepwise, and not from some grand design. Maybe that is also that the author aims at, but does not make such perspective explicit. It has to be accepted piecemeal and evolve accordingly in phase with political, social and legal developments. The limiting factor here is whether and where current views on technology really enable such a development and practice, and that is where the technical issues come in.” I quite agree. My proposal is futuristic but, I think, worth discussing in principle, read on.

One of the referees mentioned a number of important open research issues. I will report them textually below:

- *The structure of legal reasoning (argumentation, legal assessment methods, legal drafting (consistency checking; simulation, etc.)*
- *The representation of law, the representation of (legal) cases and the mapping of the former to the latter, i.e. the relationship between legal knowledge and the understanding of what happened (or may happen) in the world.*
- *Both issues face the more fundamental and theoretical problems of adequate knowledge representation formalisms, and completeness and consistency in reasoning. The issues are here:*
- *The deontic nature of law and its representation.*
- *Law has to deal with a real world so we cannot make closed world assumptions and therefore completeness cannot be acquired (maybe even more that in more technical domains). AI has learned how to cope with (some of) these problems in practical senses, which is not simply using `heuristics' as the author thinks. Knowledge representation in law is a hot issue, where e.g. ontologies are used to capture the meaning of legal terms in a consistent way (i.e. as specified, intended and, or used in law; not different from law as the author assumes; only in a different way).*
- *These insights reflect back on legal theory but also on legal practice, e.g. in drafting law. Practically, it means that where machines are good at -exhaustive (reasoning) procedures-, where humans easily fail, and vice versa (e.g. in using large amounts of common sense knowledge in particular to understand causal relationships in human actions). Therefore, the issue has shifted towards the question how machine reasoning and human reasoning can be used complementary to achieve a higher degree of accuracy in legal reasoning (e.g. in testing new legislation using simulators).*
- *Laws themselves are justified by political, and/or moral and ethical beliefs. However, legal reasoning finds its justifications in law itself. These justifications (and if necessary explanations) go often by the name of `argumentation structure' which bring events and states identified in the description of cases into relation with statements of law. The (logical) foundations of these argumentations are not undisputed.*

One of the referees also had the following valuable list of references. They are all worth reading, but I highly recommend the essay by Surden, which is the most recent and covers several of the others. BTW the referee mentions that these references are to be considered for the “necessary inequivalence between law and computer code” (which is an obvious point, isn’t it?).

- Samir Chopra & Lawrence White, A Legal Theory for Autonomous Artificial Agents (U. Michigan Press 2011)
- Danielle Keats Citron, "Technological Due Process," Washington University Law Review (2008)
- James Grimmelmann, "Regulation by Software," Yale Law Journal (2005)
- Lawrence Lessig, "The Law of the Horse: What Cyberlaw Might Teach," Harvard Law Review, (1999)
- Lawrence Solum, "The Interpretation/Construction Distinction," Constitutional Commentary (2010)
- Harry Surden, "The Variable Determinacy Thesis," Columbia Science & Technology Law Review (2011)

My conclusion: in spite of many practical and theoretical issues (in fact, because of these) e-judgments are an interesting concept and worth studying, both from the legal and the technological point of view.

3. Referee report on paper “Formal Validation of Compliance of Enterprise Regulations with Privacy Legislation” (July 2012) – Including reflections on deontic logic

The Referee wrote:

My main problem with the paper is that the underlying language is based on first-order logic and it is well know that first-order logic is not appropriate to represent legal reasoning. There are cases where deontic distinctions are necessary (how do you distinguish if something is just factual, permitted, obligatory or prohibited). Also, first-order logic cannot handle reasonig about "violation" but in some areas (for example compliance with business contracts) being able to handle violations and compensation for violations is absolutely required. (see Guido Governatori. Representing Business Contracts in RuleML. International Journal of Cooperative Information Systems 14 (2-3): 181-216, 2005.)

Second the paper seems to ignore work on business process compliance based on deontic logic approaches, claiming there are no efficient approaches (this is not true), see for example:

Guido Governatori. Law, logic and business processes. In Third International Workshop on Requirements Engineering and Law. IEEE, 2010.

Guido Governatori, Antonino Rotolo: An Algorithm for Business Process Compliance. JURIX 2008: 186-191.

Guido Governatori, Antonino Rotolo: A conceptually rich model of business process compliance. APCCM 2010: 3-12 *and on business process compliance in general:*

Guido Governatori and Shazia Sadiq. The journey to business process compliance. In Jorge Cardoso and Wil van der Aalst, editors, Handbook of Research on BPM, IGI Global, 2009.

L.L.'s answer:

The referee starts by saying that "*first-order logic is not appropriate to represent legal reasoning*", and then adds that "*there are cases where deontic distinctions are necessary*'" (I agree with the second statement, but I think that the first one is too generic). He goes on by saying that that first-order logic cannot handle reasoning about violation and compensation! This last part is strange since it seems obvious that one can represent cases of violation and compensation by using simple implications, see the Babylonian codes mentioned at the beginning. Entire legal systems have been built on this basis.

The referee clearly refers to the 'results' of long discussions in the legal logic area, after which some researchers got convinced that deontic logic is necessary for the formalization of legal texts, while others remain unconvinced. See the section above on deontic logic.

Many scientists agree that is useful to see how far one can go with a simpler tool until the more complex tool becomes necessary. Dijkstra often warned against cracking an egg with a sledge-hammer. Einstein warned to keep concepts as simple as possible (but not simpler!) Should we use an electronic microscope when a regular microscope is sufficient or perhaps even better?

It is well-known that much legal reasoning can be appropriately formalized in first-order logic. For many centuries, syllogism (a constrained form of first-order logic) was considered to be sufficient. [Sergot et al. 1986] use Horn clauses (a subset of first order logic) to reason about the British Nationality Act. [Sartor 2009] uses first-order logic in most of his book. Many other papers and books on legal logic do not use deontic logic. Tax laws of many countries have been implemented in programs without using deontic logic, although tax laws use the concepts of violation and compensation.

To extend the referee's comment, it could be said that it is also well-known that deontic logic is not appropriate to represent legal reasoning, because there are so many aspects of legal reasoning that cannot be represented in deontic logic. Since law pretty well covers all aspects of human experience, all tools of logic and mathematics can be used in the study of aspects of different laws. However if I have to determine how many apples I am owed given that Alice owes me 5 and Bob 3 I don't have to use deontic logic, trigonometry or calculus.

So I cannot believe that all legal thinking must be based on deontic logic.

I remain of my opinion that first-order logic is appropriate for much legal reasoning, it is more easily understood (than deontic or higher order logics), it has many good tools, and so we should get as much as we can from it, before we go into more complex formalisms. It is of course clear that deontic logic can be a good tool for certain types of legal reasoning, I respect the work done with deontic logic, and the papers cited by the referee. But higher order logic, and many other forms of logic and mathematics, have even more potential.

I come from a research area (design and verification of real-time systems and network protocols) where a variety of formalisms has been used for many years: extended finite state machines, process algebras, Petri, nets, predicate calculus, temporal logic, model-checking, and others. Each of these formalisms has a school, and members of schools have occasionally implied that only their formalism was the "right" one. More constructively, some researchers have attempted to combine formalisms, by showing the advantages of using, say, Petri Nets with temporal logic. Each formalism is useful for certain types of reasoning. The work continues and often there are interesting new insights, thanks also to these fruitful combinations.

I don't expect this debate to be settled soon: how many generations did it take to settle calculus or gravitation theory? And still progress is being made in these areas.

I would like to continue with some examples. Consider the following norm:

(1) "Parking on Rideau Street beyond one hour is punished at the rate of $10 for every hour in excess".

Suppose that we have a set of norms that is written entirely in this style (essentially, the ECA style discussed earlier). It is possible to reason about this code and derive all useful legal consequences by using only simple logic plus simple arithmetic. Surely, one could rewrite such norms by using deontic operators, e.g.

(2) "It is obligatory to punish parking in Rideau Street beyond one hour at the rate of ...".

But would this help? Or one could state the following fact:

(3) "It is forbidden to park on Rideau Street for more than one hour".

By doing this we have lost some information but, by using deontic logic, we may able to derive other facts such as:

(4) "It is permitted to park for one hour on Rideau Street".

Suppose now that a set of norms is written by using deontic modalities, such as:

(5) "Parking downtown is forbidden beyond one hour".

(5) alone is next to useless because it does not specify the consequences of violations (people will continue parking downtown as long as they want). It will take life by adding ECA norms specifying consequences such as (1) above. To do this, we need a small ontology specifying what are the downtown streets. Supposing that the downtown streets are Rideau, George and York, then (5) is implemented by three ECA rules, e.g.:

(6) "Parking on Rideau Street beyond one hour is punished at the rate of $10 for every hour in excess".

(7) "Parking on George Street beyond one hour is punished at the rate of $5 for every hour in excess".

(8) "Parking on York Street beyond one hour is punished at the rate of $2 for every half hour in excess".

After this is done, simple syllogisms are sufficient to derive the judgments. It can be questioned if (5) is still useful, if not as a guideline.

The interesting case is the one where we have norms of both types. Such are many codes, ancient and modern. In this case, deontic norms can be useful in order to specify properties of sets of ECA norms that can lead to interpretation and completeness checks. For example, in the situation above, suppose that (5), (6), (8) are norms, but (7) is not. There is a problem with the law, something was forgotten, and the legislator or the courts will have to remedy this in some way. This seems to imply that deontic logic belongs to some sort of "meta-legal" level.

By using the concept of ‘contrary to duty obligation’ mentioned earlier, example 1 becomes:

(1’) “It is forbidden to park on Rideau Street for more than one hour. If this is done, the perpetrator is punished at the rate of $10 for every hour in excess”

This rephrasing can be useful in some contexts, useless in others. Depending on what one wants to do …

Exercise: try to rephrase Hammurabi’s code in this fashion.