*Databases and Information Systems VIII*
*H.-M. Haav et al. (Eds.)*


*doi:10.3233/978-1-61499-458-9-275*

# Compliance and Software Transparency for the Design of Legal Machines

Vytautas ČYRAS[a,1] and Friedrich LACHMAYER[b]
[a] *Vilnius University, Vilnius, Lithuania*
[b] *University of Innsbruck, Vienna, Austria*

**Abstract.** This paper attempts to define the software compliance and transparency problem. This constitutes a high-level holistic view. Several visualizations of the problem are presented. The context is the changeover from a text culture to a machine culture in law. We note that equal access to e-procedures does not guarantee justice. The transparency of the law leads to the transparency of software and hence challenges legal informatics. We formulate two requirements for legal machines: 1) the software architecture must be accessible; and 2) the software must provide legal protection. A need therefore arises for the legal requirements to flow down to lower-level specifications. Further we explain the notion of subsumption – a legal qualification of facts in the setting of a norm. Finally we discuss the definition of the compliance problem.

**Keywords.** Informational process, legal meaning visualization, legal requirements, legal protection, software process

## Introduction

This paper addresses regulatory compliance as an ideal, and presents different abstract formulations of the problem. We start with a picture in Figure 1. Legal norms appear in the Ought world, whereas the development of a machine appears in the Is world. Is and Ought are distinguished in the theory of law: see Hans Kelsen [1].

Machines contribute to law enforcement, and their software implements legal norms that are explicitly taken into account in the requirements engineering phase. Here we call such norms *explicit norms*. The problem is that there are also norms that are not considered in the software lifecycle. We call these norms *implicit norms*. These norms can be violated, and therefore they are important. There is no strict boundary between explicit and implicit norms, and a grey area exists due to the interconnectedness of norms in the whole legal system.

This paper extends our earlier studies on legal machines [2], transparency [3], and compliance [4]. A legal machine can be defined as a machine in a system whose actions have legal importance and legal consequences.

There are simple legal machines, such as traffic lights, barriers and vending machines, and complex ones, such as the electronic forms that are used for taxes and finance. An example of the latter is FinanzOnline, which provides a one-click link to the Austrian tax administration; see https://finanzonline.bmf.gv.at/. Legal machines

[1] Corresponding Author: Faculty of Mathematics and Informatics, Naugarduko 24, 03225 Vilnius, Lithuania; E-mail: vytautas.cyras@mif.vu.lt.

translate raw facts into institutional facts, i.e., into facts that have legal importance. The raw facts come from Is, whereas the institutional facts come from Ought. For example, a fraud is committed when someone puts fake coins into a vending machine, whereas a child may put outdated coins into her piggybank.

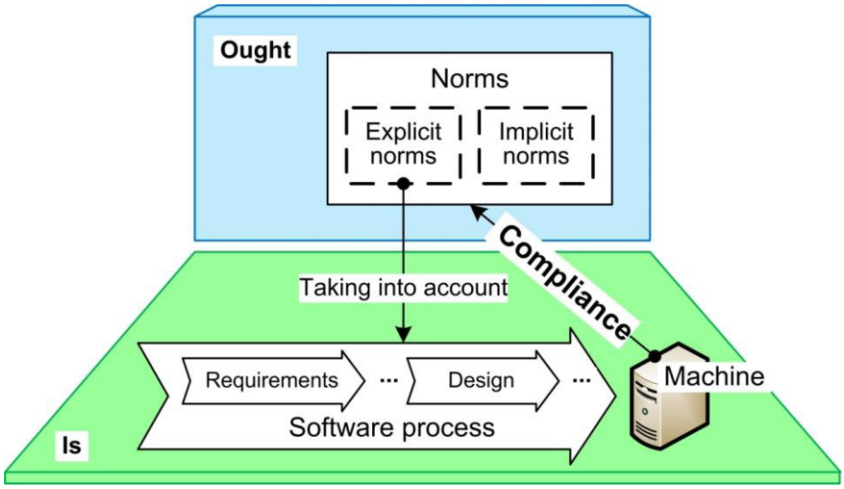


**Figure 1.** The first visualization of the compliance problem. The legal machine appears on the Is stage and has to be compliant with the legal norms that appear on the Ought stage

**Compliance in the design phase.** It is important to detect noncompliance in the early stages of the software life cycle. It should be noted that legal knowledge need not lean heavily on the sources of law that are traditionally considered in compliance frameworks. Suppose we have a business process. Each link in the process can be considered to be compliant (at least at first glance) but the whole process can be noncompliant. Such noncompliant business processes are the concern of [5]. For example, a sales chain in which the initial seller and the final buyer are identical gives rise to suspicions of tax evasion. Generic legal tasks can be related to different kinds of legal knowledge, such as legal rules, design patterns, social structures and evidence-based expectations. The new idea of Boer and van Engers is the methodology that stresses how noncompliance stories are elicited. Noncompliance in design is a concern in their methodology.[2]

## 1. Another Visualization of the Compliance Problem

Other visualizations of the compliance problem can be proposed. Figure 2 illustrates the bridge metaphor that connects computer science (informatics) and law. Legal informatics attempts to build a bridge. The transformation *legal text* → *program* is

[2] "We learned that specifications insufficiently leverage experience about the expected behavior of agents in the domain. This weakness is most evident in the absence of explicit theory construction about *noncompliance* in design, even though experience about critical incidents in the past, and the ability to predict future incidents, is usually available in the organization and certainly plays a leading, but largely implicit, role in design." [5] p. 171.

worthy of a separate study, which is outside the scope of this paper. We believe that the path connecting legal text to programming requires intermediate steps. We thus identify this approach as Multiphase Transformation [6].

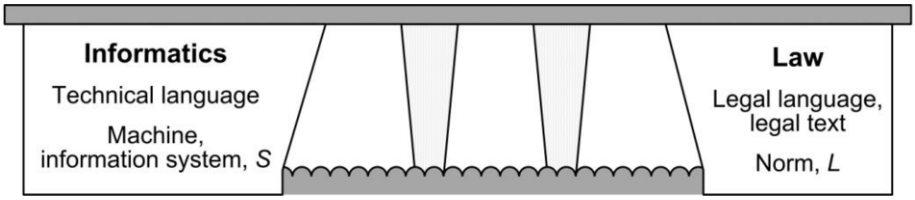


**Figure 2.** Building a multi-arch bridge between computer science and law

A person who is not legally qualified (e.g., a software engineer) and a jurist may have different views on the violation of a legal rule. The reason for this is that a layperson can barely understand the whole interconnectedness of legal norms. Therefore, determining the compliance of software with the law is a complex problem. Note that an information system can cause harm, as can any misused artifact. For example, a computer generated message can cause a heart attack in the same way as a pencil can serve as a murder tool.

Consider the question "Is this software compliant with the law?". The answer need not be "yes" or "no", but can lie on an evaluation spectrum that ranges from optimistic to pessimistic. We find this question similar to the question, "Can machines think?", which was asked by Alan Turing [7]. Turing begins with the definition and the meaning of the terms "machine" and "think". Analogously, we look at the terms "compliant" and "the law".

In practice, an auditor can certify that software is compliant with a specific rule, a law, or an audit guide. A typical rule is "A specific artifact *X* shall be present in an organization". However, the answer "yes" can scarcely be given to a question about compliance with the whole body of law. The reason is that an expert can understand the legal meaning or, in other words, the Ought of this rule, this law or this audit guide respectively. However, an understanding of the legal meaning of the whole body of law can scarcely be achieved. Hard cases in the law demonstrate that even judges may have separate opinions. Therefore a more experienced auditor can foresee a greater risk that the software does not comply with the law.

## 2. On the Transparency of Legal Machines

This section formulates the problem that is termed "legal machine transparency" [3]. Consider a) a *general norm* that is formulated in a legal text such as a law or a by-law; and b) an *individual norm* that is formulated in a court judgment. The following means of legal protection serve the different parties in the context of a text culture:

1. *Transparency of law*. Legal texts are made accessible to citizens. An example is the Austrian e-government application (portal) HELP,[3] which states the law

[3] HELP.gv.at – a government agency help site on the Internet, which offers information necessary for those living and working in Austria.

that is applicable in various situations and supports ex ante analysis ("beforehand", "before the event"); and

2. *Ex post legal protection*. An example of this is RIS,[4] which publishes cases and supports ex post analysis ("afterwards", "after the event").

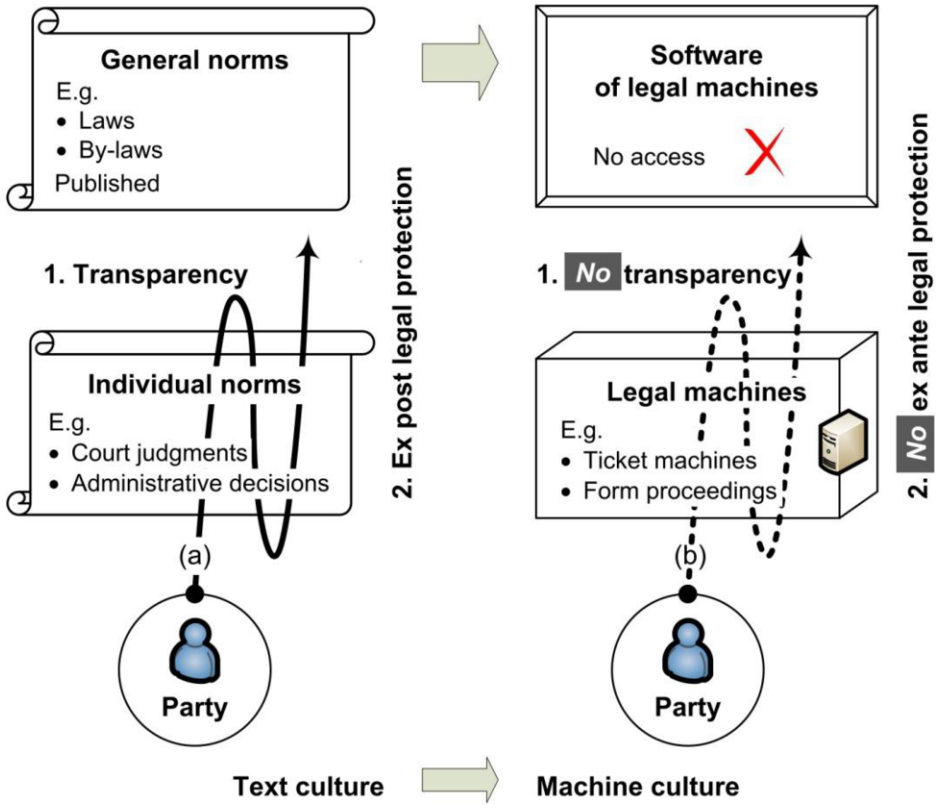


**Figure 3.** Technical implementation of a changeover from a legal text to software. The problem is to ensure transparency and legal protection at the same level in both (a) and (b)

These two types of information were not available years ago and have been improved over the course of time; they now come as standard. However, this standard was not set at the beginning of the machine culture. We see the problem of a lack of transparency and of ex ante analysis support in the machine culture, as the machine culture is still young. A party cannot, for example, wait three years to resolve a conflict with a machine (Figure 3).

Therefore we have a proposal that is formulated as two requirements for software:

*Requirement 1*: The architecture of the software for a legal machine must be made accessible to the user.

*Requirement 2*: The software for a legal machine must provide trained, effective and rapid legal protection.

[4] The Legal Information System of the Republic of Austria; http://www.ris.bka.gv.at/.

We aim to ensure that there is the same standard of transparency and legal protection in the machine culture as there is in the text culture.

As examples, consider the following imaginary situations:

*Example 1*: The law provides for ten variations/possibilities but the program only contains nine. Let us suppose this applies to expenditure categories in tax law. Suppose you want to declare expenditure in the tenth category in your tax declaration, but the program does not allow this option. What do you do?

*Example 2*: A vending machine is designed not to give money back. This causes a problem for customers who expect to receive change.

The latter situation can be observed in car parks equipped with unsophisticated ticket machines that give a very limited amount of money back.

The different variations/possibilities like the ten in Example 1 may be listed in different articles or laws. Ordinary people who are not legally trained or qualified can hardly understand a complete list of variations. A legal norm is a result of the interpretation of legal texts, and need not be extracted from one article; its formulation may extend to several text passages.

## 3. Notes on Compliance

Suppose that a driver violates the speed limit in order to avoid greater harm, and a radar gun records the violation. The speed limit rule is violated and a sanction applies. However, a more general question can be asked: "Has the driver violated the law?". To answer this you have to understand what the law is. The answer is not a truth value, "yes" or "no", but a complex judgment that depends on the procedure and the parties.

Rules typically have exceptions, exceptions to the exceptions and so on. The speed limit rule has exceptions such as, "This rule can be violated in order to avoid greater harm." An expert in the law, and not a layperson, can understand exceptions in their entirety and the whole interconnectedness of conflicting rules.

Again, we find that the question "Is a program compliant with the law?" is similar to Turing's, "Can machines think?". The term "the law" denotes the whole system of rules that everyone in a country or society must obey. Understanding the system may lead to understanding law as a phenomenon. The complexity of our question could be reduced by substituting "the law" with "a rule": "Is a legal machine compliant with a rule?". However the speed violation example shows that the reduction is not essential and does not reduce the problem to a mechanical decision "yes – complies" or "no – does not comply." Institutional thinking and the concept of a legal institution distinguish legal informatics from the formal theory of reasoning that is typically used in computer science.

**Legal norms ≠ legal text.** The notion of a legal norm is not as simple as it may appear at first impression. Legal texts are made not of norms but of structural arrangement units such as parts, sections, paragraphs, etc. Moreover, a legal norm is not a primary elementary notion of the law. Legal documents, as a form of legal information, do not know the notion of norm. Legal theory holds that a legal norm is a mental product. A norm is obtained by interpreting a legal text. A paragraph of a document can contain several norms of behavior, or a norm can continue through several paragraphs, part here and part there. In addition to legal norms as mental constructs, legal institutions and situations can be extracted interpretatively from legal

texts. Institutional thinking is also a mental construct and is obtained through interpretation. Moreover, the texts of laws constitute only a part of a whole legal system. A concluding remark is that treating a norm as a text can lead to formalistic consequences.

**Representing legal norms as rules.** When formalizing legal norms, we find it useful here to follow [8] pp. 291–294, where references to related approaches are provided. We recall the early works on artificial intelligence and law in the 1980s and on modeling legal reasoning, and the good old days of Prolog [9].

Typically, a legal norm determines a "legal consequence" (*LC*), given one or more "state of affairs" (*SF*) that fall within the scope of the norm. Schematically, this can be expressed as a logical "rule": $SF \rightarrow LC$. This is to be read as: "When a state of affairs (*SF*) is given, then the legal consequence (*LC*) applies" or "If *condition* then *normative position*." A more general format is: $SF_1,\cdots SF_n \rightarrow LC_1,\cdots LC_m$. This is generally used in computer science and specifically in ontology-related issues, in the context of semi-automated legal reasoning and in engineering software that is compliant with the law.

There is a distinction between *technical rules* and *legal rules*. You cannot violate a technical rule unless you destroy the legal machine (consider a bank machine or a barrier). However, you can violate a legal rule.

In this respect the question is the extent to which business rules can implement norms. Therefore the research on implementing rules in information systems, business rules frameworks and ontologies is a talking point; see, for example, [8, 10].

## 4. The Context of Justice in the Paradigm Change to Electronic Procedures

Hans Kelsen, one of the most important legal scholars of the twentieth century, pursued several objectives in his *Pure Theory of Law* (PTL) [1]:

- *First, a paradigm shift in legal theory*. PTL became a new jurisprudential standard and, in this respect, shaped the twentieth century.
- *Second, the overcoming of the natural law*. PTL has outlived the natural law.
- *Third, to stop arguments that use the concept of justice*. However, Kelsen did not succeed with this objective.

The discussion today is more about injustice.

### *4.1. The Principle of Equality before the Law*

One of the fundamental principles of justice is equality. The application of justice is less of a normative problem, because it is preceded by a cognitive problem concerning matters such as interpretation. A preliminary issue is that what is found has to be interpreted. However, this is not a normative issue: the subsumption is interpretative, i.e., cognitive at its core. Depending on the legal concepts that are applied in the interpretation, the subsumption, which follows the interpretation, will be different.

The context of justice does not only affect politics (large and small), but also affects legal machines. The upcoming world of electronic administrative and judicial proceedings brings a new context for the application of justice. With electronic proceedings, the context is all about everyday justice for those citizens who are affected, and about the procedurally mediated law that they enjoy. But even with IT-based everyday justice, the cognitive problem is upstream of the normative problem.

### *4.2. Subsumption*

The key to the application of the law and thus to access to justice is subsumption. The special feature of matters carried out with electronic forms (which we will refer to as e-proceedings or e-procedure) is that subsumption is accomplished by the parties and not by the court or the administrative authority. In traditional proceedings, the parties submit their allegations, these are confirmed or rejected in the course of the investigation, and finally the state of affairs is established by the authority. The state of affairs serves as the basis for the subsumption. Thus, there are a number of regulatory steps between the parties' assertions and the subsumption.

In e-procedures this is different. One party fills in the input fields. How the entries in these fields need to be made, however, is not commonly known but requires a knowledge of jargon. Thus, the performance of subsumption is delegated to the parties. To make entries, the parties have to use their knowledge of the legal terms that are defined by the sources of law in a complicated way.

Back-office programs for e-procedures are very quick and accurate at drawing legal conclusions from input data. However, the content of the conclusion depends to a large extent on the starting material that was completed in the parties' front offices. A party who is not skillful in law may be overwhelmed by this. This is an opportunity for intermediaries, allowing them to continue to position themselves professionally. The complexity of the law is not reduced by e-proceedings, but, as it did in the Middle Ages, continues to require intermediaries who now must be expert in information technology.

All parties are equal in e-procedures, and in this respect e-procedures seem to be fair. However, in the actual performance of an e-procedure there may be a big difference between communicating with a human operator and communicating with a machine. A human operator can offer flexibility, and this cannot be offered by a machine. As an example, suppose a traveler sees a train already approaching and tries desperately to buy an appropriate ticket from the ticket machine. A human being would assist better than the machine.

An example of noncompliant software design is provided in [8]. This example demonstrates legal reasoning that leads to a violation of a data privacy law. The running example in [8] is a situation in which a user's consent, given by clicking "yes" on his mobile phone, is not treated as an effective consent in the legal sense.

**Equality ≠ justice**. Equality is far from able to guarantee justice. An example is a transactional tax. This is the same for everybody, but has a greater impact on the income of the poor than on the income of the rich. Therefore, this is perceived as unjust.

Knowledge of the law is similar. Today the law is conveyed electronically in the same way for all people. However, there are big differences in individuals' ability to deal with this knowledge, in addition to the differences in technical skills and aptitudes in dealing with the new media.

**Ex ante legal protection**. An open issue is certainly the lack of ex ante legal protection against incorrect electronic forms in e-procedures. Consider again Example 1, in which the law provides for ten cases but only nine of these cases are mentioned in the e-forms. The question arises of how someone should behave if he wishes to enter information in the tenth case, when he cannot do this electronically. Any ex post legal protection comes too late after the first instance of this and of the expiration of the deadline; and it is dysfunctional, since it only involves the input stage.

The paradigm shift to e-procedures provides small- and medium-range domains with a new standardized legal culture, which could not be achieved before. This

paradigm shift provides equality and is "fair," apart from the concern over different starting conditions. However, as in the past, injustice will be a massive problem in the areas in which it is so great that it cannot be consciously perceived. For nothing is as invisible as the things that may be overlooked. Here the paradigm shift to e-procedures will not change injustice.

## 5. Explaining the Notion of Subsumption

Subsumption refers to the application of the law, or more precisely, the application of a norm to a fact, thus concluding the legal qualification. An English dictionary defines subsumption as: 1. that which is subsumed, as the minor clause or premise of a syllogism; or 2. the incorporation of something into a more general category. Subsumption is central in making a legal decision.[5]

Legal qualification, which results in the subsumption procedure within the legal domain, is central for ontologies in law. For example, an answer if a killing (world knowledge) is murder, legal sanction in the form of an execution or allowed act in an international armed conflict is given only by the legal qualification of the act [11], p. 9. We model this statement in Figure 4.

Subsumption is of key importance in judging a violation. In order to judge whether there is a violation of the law, a factual situation (which is from the Is world) has to be subsumed under a legal norm (which is from the Ought world). Note that the factual situation can be subsumed under different norms. Therefore, conflicting norms and norm exceptions have to be taken into account. Only a jurist is capable of doing this. Here a balance is more important than an absolute rule.

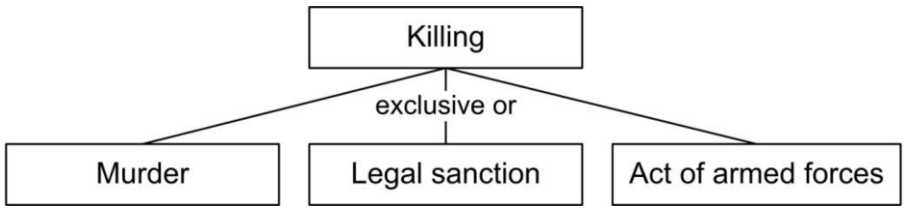


**Figure 4.** Different legal qualifications of killing

We divide the concept of subsumption into two types: *terminological subsumption* and *normative subsumption*. These are explained further below.

The facts of a case are transformed into legal terms. Suppose that an action, $\mathsf{a}$, is treated as a theft, $A$, not a burglary. This corresponds to the first kind of legal subsumption, called terminological subsumption. We write $\mathsf{a} == A$ (Figure 5) or *instance-of*($\mathsf{a}$,$A$). A pool of legal terms is used for the terminological subsumption. We use a visualization pattern that is composed of a vertical stage (the Ought realm that comprises institutional facts) and a horizontal one (the Is realm that comprises raw facts).

[5] http://de.wikipedia.org/wiki/Subsumtion_(Recht)

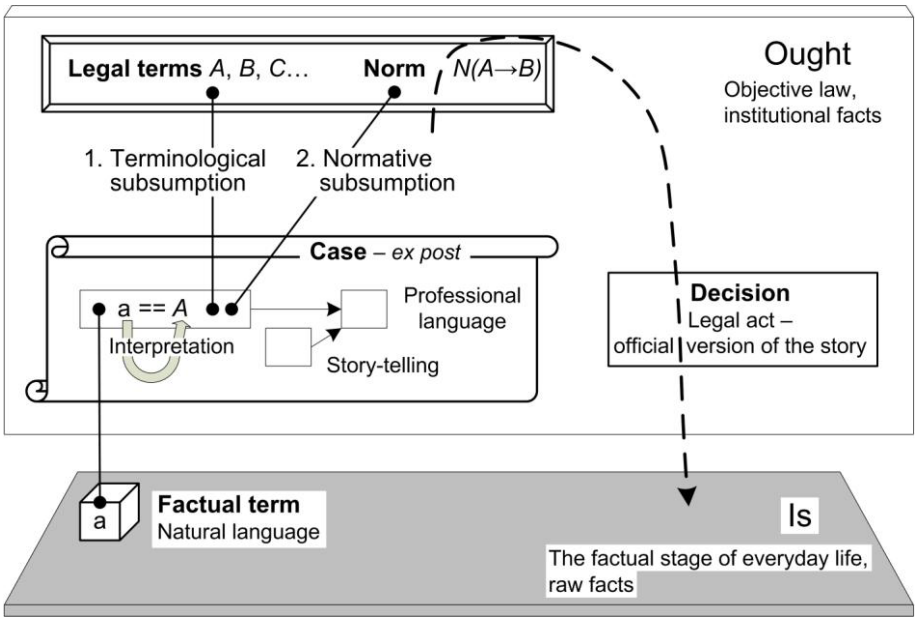


**Figure 5.** Subsumption: the facts are assigned the legal qualification in accordance with a norm

The second step is *normative subsumption*. Here the norm $Norm(\forall x\, A(x) \rightarrow B(x))$ is applied to subsume $B$. The first step, terminological subsumption, corresponds to the unification. It is linked with the minor premise. The second step, normative subsumption, corresponds to the major premise $\forall x\, A(x) \rightarrow B(x)$. The conceptualization above reflects inference with a syllogism (cf. also the *modus ponens* rule):

| | | |
|---|---|---|
| Minor premise: Socrates is a human. | | $human(Socrates)$ |
| Major premise: Humans are mortal. | | $\forall x\, human(x) \rightarrow mortal(x)$ |
| Conclusion: | Therefore, Socrates is mortal. | $mortal(Socrates)$ |

**Modeling terminological subsumption**. To model the subsumption procedure, conceptual modeling formalisms can be applied. Suppose we have the fact that *mydoor* is open and the norm $N$ "The doors ought to be closed." The norm can be formalized with the following rule: if $x$ is an instance of *Door*, then $x$ ought to be closed. Formally, $\forall x\, x \in Door \Rightarrow O\, closed(x)$, where $O$ is the deontic operator and *closed* a predicate. A situation (i.e., a fact) with the instance *mydoor* is from the Is world. The fact is interpreted according to a norm from the Ought world which contains the door concept *Door*. Then *mydoor* is matched with *Door*, formally *match*(*mydoor*, *Door*). This can be simplified and expressed with a truth statement *instance-of*(*mydoor*, *Door*) or $mydoor \in Door$. This truth statement is from the Is world. A duty which is conferred on me, to close *mydoor*, is from Ought. In the Is world I can decide to leave *mydoor* open, thus violating the norm. A sanction can be conferred on me.

Finally, the subsumption procedure is illustrated in Figure 6. A fact a is qualified as a legal term $A$. The norm $A \rightarrow B$ is applied. The conclusion is $B$(a). The law can comprise a more specific norm which involves $A$, e.g., $A$, $C \rightarrow D$.

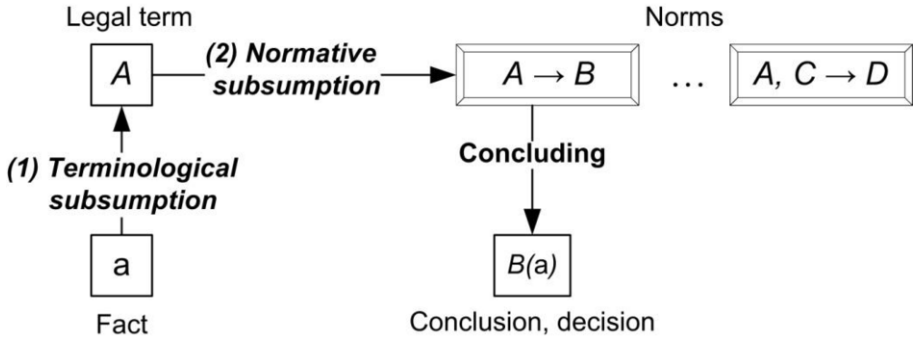


**Figure 6.** The subsumption procedure

Next it is worth noting that the application of the law has to avoid formalism (mechanistic approaches). This is stressed in legal theory. The idea of constructing a subsumption machine (Ger. *Subsumtionsautomat*, "mechanistic judge") is rejected; see [12] pp. 212, 292 et seq.

To sum up, we aim to contribute to the problem of the relation between fact and circumstance (Ger. *Tatsache und Sachverhalt*). Our approach can be treated as formalization through symbolization.

## 6. Transparency in Software Engineering

Requirements 1 and 2 in the introduction are formulated at a very high level. The next question is how they can be implemented in a legal machine. In principle, the architecture of the legal machine can be made available to the user. Transparency of computer programs is a concern of both software engineering and human–computer interaction.[6] Here transparency is combined with the principles of information hiding and separation of concerns. On different abstraction levels, a software element is treated as a black box that can be viewed in terms of its input, its output and its transfer characteristics without any knowledge of its internal workings.

There is no automatic way to achieve transparency in software engineering. Different users may be interested in different architectural elements. Not every design solution can be made accessible to each user. This is for reasons of security and complexity, to mention just a few. Software requirements, including transparency requirements, are formulated in the early phases of the software development life cycle. Legal requirements are at a high level and are also a concern of requirements engineering. A flowdown of the requirements is needed to develop lower-level specifications, and these are formulated in the system specification or a similar document. There is a need to flow down the high-level Requirements 1 and 2 in the introduction above, too. After this the software is designed and implemented. Thus, the resulting software can be made compliant with the initial legal requirements.

[6] Transparency means that a distributed system hides its distributed nature from its users, appearing and functioning as a normal centralized system. In software engineering, it is also considered good practice to use different abstraction layers. There are many types of transparency; see http://en.wikipedia.org/wiki/Transparency_(human%E2%80%93computer_interaction).

However, in practice the story above is rarely so simple. There may be a wide gap between the legal requirements and the technical specifications [13]. Failure to understand the law is one of the reasons why the program may be noncompliant [14], pp. 59–61. The meaning of the law (Ger. *Sinn*) – the Ought realm – can hardly be understood from a single legal text. Therefore it is difficult for a beginner to understand the spirit of law while reading a statute in isolation. For this reason, only well-defined compliance problems can be implemented by ticking boxes in an audit document.

**On implementing Requirement 2 – Legal Protection.** Legal protection, which is mentioned in Requirement 2, is even more difficult to implement than the architectural transparency in Requirement 1. The efforts of software engineers are not enough to make sure that Requirement 2 flows down. Organizational means and invention may be needed. The following paragraph sets out an imaginary situation.

Consider an automatic barrier to a pay parking lot. Suppose that the barrier does not lift up to let you out. Suppose that the cause is purely technical, such as a malfunction of the barrier's motor. What action can you take and what means of legal protection do you have? Some parking lots provide a 24 hour/7 day voice connection to a human operator. You can therefore explain the matter to the operator and he can arrange for the barrier to be lifted manually. However this is not always the case.

As far as legal protection is concerned, the architect of the parking lot should, under Requirement 2, supply instructions for how the user should behave in an emergency. To sum up: although legal machines are typically treated as black boxes, transparency can be increased with a transparent flowdown of legal requirements.

## 7. Formulating the Compliance Problem

We now attempt to formulate the problem of compliance with the law. Klaus Julisch [15] suggests that academia should undergo a paradigm shift away from "selling" security, because organizations seek to "buy" compliance:

> [A]s long as careers are terminated and people go to jail…for failures in compliance – rather than security – the commercial world will continue to pursue compliance rather than security as their primary goal. [15], p. 71.

Julisch defines security compliance as follows: "Security compliance, in IT systems, is the state of conformance with externally imposed functional security requirements and of providing evidence (assurance) thereof" ([15], p. 72). He defines the security compliance problem as follows:

> *Definition*: Given an existing IT system *S* and an externally imposed set *R* of security requirements. The *Security Compliance Problem* is to make system *S* comply with the security requirements *R* and to provide assurance that an independent auditor will accept as evidence of the compliance of system *S* with requirement *R*. [15], p. 72.

Following the definition above, we would like to formulate the *Compliance Problem*, as follows: (1) to make a legal machine's software *S* comply with requirements *R* that relate to a law *L*; and (2) to provide assurance that an independent auditor will accept as evidence (Figure 7).

We have simply added a law *L* to Julisch's formulation. The semi-formal definitions above can only serve as a first iteration. In practice it will be difficult for solutions to the problem to result in a "yes" or a "no". In practice, more elements are

involved. Feedback loops would improve *S*, *R* and *L*. The conceptualization of *L* may involve different elements, depending on the abstraction level. *L* may stand for a legal principle, a statute or a provision.

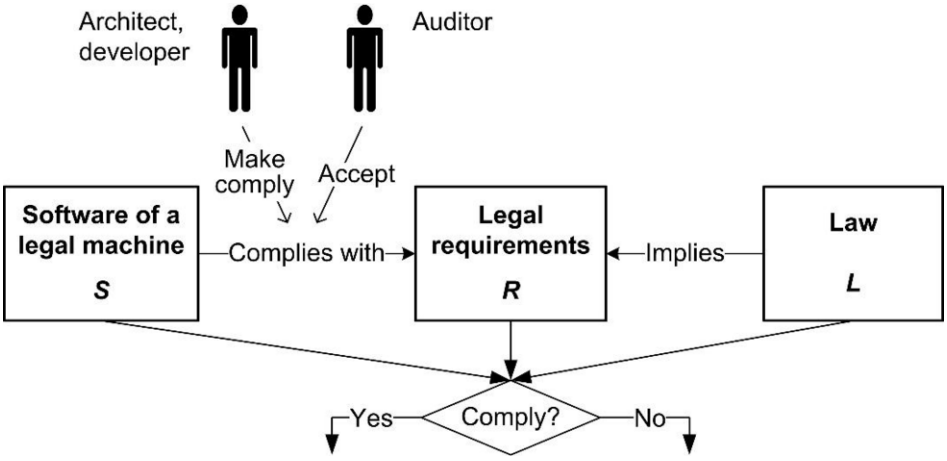


**Figure 7.** The compliance problem

Attempts to formalize the law in the context of the software compliance problem will meet complexity issues. Failure to understand the law is one of the reasons for noncompliance. This failure can be examined from the software development perspective, and also from the legal perspective. A freshman can barely understand the meaning of the law when reading an isolated statute. Similarly, the compliance problem can hardly be reduced to checking a box. Hence, the law is not easily interpreted by the developers whose purpose it is to enforce the law.

The variety of legal areas to consider is a challenge. There is a legal challenge even in a specific domain, such as the design of a federation of digital identities (an identity ecosystem) as an identity management system on the internet [16], p. 238.

There is no silver bullet to solve the compliance problem. This is also explained for an IT compliance framework by Bonazzi, Hussami, and Pigneur, a work which is worthy of special attention [17]. Two dimensions, *Legal* and *IT*, and two kinds of sources for regulations with which a company must comply, *External* and *Internal*, are depicted by rectangles (Figure 8). Different alignments are represented by arrows that point to the artifact that is defined.

Every concept in Figure 8 denotes a broad field. Corporate noncompliance, corruption, etc., are just a few examples of violations. Noncompliance can be civil, criminal, or administrative, but also reputational or market-based. To analyze a company for compliance, legal norms in various branches of the law should be taken into account. This would make a task that is barely feasible for one analyst.

## 8. Conclusions

We have attempted to define the software compliance problem. Software transparency is an issue. The visualizations serve as symbolizations on the way to formalizations.

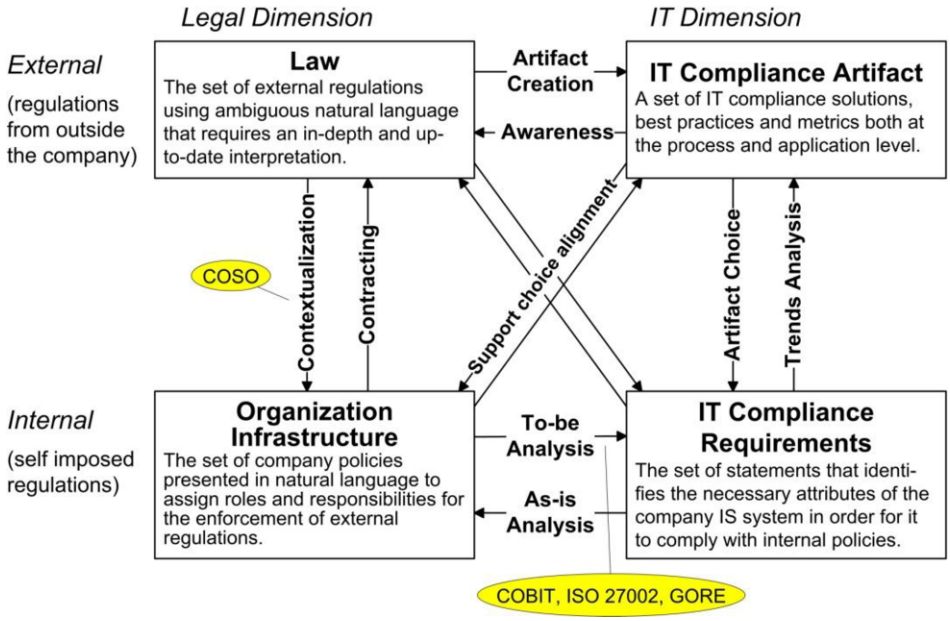


**Figure 8.** The regulation and IT alignment framework adapted from [17]

We depart from the view that machines are tools, and instead take the view that legal machines are legal actors capable of triggering institutional facts. We have identified the transparency problem and formulated two requirements for legal machines. However, implementing these requirements is not an easy task, even if it is possible for it to be achieved at all. There is no silver bullet for attacking regulatory compliance requirements – no one-off, best-of-breed solution. Moreover, there are different contexts of justice, and e-procedures do not guarantee justice.

In the legal domain, mechanical judges are undesirable and machine-assisted decision making outweighs machine-based decision making.

In practice, an auditor can certify that a computer program is compliant with a specific rule, a law or an audit guide. However, the answer "yes" can barely be given about compliance with the whole body of law. The reason is that an expert can understand the legal meaning – in other words, the Ought of this rule, this law or this audit guide respectively. However, understanding the legal meaning of the whole body of law can barely be achieved. Hard cases in law demonstrate that even judges may have separate opinions. Therefore a more experienced auditor can foresee more risks of noncompliance.

## Acknowledgement

V. Čyras has been supported by the project "Theoretical and engineering aspects of e-service technology development and application in high-performance computing platforms" (No. VP1-3.1-ŠMM-08-K-01-010) funded by the European Social Fund.

## References

[1] Kelsen H. *Pure Theory of Law*. 2nd ed., Max Knight, trans. (Reine Rechtslehre, 2. Auflage. Deuticke, Wien, 1960) Berkeley: University of California Press, 1967. 368 p.

[2] Čyras V., Lachmayer F. Legal machines and legal act production within multisensory operational implementations. In: Philipps L., Bengez R. (eds.) *From Specification to Conclusion* (Von der Spezifikation zum Schluss: Rhetorisches, topisches und plausibles Schließen in Normen- und Regelsystemen). Baden-Baden: Nomos, 2013. 156-176.

[3] Čyras V., Lachmayer F. Program transparency for legal machines. *Jusletter IT*, 2014, February. [WWW] http://jusletter-it.weblaw.ch/issues/2014/IRIS.html (accessed 16.07.2014).

[4] Čyras V., Riedl R. Formulating the enterprise architecture compliance problem. In: Čaplinskas A., Dzemyda G., Lupeikienė A., Vasilecas O. (eds.) *Databases and Information Systems. Tenth International Baltic Conference on Databases and Information Systems. Local Proceedings, Materials of Doctoral Consortium, DB&IS 2012, 8-11 July, Vilnius*. Vilnius: Žara, 2012. 142-153. Also [WWW] *CEUR Workshop Proceedings*, Vol-924, http://ceur-ws.org/Vol-924/paper14.pdf (accessed 16.07.2014).

[5] Boer A., van Engers T. An agent-based legal knowledge acquisition methodology for agile public administration. In: *Proceedings of the Ninth International Conference on Artificial Intelligence and Law* (ICAIL '11), June 6-10, Pittsburgh. ACM Press, New York, 2011. 171-180.

[6] Čyras V., Lachmayer F. Multiphase transformation in the legal text-to-program approach. In: Sturm F., Thomas P., Otto J., Mori H. (eds.) *Liber Amicorum Guido Tsuno*. Frankfurt/Main: Vico, 2013. 57-70.

[7] Turing A. Computing machinery and intelligence. *Mind*, 1950, 59(236), 433-460.

[8] Oberle D., Drefs F., Wacker R., Baumann C., Raabe O. Engineering compliant software: advising developers by automating legal reasoning. *SCRIPTed*, 2012, 9(3), 280-313. [WWW], http://script-ed.org/wp-content/uploads/2011/12/oberle.pdf (accessed 16.07.2014).

[9] Bench-Capon T., Araszkiewicz M., Ashley K., Atkinson K., Bex F., Borges F., Bourcier D., Bourgine P., Conrad J. G., Francesconi E., Gordon T. F., Governatori G., Leidner J. L., Lewis D. D., Loui R. P., McCarty L. T., Prakken H., Schilder F., Schweighofer E., Thompson P., Tyrrell A., Verheij B., Walton D. N., Wyner A. Z. A history of AI and Law in 50 papers: 25 years of the international conference on AI and Law. *Artificial Intelligence and Law*, 2012, 20(3), 215-319.

[10] Vasilecas O., Kalibatiene D., Guizzardi G., Towards a formal method for transforming ontology axioms to application domain rules. *Information Technology and Control*, 2009, 38(4), 271-282.

[11] Schweighofer E., Lachmayer F. Ideas, visualisations and ontologies. In: Visser P.R.S., Winkels R.G.F. (eds.) *Proceedings of the First International Workshop on Legal Ontologies, LEGONT '97, 4 July, Melbourne, Victoria, Australia*. University of Melbourne, Law School, 2007. 7-13. [WWW] http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.123.2678&rep=rep1&type=pdf (accessed 16.07.2014).

[12] Ogorek R. *A king of judges or a subsumption machine? (Richterkönig oder Subsumtionsautomat?)*. Frankfurt am Main, Klostermann, 1986. XVI, 424 p.

[13] Riedl R. Difficulties of transdisciplinary legal informatics (Die Schwierigkeiten einer Transdisciplinären Rechtsinformatik). In: Schweighofer E., Handstanger M., Hoffmann H., Kummer F., Primosch E., Schefbeck G., Withalm G. (eds.) *Symbol and Magic of Law (Zeichen und Zauber des Rechts)*. Bern: Editions Weblaw, 2014. 175-183.

[14] Silverman M. *Compliance Management for Public, Private, or Nonprofit Organizations*. New York: McGraw-Hill, 2008. 304 p.

[15] Julisch K. Security compliance: the next frontier in security research. In: *Proceedings of the 2008 workshop on New Security Paradigms, NSPW'08*, 22-25 September, Olympic, USA. New York: ACM, 2008. 71-74. [WWW] http://www.nspw.org/papers/2008/nspw2008-julisch.pdf (accessed 16.07.2014).

[16] Schweighofer E., Hötzendorfer W. Electronic identities – public or private. *International Review of Law, Computers and Technology*, 2013, 27(1–2), 230-239.

[17] Bonazzi R., Hussami L., Pigneur Y. Compliance management is becoming a major issue in IS design. In: D'atri A., Saccà D. (eds.) *Information Systems: People, Organizations, Institutions, and Technologies*. Heidelberg: Springer, 2010. 391-398.