
DOWNLOAD FULL VERSION HERE

The Second International Conference on Artificial Intelligence and Law proceedings of the conference : June 13-16, 1989, the University of British Columbia Vancouver, BC Canada, University of British Columbia. Faculty of Law, SIGART., 1989, Computers, 258 pages. .


Transforming the law essays on technology, justice, and the legal marketplace, Richard E. Susskind, 2000, Computers, 301 pages. This is the latest book from law and technology guru Richard Susskind, author of the best-selling The Future of Law. It brings together in one volume eleven significant essays ....


Statutory interpretation , Sir Rupert Cross, 1976, , 180 pages. .

Cognition and categorization , Eleanor Rosch, Social Science Research Council (U.S.), 1978, Psychology, 328 pages. .


Law's Empire , R. M. Dworkin, 1986, History, 470 pages. A renowned legal scholar presents a theory of law based on Anglo-American legal principles and practices, juridical interpretations, legal precedence, and a forcefully argued ....


Short Description for Latent Damage Law Designed as part of a package with complimentary computer discs, this book serves as an introduction to the history and use of the Latent Damage System, a computer system aimed at legal advisors and developed by the authors to demonstrate computer potential in the legal profession.
Expert systems are sophisticated computer applications that contain representations of human expertise. They can apply this knowledge to solve problems and offer advice much as a human expert does, through interactive dialogue with non-experts. The systems offer explanations of the lines of reasoning they explore and of the conclusions they reach. The authors of this book have developed the first such expert system, called the Latent Damage System, to be built in the United Kingdom for lawyers and other advisors in professional practice. The book should be used in conjunction with the complimentary computer discs. The text begins with an introduction of expert systems in law and latent damage law. The early investigative stages of the Latent Damage System and its development as a computer system are outlined. The final chapter of the book highlights the power and limitations of the system, itemizes the skills required for a major project in this field, considers legal liability for expert systems and discusses the impact of expert systems in law on legal publishing. A combined book and disc edition is also available as well as a separate disc pack.

Ex Library book with usual stamps and stickers. Good Clean Condition Book. Good condition is defined as: a copy that has been read but remains in clean condition. All of the pages are intact and the cover is intact and the spine may show signs of wear. The book may have minor markings which are not specifically mentioned. Most items will be dispatched the same or the next working day. Bookseller Inventory # mon0000570974

Portions of this page may be (c) 2006 Muze Inc. Some database content may also be provided by Baker & Taylor Inc. Copyright 1995-2006 Muze Inc. For personal non-commercial use only. All rights reserved. Content for books is owned by Baker & Taylor, Inc. or its licensors and is subject to copyright and all other protections provided by applicable law.

Honourable chairman, honourable opponents, honourable professors, ladies and gentlemen, I have been requested by the committee to give a lecture about the following topic: "Is modal logic necessary or to be desired for expert systems in law?" I shall start by clarifying what is meant by the terms "modal logic" and "expert systems".

"Modal logic" is the logic of necessity and possibility (cf. Chellas 1980). As with logic in general, its roots lie in the works of Aristotle. Aristotle reasoned that the necessary must imply the actual, the actual must imply the possible, and hence the necessary must imply the possible, but not vice versa. He also found the impossible to be equivalent to what necessarily is not the case. Modern modal logic started early in this century with the attempts of C.I. Lewis to overcome the paradoxes of material implication. In his attempts to describe a notion of "strict implication" without these paradoxes, he discerned five different logical systems, each of which can be seen as formalizations of various notions of necessity and possibility. A few other modal logics were also discovered at the time, but a fuller and more systematic understanding of the various families of modal logics had to wait until the 1960s.

After the Second World War, there were other philosophical logicians who saw the possibilities afforded by these logics for analysing other concepts as analogous to the concepts of necessity and possibility. This idea that certain notions such as obligation and permission, knowledge and belief, all and some, always and sometimes, etc. can be seen as forms of necessity and possibility was first presented by Leibniz. In this context, it is pertinent to mention Georg Henrik von Wright, who in 1951 published a paper titled "Deontic Logic". In this paper he presents a modal logic for obligation and permission, which, with some minor changes, has become the standard account of the logic of normative reasoning, so called "Standard Deontic Logic". I should also like to mention Stig Kanger, who in 1957 published a small pamphlet called "New Foundations for Ethical Theory" (cf. Kanger 1971). In it he suggested a formal representation of the concept of rights based on Standard Deontic
Logic, a representation which is discussed in some detail in my doctoral thesis (cf. Herrestad 1996).

At the time, Kanger also published a thesis on "Provability in Logic". Some of his ideas about how to make semantic models were seminal for Saul Kripke's development of the now standard account of semantics for modal logics, so called "possible world semantics" (cf. Kripke 1963a, 1963b). The basic idea is that what is necessary must be true in all possible worlds, while what is possible must be true in at least some possible world, even when false in the actual world. By varying the properties of the relation between different possible worlds, models are given for the various systems of modal logic. We are now in the 1960s when the theory of modal logic was developed systematically by logicians like Dana Scott, Erik Lemmon and Christer Segerberg.

Modal logic has been developed further in several different directions. Today it may be seen to encompass a vast family of different logical systems and different semantic models (cf. Bull and Segerberg:84 p. 2). It has been used as a method to clarify numerous different concepts of philosophical importance. To follow the development of modal logic further would take too long. In summary, I want to say that modal logic is used by philosophers because, as Alan Ross Anderson put it; "we like to know with as much exactitude as possible just what we are committing ourselves to, that is, just what consequences our assumptions have" (cf. Anderson 1958 pp. 69-70).

An "expert system" is a computer system based on a model of intelligent behaviour which is able to perform an activity for which a specific competence, or expertise, is normally required (cf. Sartor 1993 p. 25). Typically they have been developed to mimic an expert's diagnostic capabilities in various fields such as medicine or engineering. It is possible to see the process of finding the correct solution to a legal problem as a diagnostic process. Attempts have been made, therefore, to make expert systems in law which can aid the user in solving legal problems.

Expert systems are supposed to adopt the systems architecture of "knowledge-based systems". A knowledge based system has a database of assertions, describing a problem domain in some computer representation, and it has a program - an inference mechanism - which is able to draw inferences from these assertions (cf. Sartor 1993 pp. 20-23). The system is able to indicate the premises and inference steps that have led to a certain conclusion, and, thus, to justify its behaviour. This ability has been seen as particularly appropriate for expert systems in law, as, in law, the justification of the solution tends to be more important than the solution itself.

I understand modal logic as a tool or a technique which has shown itself to be useful for philosophical analysis. I also believe it has been shown to be a useful tool for making informal specifications of various computer programs. It may also be shown soon to be a useful tool for making certain programs. However, I believe it is possible to reject out of hand the case that modal logic is necessary or required for expert systems in law. Even if we could justify the necessity of incorporating in an expert system in law certain modal concepts or certain principles for reasoning with such modal concepts, it would be more difficult to argue that there existed no other way to incorporate them other than by using modal logic. Consider the following analogy: we want to solve the problem of how to fly. If we want to fly, there are certain necessary requirements to be met. To maintain that any specific technique such as making wings is necessary implies that flying could not have been achieved by any other technique. As we know, there are several different techniques which can be used to fulfill these requirements. I believe the same can be said about making expert systems to solve legal problems.

One might deny that modal logic is simply a dispensable tool. One might claim that the logic used and the analysis achieved are inseparable. Of course, if the system developer, in order to incorporate certain philosophical intuitions, has to reconstruct a model isomorphic to that of an existing modal logic, he may be said to have incorporated the logic itself, regardless of what formalization method has been used. I believe, however, that the set of intuitions or insights, which a philosopher seeks to express and clarify by constructing a modal logic, are separable from his mode of expression and analysis. Hence, I believe modal logic is a dispensable tool. If so, it can be dismissed out of hand that modal logic is necessary for expert systems in law. For the rest of this lecture, I shall therefore discuss whether or not modal logic is to be desired for expert systems in
I shall first distinguish three different levels of system development for which it may be desirable to use modal logic. Second, I shall describe how the desire to use modal logic can be seen as related to a desire for making "deep models". Third, I shall refer to the discussion in the field of Artificial Intelligence and Law concerning whether deontic logic is to be desired for expert systems in law. As noted, deontic logic is one application of modal logic in which the modal notions of obligation and permission are regarded as analogies to necessity and possibility. It has been argued that deontic logic is to be desired for expert systems in law in order to represent the specific character of legal reasoning as being predominantly normative reasoning. Fourth, I shall describe how deontic logic has been used in combination with other modal logics to give a precise characterisation of various normative relations, positions of power and responsibility, etc. I shall argue that the desire for making such precise specifications makes the use of modal logic desirable at least for making informal program specifications. Finally, I shall argue that we need to distinguish today between the traditional aims for making legal expert systems and a host of new ambitions for making computer systems that are based on further developments of the technological ideas behind expert systems in law.

The first level of system development is the making of an informal specification of the program. Here "informal" means that there will be no strict interrelation between the specification and the resulting program so as to allow a formal verification of the program as corresponding to the specification. At this level of development, the system developer may find modal logic desirable for much the same reason as philosophers do, because of its usefulness in giving a clear understanding of the central concepts of the domain and their interrelations. Moreover, by making a logic, the system developer may secure that the program specification is consistent. At this level, we still suppose that he is working on paper, with a paper logic, and we do not need to consider any of the problems about computability and efficiency connected to actual program implementation.

Sometimes, but not always, making a formal specification of program-behaviour will be required. This may be seen as a second level of system development. If such a formal specification is made, a strict correspondence is expected between statements in the specification language and internal states in the program built after the specification, a relation which allows a formal proof of the correspondence, a program verification. There need not be a one-to-one correspondence between statements in the specification language and particular states of the program. The specification can be at a higher level of abstraction.

It may be desirable to use a modal logic at this level of development for the same reasons as it was desirable at the previous level. At this level, however, we encounter certain technical questions, such as whether modal logic is more or less adequate than other formal calculi with respect to program verification. I am not competent to assess this. I have just registered that modal logic has also been used at this level of system development. It has been used by, for instance, Yohav Shoham at Stanford University, in specifying certain autonomous agents. According to Krogh, it is also common to use modal logic at this level of specification in the field of cryptographic protocol analysis.

The third level of system development is the actual implementation of a program. Previously it was difficult to conceive of actually including modal logic in the program itself. Now there exists proposals for the development of automatic theorem provers for certain classes of modal logics (cf. Wallen 1987). Through Professor Andrew Jones, I receive reports from the ESPRIT project MEDLAR, in which such automatic theorem provers have been developed. My impression is that, every year, these programs are getting more efficient and are able to handle more systems of modal logic.

Hence, recent developments in automatic theorem proving have made it a practical possibility to include several systems of modal logic in an actual program. I do not believe, though, that there are as yet any commercial products like PROLOG which include modal theorem proving capabilities. There may be certain technical reasons for preferring to use modal logic to make a program rather than some other computable formalism. Possibly modal logic will be able to compete with other
Traditionally, expert systems are built to mimic the capabilities of certain experts. Hence, an analysis is needed of what kind of knowledge and reasoning is used by these experts. Often such knowledge is sought using interviews and other so called "knowledge elicitation techniques". However, system developers have been criticised for having a too simplistic view of what it takes to mimic the reasoning of an expert. It is not enough to interview some arbitrary expert about the heuristics - the rules of thumb - that he uses, and then represent these in a tree of simple IF-THEN rules. It has been argued that the resulting expert system would have a too shallow model of the problem domain. An expert system in law fitting this description is the Latent Damage Advisor, developed in 1988 by Richard Susskind, the first legal doctor on expert systems in law, and Philip Capper, an expert on the Latent Damage Act (cf. Susskind and Capper 1988).

Let me apply a commonly used example in order to illustrate the distinction between shallow and deep models (cf. Bench-Capon 1989 p. 38). Consider wanting to build an expert system to give advice on auto repairs. We decide to elicit the heuristics followed by an experienced garage hand. He might prescribe a useful diagnostic procedure such as: "First, try to ignite the engine. If it doesn't work, check the batteries.", etc. What makes the system shallow is that it contains no information as to the various causal chains which underlie the empirical associations used by the mechanic. If we want a fuller justification for why it works, we would need a model of the inner workings of the engine.

It has been argued that even a shallow model may be acceptable for certain purposes. It may be argued, however, that the principle aim with an expert system in law is to make a system which is able to infer and justify certain conclusions to a question about the correct application of law. It is not to make a system which models the actual heuristics followed by a legal expert or the prejudices which actually may influence the outcome of a court decision. Hence, the emphasis is on making a model of legal knowledge and reasoning that is according to some ideal, rather than on making a psychologically realistic model. One must then make a deep model which takes into account the standards which mark the specific character of legal knowledge and reasoning.

Hence, it may be argued that if we want to make a deep model of legal problem solving, we cannot rely only on the tips of an arbitrary legal expert, and not only on the dogmatic legal theories in the relevant area of law. In addition, we are required to confront theories about the specific character of legal knowledge and reasoning. Such theories are termed "theories of jurisprudence" or "the philosophy of law". Once we have gone this far, however, we may want to go one step further. This is for two reasons.

First, theories of jurisprudence are seldom given the formal rigor which may be desirable if we want to make a program. This is because, whereas a human being can fill out a vague description by using common sense, the program must have every conceptual connection and every principle of reasoning spelled out to it in full detail. Moreover, one would like to see expressed certain principles which allow us to evaluate the computer representation with respect to its consistency. Of course, these are also the purposes for using modal logic when making an analysis of philosophically important concepts.

Second, when we prod the jurisprudential arguments, we often find that in order to justify their claims, they refer to other philosophical theories. Some of these theories are in fact formal theories which try to make every conceptual connection explicit and have every principle of reasoning spelled out in full detail. And some of these formal theories use modal logic. I have already mentioned von Wright and Kanger as examples of philosophers who have used modal logic to make a formal theory about central legal concepts. If we were system developers, we might find it desirable to use these theories instead of theories that are less rigorous.
There are many different aspects of legal knowledge and reasoning for which various attempts at making formal theories exist. Several of these aspects might illustrate modal logic as desirable for expert systems in law. There is one particular aspect of legal knowledge and reasoning that has been the focus of attention in discussions about whether modal logic is desirable. There has been a discussion about whether or not it is desirable to use deontic logic in legal expert systems in order to represent the particular character of legal reasoning that it is normative reasoning. It may be argued that, if deontic logic is desirable, we have at least one good example which may justify that modal logic is to be desired for expert systems in law. Let me, therefore, describe the main arguments of this debate.

One of the puzzles I confronted when working on my Masters thesis was the following: although authorities in the field of making legal expert systems, such as Thorne McCarty and Richard Susskind, agreed that deontic logic was desirable, or even required for inclusion in legal expert systems, no examples could be found of any legal expert systems that used deontic logic in the program (cf. McCarty 1983, Susskind 1987 section 6.4). Not even Susskind's own system, the Latent Damage Advisor, has the slightest trace of deontic logic or normative reasoning whatsoever. When deontic operators were used at all, as in the ESPLEX system, the logical treatment was so rudimentary that it hardly deserves to be termed a deontic logic (cf. Sartor 1993 p. 66).

There were certain practical reasons as to why no deontic logic was used. There were at the time no automatic theorem provers for modal logic, and there were no specifications of how to rewrite a modal logic to other available computational formalisms. With regard to practical system development, apparently the relevant question was not "how do we take the specific normative character of legal reasoning into account?", but rather "what use can we make of the available expert system tools for making expert systems in law?". The practical failure of the Dutch PROLEX project (cf. Oskamp et.al. 1989), and several similar projects, in gaining acceptance with the users, and the relative success of shallow programs like Susskind and Capper's "Latent Damage Advisor" (cf. Bench-Capon 1989 p. 38), apparently showed that an accurate modelling of standards which marked the specific character of legal knowledge and reasoning was less important than other factors for gaining acceptance with the users or for achieving commercial success.

Two exceptions to the overall disuse of deontic logic deserves mention. Thorne McCarty made an informal specification on paper of how to incorporate a deontic logic based on dynamic logic. His aims, however, were strictly theoretical. So were the aims of Marek Sergot when already in 1982, he suggested on paper how obligations and permissions could be included in a non-modal, purely extensional logic program (Sergot 1982). Sergot's program could readily have been implemented as a PROLOG program. In my Masters thesis I tried to show that according to Sergot's specification it would be impossible to give a consistent representation of a conflict between primary obligations and so called contrary-to-duty obligations. I now believe that I may have been wrong and that the non-monotonic properties of a logic program using negation-by-failure make a consistent representation possible. However, the program will have certain counter-intuitive properties. For instance, violated obligations simply vanish. Nothing more can be inferred about them, as the condition for something being obligatory no longer applies. One might argue that in actual life violated obligations do not vanish, but are used in, for instance, arguments concerning guilt or blame.

Sergot is more renowned, however, for arguing that it is perfectly well justified to make expert systems in law without any deontic logic (cf. Sergot 1982 p. 33, Sergot 1988, Jones and Sergot 1993 p. 282). According to Sergot, it is possible to distinguish fragments of the law which are definitional rather than normative in nature. For many practical purposes, only these definitional parts need to be represented in the program. Sergot's research group, The Logic Programming Group at Imperial College London, is well known for its formalization of the British Nationality Act (cf. Sergot et.al. 1986). The system is purely definitional. It can aid in ascertaining whether or not the conditions for becoming a British citizen are fulfilled. The set of rules in the system may be seen as providing a definition of what it means to be a British citizen - how British citizenship is qualified. According to Sergot, it is often irrelevant to draw any further conclusions concerning what normative consequences follow from being entitled in these ways.
Trevor Bench-Capon argues that this is because such expert systems are supposed to be used by low-level adjudicators (cf. Bench-Capon 1989 pp. 43). They are simply supposed to ascertain whether or not certain factual conditions have been met. They are not supposed to weigh up different obligations against each other, nor to consider violating any of their obligations. The core of Bench-Capon's argument is that, in a context where violation is not a possibility which needs consideration, we do not need to distinguish between different modes of necessity; hence we do not need to take into account the specific normative character of legal reasoning. We do not need deontic logic.

Andrew Jones and Marek Sergot describe the essence of normative reasoning as reasoning about the possible discrepancy between the actual and the ideal (cf. Jones and Sergot 1993 p. 279). However, programs are often not made to help us reason about what to do when the actual state is less than ideal. Instead, programs are often made to secure that the actual state will be ideal. The program regiments reality; it forces the actual state of affairs to be in a certain way. Instead of having a program that tells a librarian that the present borrower should not be lent more books, as she has already borrowed the full number she is allowed, the program will just refuse to register further loans. Jones and Sergot agree with Bench-Capon that, if there is no possibility of violation, then there is no need to represent the specific character of normative reasoning. There is no need for deontic logic.

However, Jones and Sergot suggest that there are many circumstances in which regimentation cannot be guaranteed or where regimentation will be undesirable. The maker of a system specification may desire to use deontic logic in the informal program specification in order to give a consistent specification of both a set of primary requirements that the system ought to meet and a set of requirements contingent on the failure of the program to satisfy the primary requirements. Such failures are frequent, and it is rational to make certain contingency plans.

Another set of circumstances are created by malfunctioning of hardware or software components. It is desirable that programs should be fault-tolerant, and inclusion of deontic logic in the program might help in making the program reason consistently about its primary and its fault-contingent requirements. Finally, the above mentioned librarian may not want a strict regimentation of her practice of lending out books. Consider that the librarian finds out that the woman has good reasons for borrowing more than her allowance. The librarian might then want to violate the rule rather than be forced by "that damned computer" not to lend another book to the woman. In Jones and Sergot's terminology, she would want the computer to impose "soft" violable constraints, rather than "hard" inviolable constraints, on the practice of lending out books. Again it may be desirable to include a deontic logic in the program in order to make the program reason consistently about her primary and contrary-to-duty obligations. Jones and Sergot accept that for certain purposes nothing more than a very rudimentary deontic logic is required. But we are very quickly confronted with the unresolved problem of defining appropriate detachment principles for, respectively, factual and deontic detachment. Factual detachment is the detachment of an obligation when certain factual conditions have been met. Deontic detachment is the making of deontic inferences.

As was already shown by Roderick Chisholm in 1963, Standard Deontic Logic has a combination of factual and deontic detachment principles which makes it impossible to represent situations where the consequences of primary and contrary-to-duty obligations are in conflict (cf. Chisholm 1963). This has been termed "Chisholm's paradox". Consider that we want the program to tell the librarian that she ought to insert a green note warning of a four week loan limit on normal loans, and that when the allowance is superseded she ought not to insert the green note, but instead she ought to insert a red note warning of a two week loan limit on books superseding the allowance. This set of obligations cannot be given a representation in Standard Deontic Logic that is both intuitively plausible and which yields consistent inferences, when the situation arises that the allowance is superseded. Though there exist several deontic logics which can give a consistent representation of this situation, none have gained wide acceptance for doing so in a way which is intuitively acceptable. There is no wide agreement on the principles for factual and deontic detachment. That this is the case thirty-three years after Chisholm published his paradox has been termed "the
Scandal of Deontic Logic.

I believe this scandal attests to two things. First, there are no more than a handful of people in the world who from time to time try to solve Chisholm’s paradox. Second, there is a great deal of variation between different people when it comes to their intuitions about normative reasoning. Just as with ethical theories, it appears difficult to find good criteria for reaching a wide consensus. Hence, if our system developer believes that he ought to make a consistent representation of the above situation, he will have to choose the deontic logic which gives a consistent representation with minimal offence to his own intuitions concerning normative reasoning.

In my doctoral thesis, I have followed yet another suggestion by Jones and Sergot, the suggestion that even Standard Deontic Logic is to be desired for expert systems in law when it comes to specifying various normative positions (cf. Jones and Sergot 1992, Herrestad 1996). To be more exact, deontic logic has been used in combination with other modal logics to give a precise characterisation of various normative relations, and various positions of authority, power and responsibility, etc. I shall argue that the desire for making such precise specifications makes the use of modal logic desirable at least for making informal program specifications.

The use of a formal logic allows exact specification of logical implication and consistency. If we have a statement that a certain obligation exists, a deontic logic can be used to specify what is entailed by such a statement. Furthermore, we can give a complete specification of what other statements concerning obligations and permissions can be maintained, in conjunction with this obligation, without being inconsistent. We can even give an exhaustive list of such complete unambiguous statements. For instance, according to Standard Deontic Logic, the list of completely specified statements contains only three statements: it can be obligatory and permitted that the state of affairs be the case; it can be obligatory and permitted that the state of affairs not be the case; or it can be permitted that the state of affairs be the case and permitted that the state of affairs not be the case.

Kanger introduced a modal action operator in the scope of the deontic operator (cf. Kanger 1971, Kanger 1972). This increases the expressive richness. We no longer have just the two possibilities that a state of affairs is the case or that its negation is the case; we can now express that someone sees to it that a certain state of affairs is the case, that he sees to it that the state of affairs is not the case, and thirdly that whether the state of affairs is the case or not, is not a result of his actions. As shown by Lars Lindahl, we now get an exhaustive list of seven, rather than three, consistent conjunctive statements concerning what is obligatory or permitted for one person to do (cf. Lindahl 1977 p. 92). If we want to make conjunctions concerning what is obligatory or permitted for two persons to do, we get 35 such statements, and, if we consider their joint actions as well, we get 127 statements (cf. Lindahl 1977 p. 130 and pp. 164-165). All these lists presume that there is only one single state of affairs under consideration.

What use can there be in making such lists? In the case of regulations concerning the actions of two persons, it makes possible an unambiguous statement of their relative positions concerning what actions they are required or allowed to carry out. What use can there be in making such unambiguous regulations? Layman Allen has argued in several papers that this is a valuable tool for legal drafters to avoid unwanted ambiguities (cf. Allen 1965, 1974, Allen and Saxon 1995). Together with a Dutch colleague, Cees Groendijk, I have investigated the possibility of developing this into a simple consistency check mechanism, a logic-checker to use in conjunction with a spell-checker (cf. Herrestad and Groendijk 1993). However, it requires a willingness on the part of the drafter to do a substantial amount of translation into a formal language. Moreover, in many cases I doubt whether the drafter would want to make his proposals more unambiguous. If they are proposals for controversial legislation where a consensus has to be reached, he may want just the opposite. One reported attempt, in the American state of Tennessee, to apply such techniques manually on a routine basis when making legal drafts did not meet with any success (Gray 1985).

Jones and Sergot suggest that the possibility of making such unambiguous regulations could be a valuable part of an institutional design process (cf. Jones and Sergot 1992). In particular, where the institutional design involves certain computer systems which require various measures of computer
security such as certain access control mechanisms. They also suggest that one could generate exhaustive lists of maximally consistent conjunctions involving combinations of several other modal operators. We could then analyse not only their relative positions concerning what actions they are required or allowed to do, but also what relations of physical influence and control exist. All in all, these unambiguous statements may be used to give a detailed picture of the normative and factual relations of responsibility and power present in an existing institution or required in a new institutional set-up. Finally, Jones and Sergot suggest making automatic tools for generating such exhaustive lists of unambiguous statements, and for guiding the user to the most relevant parts of these lists, as the numbers of statements may be substantial.

Usually we think of a legal expert system as a decision support system to be used in the process of solving traditional legal problems. However, it is not difficult to see that an expert system could give aid to a person drafting new law rather than give aid to a person applying existing law. Furthermore, we can think of an expert system as a tool for specifying computer systems which are supposed to operate automatically, but which are required to reflect or conform to certain legal regulations.

As argued by Dag Wiese Schartum, system development may often involve a legal decision process (cf. Schartum 1993 pp. 244-247). It is a legal decision process because the system developer has to interpret the law in order to make the program reflect the law, and he will often have to supplement the law as the law is seldom specified at a sufficient level of detail to make the program work. The kind of automatic tools described by Jones and Sergot may be seen as a kind of expert system that can aid in clarifying the possible interpretation of legal regulations as a basis for making such supplements. Schartum has argued that making such supplements is not a legislative activity; it is still an interpretation of the law. But some of these supplements may be so substantial that they ought to be enacted as statutes or regulations in the usual way, rather than just being buried in the program code.

Finally, more and more automatic mechanisms are made whose operations are required to reflect or conform to certain legal regulations. One of the aims of the Section for Information Technology and Administrative Systems, at the University of Oslo, is to study such mechanisms in the programs of public and private administration. New types of mechanisms are still at the stage of informal specification. These are mechanisms for turning the Internet into a huge information market, mechanisms for automatic contracting, and mechanisms for intellectual property protection.

A legal expert system was earlier perceived as a program advising a human on how to solve a legal problem. Today we can perceive the program becoming an autonomous agent, authorised to go on a shopping spree on our behalf, which itself will need expert knowledge to solve the problems which may occur. Legal expertise will be needed, not to advise us on making contracts, but in order that the program can go about its own activities without incurring unwanted liabilities on our behalf (cf. Krogh 1995).

Already the Internet is the habitat of several species of autonomous agents, from the postmasters passing our mail to the web crawlers weaving a web of links to the servers where the information we search for is found. It is already so crowded that there is talk of requiring the agents to comply with certain social standards, and of sanctioning malevolent crawlers (cf. Eichmann 1994). They can be malevolent by, for instance, soaking up computer resources so as to make delays or block other more urgent uses. The specification of social standards can come to mean that these agents have a set of internal norms of conduct where they assess independently whether they serve their owners best by violating the norms even when there is a risk of being met with sanctions. The reason for making violations possible would be that the cost of rigorous control is too high. The difference between these standards being social standards or legal standards may simply depend on whether or not the legislators have been able to catch up with the development.

In conclusion, modal logic is to be desired for expert systems in law in order to make computer programs more flexible and in order to make programs that reflect or conform to legal regulations. In the end, this means that we, the end users, will get more utility out of our computers, that we will have less reasons to feel that our rights have been violated, and that there will be less cause for
legal battles to be fought. Hence, the new uses of modal logic for expert systems in law may hopefully prevent the need for legal expert systems to aid in the kind of legal problem-solving, for which such systems may originally have been intended.