Early Lessons Learned in the ENDORSE Project: Legal Challenges and Possibilities in Developing Data Protection Compliance Software

Sandra Olislaegers

Early Lessons Learned in the ENDORSE Project: Legal Challenges and Possibilities in Developing Data Protection Compliance Software

Sandra Olislaegers

Tilburg Institute for Law, Technology and Society, P.O.Box 90153, Tilburg 5000 LE Netherlands

s.olislaegers@tilburguniversity.edu
http://ict-endorse.eu, http://tilt.nl

Abstract. Software can help businesses comply with data protection regulation. The development of such compliance software may be desirable because data protection compliance is difficult or burdensome to achieve, and especially for small and medium enterprises. Building software capable of providing proper legal guidance, or even enforcing the law, is, however, non-evident. This paper addresses some of the complexities and opportunities involved in the development of such software from a legal perspective. The paper focuses on the identification of relevant regulation, legal norm conflicts and the implementation of open norms in software.

Keywords: data protection, data protection compliance, compliance software

1 Introduction

As more and more data is handled by computers, data protection compliance is increasingly considered important, both from the perspective of regulators and companies.[1] Companies processing personal data that fall under the definition of 'data controller'[2] are responsible for compliance with European data protection law.[3]

1 According to the 2010 Ernst & Young's Global Information Security Survey, organizations consider achieving compliance with regulations as the most important activity in the context of information security. Ernst & Young (2010) 13th annual global information security survey, retrieved from: http://www.ey.com/Publication/vwLUAssets/Global_information_security_survey_2010_advisory/$FILE/GISS%20report_final.pdf.

2 When a natural or legal persons fall under the definition of "data controller" as defined in the European Directive on Data Protection 95/46/EC (DPD), they are obliged to ensure that the data processed under their responsibility is processed in accordance with the law. A data

However, these companies, and particularly small and medium enterprises, may find it difficult or burdensome to guarantee that their IT systems handle personal data in accordance with data protection law. Reasons for this are that the regulation is complex and requires specific expertise, which is highly costly to obtain. Compliance software could be helpful for data controllers to enable their systems to handle data in a legally compliant manner. I define legal compliance as the state in which a norm-addressee conforms to all legal obligations that apply to him. Legal compliance should be guaranteed at three levels:

1. Adherence with the legal obligations as imposed by data protection law;
2. Compatibility of the enterprise's data handling policies (hereafter: company policies)[4] with data protection law: companies are free to specify company policies within the bounds of law[5]; and
3. Compliance of the actual data processing with both the law and company policies.

The question of what law is, encapsulates two sub questions: where can law be found (how to determine the applicable set of rules given a specific context), and what do these laws mean? Related to these questions are three legal issues that provide further challenges in compliance software development: scoping and fragmentation, norm conflicts, and open norms. I will explore these challenges in this paper.

The first challenge relates to scoping and fragmentation. Data protection regulation can be found in many different sources and the scope of data protection law is extensive and obscure. Moreover, data protection law covers more than what is labeled as 'data protection law, particularly in the context of the development of compliance software, as many auxiliary legal provisions are necessary to enforce data protection legislation by software. The sources needed for the *interpretation* of legal norms are also scattered. For instance, not only the statutes, their explanatory memorandums and court cases are relevant for the interpretation of data protection law, but also opinions voiced by the so-called Article 29 Working Party, an advisory body composed of all European Data Protection Authorities. The Art. 29 Working Party produces non-binding opinions, but these are nevertheless influential. In para. 2,

---

controller is defined in Article 2(d) DPD as "the natural or legal person, public authority, agency or any other body which alone or jointly with others determines the purposes and means of the processing of personal data …". See also Articles 5 and 6(2) DPD.

3 Cf. Article 6(2) DPD. Many other legal provisions in the Directive 95/46/EC on Data Protection apply to both data processors and data controllers, but since the data controller is directly responsible for the conduct of a data processor (see Article 17(2) and (3)), the focus is on data controllers. Cf. Articles 22-29 of the new Data Protection Regulation (Regulation of the European Parliament and of the Council on the protection of individuals with regard to the processing of personal data and on the free movement of such data (General Data Protection Regulation) Brussels, 25.1.2012 COM(2012) 11 final 2012/0011 (COD).

4 Data handling policies, or company policies, are policies that define how personal data is, or should be handled by the specific company. These are stipulated by the company, and may restrict their internal data handling practices further than law does.

5 Companies are specifically given the legal competence to create their own rules on data handling on the basis of Article 6(1)(d) of Directive 95/46/EC on Data Protection. This provision requires data controllers to specify the purposes for data processing. Since the law does not give many guidelines as to what these purposes must look like (only that they must be legitimate, specific and explicit), data controllers are given a very wide discretion.

I will argue that the complexity of applicable law can be managed by limiting the scope of the compliance software.
The second challenge in building legal support tools is the handling of norm conflicts. After explaining how this issue plays a role both in law and in compliance software, I will set forth three options in dealing with this issue in para. 3, being human choice, a closed system with a hardcoded legal hierarchy, and a flexible system that is able to apply legal conflict resolution rules.
The third and final issue, which will be the primary focus of this paper, is related to the prevalence of open norms in data protection law. Because many norms are by nature open textured, they, as I will argue, cannot be hardcoded in compliance software. However, in para. 4 I will demonstrate that, considering compliance software to be a "choice architecture", open legal norms can be incorporated, thereby improving overall compliance with such norms.

2 Sources of law and the complexities of scoping and fragmentation

The first step in achieving legal compliance is identifying the set of relevant legal norms, which in our case amounts to data protection regulation. The applicable norms can be found in written legislation, court cases, uncodified customary law and jurisprudence. These, in turn, can be found in international, supranational (e.g., EU) and national[6] legal sources. The main source of data protection legislation in the EU is Directive 95/46/EC[7], usually addressed as the Data Protection Directive (henceforth the DPD). Another important source of data protection law is the 2002/58/EC Directive on Privacy and Electronic Communications[8]. In order to have legal effect, European directives have to be implemented[9] by the EU Member States in their respective national law systems. How these Directives are implemented is left to the individual Member States. As a result of this discretion, the norms in the Directive can end up in single or multiple national regulations[10], and in different interpretations

6 National sources of law in continental law systems can be, for instance, the Constitution, Acts of Parliament, Royal Decrees, Ministerial Regulations, provincial law and municipal law.

7 Directive 95/46/EC of the European Parliament and of the Council of 24 October 1995 on the protection of individuals with regard to the processing of personal data and on the free movement of such data.

8 Directive 2002/58/EC of the European Parliament and of the Council of 12 July 2002 concerning the processing of personal data and the protection of privacy in the electronic communications sector (Directive on privacy and electronic communications), as amended by Directive 2006/24/EC (the Data Retention Directive) and Directive 2009/136/EC (the Cookie Directive).

9 Implementation means that, after the Directive has been signed by the representatives of each of the 27 EU Member States, the Directive has to be ratified by the national governments of these Member States. Subsequently, a national legislative act is created or amended that includes the legal provisions of the newly adopted European Directive.

10 In the Netherlands, for instance, the DPD is implemented in the Dutch Personal Data Protection Act (*Wet bescherming persoonsgegevens*), while the e-privacy Directive is

of the norms in the directives. Furthermore, because European Directives only aim to harmonize law at a minimum level[11], national legislators have the discretion to enact stricter rules than contained in the Directive, or they can provide new rules where European law does not regulate.[12]

National data protection law of the member states not only consists of the implementation of the DPD, but is supplemented by other national laws. First, it also covers legal provisions contained in other legal domains that relate to personal data processing. This includes labor law (e.g., on how to handle personal data relating to employees for instance), tax law (e.g., on the retention of certain personal data), civil law (e.g., on data retention on the basis of contract), and so on. Second, in the context of compliance software, it includes auxiliary legal provisions that are necessary for the software to properly enforce the relevant legislation. An example here is the legal provisions on the processing of personal data concerning minors in the Dutch Personal Data Protection Act.[13] The system must know until which age people are minors, which differs from country to country, and hence knowledge derived from civil law has to be incorporated in the software as well. As a result of the different levels from which regulation originates and the fact that data protection provisions are to be found in a multitude of legal sources, compliance software development is difficult.

The scoping and fragmentation issue play a significant role in compliance software development. It is important to incorporate all relevant legislation in the software, since not doing so could cause the software to produce incorrect results. But covering the entirety of data protection legislation, including all additional provisions relevant for the processing of personal data derived from labor law, tax law, etc. is almost impossible. However, this should not be an argument against the development of compliance software. A way to manage with the vast amount of legal sources is to limit the legal scope of the software. For instance, domains such as labor law and tax law can be excluded as long as the system is not used in domains where these are relevant. This requires the software to advertise its limits. Of course, the core provisions of the data protection regulation must be covered. Properly informing users about the limits of the software is a distinctive challenge that needs to be addressed by the developers of compliance software.

---

implemented in the Telecommunications Act (Telecommunicatiewet). In Italy, for example, all European directives relating to data protection are implemented in one single act, the Codice In Materia Di Protezione Dei Dati Personali [Italian Personal Data Protection Code].

11 The General Data Protection Regulation, whose draft was presented in January 2012, aims to provide greater harmonization because it will be binding law without the need of implementation in national law. The discretion to impose stricter data protection rules at the national level will remain.

12 See Articles 5 and 37 of the *Wet bescherming persoonsgegevens* [Dutch Personal Data Protection Act]. This is an example of a nationally enacted rule that complements the rules imposed by the European DPD (this Directive does not address the processing of personal data concerning minors).

13 *Ibid*.

3 Legal hierarchy and the complexity of legal norm conflicts

The second step in achieving legal compliance is assessing the set of relevant laws to identify if there are conflicts between legal norms. A norm conflicts exists when norm-addressees have to comply with incompatible legal obligations resulting from different applicable norms.[14] In many cases these conflicts are resolved by explicit conflict resolution rules; a legal provision will indicate that in case of conflict, one of the legal provisions or acts prevails over the other, as is the case with for instance conflicts between European and national law (EU law prevails). Where legislation does not clarify the hierarchy of rules, the following non-codified legal conflict resolution norms provide an answer to the question of which legal rules (or act) prevail over others:

1. Legal provisions of a "higher" legal document prevail over provisions contained in a "lower" legal document (e.g. EU vs. national law)[15;]
2. Where two norms or acts of the same level[16] conflict, specific legislation prevails over general legislation[17] (hereafter: *lex specialis*);
3. Where two norms or acts of the same level conflict, newer legislation prevails over older legislation[18].

With respect to the *lex specialis* norm, the hierarchy of legal norms is dependent on the factual context in which the norm addressee operates. For example, one legal act may provide general rules on the processing of sensitive data while another act may state more specific rules on the processing of health data (a type of sensitive data). Since the latter rules are more specific than the general rules on sensitive data, the latter rules trump the more general rules.

There are three options in addressing legal hierarchy in compliance software development:

1. Let the system flag rule conflicts, and let the human user decide which legal norm prevails;
2. Enable the system to apply legal conflict resolution norms.
3. Hardcode the legal hierarchy in the system.

The first option is undesirable, because the system potentially flags many rule conflicts (there are many exceptions in law) and hence poses a significant burden on the (layman) user. The second option involves the implementation of legal conflict resolution norms, which will then during runtime be applied to the legal rules stored in the system. The legal conflict resolutions rules decide automatically and autonomously which legal rule prevails over the other in case two legal rules produce incompatible outcomes. The third option requires compliance software developers to

14 Legal norm conflicts, for instance, also arise in case someone exercises a right that affects a norm-addressee (think for example of the right, in the EU DPD, of a data subject to request information as to whether and which data are being processed by the data controller), which is in conflict with an obligation which that norm-addressee has to comply with (for example that it is not permitted to give such information to data subjects).

15 *Lex superior derogat legi inferiori*.

16 For example two Acts of Parliament or two Ministerial Regulations.

17 *Lex specialis derogat legi generali*.

18 *Lex posterior derogat legi priori*.

identify and solve all legal rule conflicts in a particular domain and subsequently hardcode the rules in such a manner that no conflicts may arise during runtime. This will result in rules in the system with many conditions, because all exceptions to the legal norm will have to be incorporated in the machine executable version of that norm. This is not a desirable option from a legal perspective, because which exceptions apply depends on the factual context (think of the earlier mentioned example of sensitive data and health data), which would limit the compliance software to very small legal domains [1]. In addition, hardcoding the legal hierarchy is undesirable because law is often subject to change, and changes in the law may affect the hierarchy, which would then have to be revisited. Instead, the dynamics of law require compliance software to be flexible. From the perspective of law the second option is therefore preferable. Moreover, it implements defeasibility of legal norms – meaning that the conclusions derived from applicable rules can be defeated by those of other legal rules, e.g. by exceptions – into the architecture of the system [1].

4 Interpretation of legal norms and open texture

Gathering the relevant legislation and identifying the legal hierarchy renders the set of applicable legislation given a specific context or domain. The next step in achieving legal compliance is to determine the meaning of the norms.

Data protection law contains different types of legal norms[19], but is dominated by open norms.[20] Open norms are intentionally formulated in a vague and open manner, because they must be applicable to a broad range of (unforeseen) situations. They are by nature open textured; they require interpretation in order to have meaning. As Claes et al. state: "(…) [the] meaning [of legal norms] is not encapsulated in the words, but reveals itself in the way the rule is used, followed, interpreted, enforced, and so on" [2, p. 14]. The notion of "open texture" should be distinguished from "vagueness" [3]. "Vagueness should be distinguished from open texture. A word which is actually used in a fluctuating way (such as 'heap' or 'pink') is said to be vague; a term like 'gold', though its actual use may not be vague, is non-exhaustive or

19 Other than open norms, these include:
Substantive norms: norms that "define the legal relationships of people with other people and the state in terms of regulative and constitutive norms, where regulative norms are obligations, prohibitions and permissions, and constitutive norms define what counts as institutional facts in the normative system", cf. [4].
Procedural norms: these norms aim to ensure that substantive norms are effectuated, by imposing instrumental, or procedural rights and obligations;
Competence norms: these norms grant the norm-addressee the right or obligation to create new (legal) rules. An example is the right to conclude a contract, or the obligation to specify purposes before data collection (Article 6(1)b DPD);
Behavioral norms: norms that regulate the behavior of their norm addressees, in the form of rights or obligations.

20 Examples of open norms in the DPD are Articles 6, 7(b-f), 8(b-e), 9, 10(c), 17, etcetera.

of an open texture in that we can never fill up all the possible gaps through which a doubt may seep in. Open texture, then, is something like possibility of vagueness. Vagueness can be remedied by giving more accurate rules, open texture cannot. An alternative way of stating this would be to say that definitions of open terms are always corrigible or emendable." [3] In law, vagueness relates to terms such as 'reasonable care' or 'fair'. Whether or not a person exhibits reasonable care depends on all sorts of factors and the outcome of the weighing can be placed on a scale from naught to full. Open texture is extensional; in principle the words or concepts are clear, but they may become vague in certain border-line cases, i.e. under extreme conditions. Think for example of the concept 'goods', which initially were considered to be physical objects having the property 'tangible'. The theft of electricity (only the theft of goods is criminalized) suddenly makes the concept of goods unclear, because electricity is nontangible, but can apparently be appropriated.

There are two difficulties with (the interpretation of) open norms. The first is that many norm-addressees are unaware of all legal sources necessary to interpret open norms. These sources include court cases, doctrine, authoritative opinions and customary law.[21] The second difficulty is to employ a correct, i.e. legally valid interpretation. With the exception of court cases of the highest court, there is not one single valid interpretation that prevails over others. As a result, open norms are difficult to authoritatively hardcode in a machine executable format.

There are also normative reasons for not hardcoding (interpretations of) open legal norms. First, open norms are enacted by legislators for the purpose of allowing the norm-addressees choose how to give substance to these norms. This choice must be made in the context of the prevailing normative notions in society, but there is a very wide discretion for norm addressees. Open norms, by their nature, appeal to the judgment of norm addressees on the correct application of these norms in context. They are given meaning by use and interpretation in a specific context (think of what I said earlier about open texture) [4, p. 14]. Hard-coding interpretations of open legal norms in software would thus go against the nature and purpose of those norms, and that of the legislator. Second, hardcoding legal norms and therewith depriving norm addressees of their competence to choose an interpretation could cause a "de-moralising" effect, as it "remove[s] opportunities for the individual to engage in moral reasoning and judgment" [5, p. 97]. And third, hardcoded interpretations of open norms would create the effect that norm-addressees are no longer aware of the legal consequences of the interpretation-choice. This is undesirable, given that norm-addressees are legally accountable for their actions, even if the action was not based on their choice; the choice may have adverse consequences for which the norm-addressee may subsequently be held liable.

Because of the prevalence of open texture in data protection law, some legal scholars are skeptical with respect to the implementation of open norms and, hence, the feasibility of (the development of) compliance software. For instance, Koops argues that "techno-regulation as enforcement of a legal norm is problematic if the norm itself is more representationally complex, be it due to openness, fuzziness, [or] contextual complexity (…)" [6, p. 192]. Pagallo argues, in the context of

21 Beware! Customary law and court cases are not only significant for interpretation, but can also contain legal norms.

interpretation of the law, that "... privacy is not a zero-sum game, but concerns personal choices on levels of access and control over information that often depend on the context. Making all the provisions of data protection automatic is simply out of reach" [7]. To some extent these authors are right. It is fair to say that open legal norms cannot be implemented in compliance software at a level where these norms are 'fully automated', which Kitchin & Dodge define as "… automated (technologically enacted), automatic (regulation, discipline, and outcomes are enacted without prompting or direction), and autonomous (regulation, discipline, and outcomes are enacted without human oversight) …" [8, p. 85]. However, the conception that creating compliance software is infeasible because open legal norms are not fully automatable is flawed, because this is not a necessary part of compliance software. Compliance software can (instead) also provide data controllers guidance in applying the law correctly.

4.1 Open legal norms and choice architecture

Compliance software could be designed in such a way that it lets norm-addressees, i.e. data controllers, choose how to interpret open norms, while guiding them in doing so in a legally compliant manner. The goal is thus not to hardcode[22] open legal norms, but to guide data controllers in achieving a greater level of compliance in view of those norms. Next, I will show how this idea can come to life in compliance software by considering the software to be a "choice architecture" [9, p. 6].

In my view, compliance software consists of hardcoded legal knowledge on the one hand, and, where the law cannot be hardcoded of nudging mechanisms, on the other hand. The term "choice architecture" was introduced by Thaler & Sunstein and is defined as: "the context in which people make decisions" [9, p. 6]. The authors define nudging, or libertarian paternalism, as "any aspect of the choice architecture that alters people's behavior in a predictable way without forbidding any options or significantly changing their economic incentives" [9, p. 6]. Examples of (intentional) choice architecture and nudging are the arrangement of food in cafeteria's (people tend to choose food that is placed on eye height) [9, p. 1-3]; the placement of images of flies in urinals for men to aim at, thereby reducing spillage [9, p. 4]; and giving people incentives not to pollute by imposing taxes on those who do [9, p. 196]. With respect to compliance software, choice architecture consists of an interactive environment where data controllers are provided with relevant legal information on the basis of which they are nudged to make personal, context specific decisions that lead to legal compliance.

Thaler & Sunstein identify six "principles of good choice architecture", together forming the acronym "NUDGES" [9, p. 109]:

22 See again the definition of 'fully automated' in Kitchin & Dodge: "… automated (technologically enacted), automatic (regulation, discipline, and outcomes are enacted without prompting or direction), and autonomous (regulation, discipline, and outcomes are enacted without human oversight) …" [8, p. 85].

- iNcentives: the incentives for making good, i.e. wealth increasing choices should be made clear, or visible, and directed towards the right people[23];
- Understand mappings: mapping is defined as "the relation between choice and welfare" [9, p. 100-101]. Good choice architecture helps people to understand the consequences of a choice, both now and later, by giving them the right information before making the choice[24];
- Defaults: designers should be aware of the power of default settings[25] and consider when and how to use defaults;
- Give feedback: "[w]ell-designed systems tell people when they are doing well and when they are making mistakes" [9, p. 99];
- Expect error: "[a] well-designed system expects its users to err and is as forgiving as possible"[26];
- Structure complex choices: good choice architecture provides structure when it is not possible to identify or value all potential options one has when making a choice.

Next, I will explain how these principles can be used for the interpretation and implementation of open legal norms in compliance software.

4.2 Implementation of choice architecture within compliance software

As outlined in the introduction, compliance software should achieve legal compliance at three levels: overall compliance with legal obligations, compatibility of company policies with law and compliance of data processing with both. In practice, this could work as follows.

There are three actors that interact with the software: the data controller, the data subject[27] and the company's data management systems (see Fig. 1). The data controller communicates with the software from the perspective of two distinct roles: i.e. as a Privacy Officer (rule manager or editor), displayed on the top left of Fig. 1, and as a human/machine who wants to access or perform actions on data on the engine side via the company backend system(s), as displayed on the top right of Fig. 1. The software includes at least an engine and an editor. The editor has three functions:

1. Write/modify company policies that specify the rules pertaining to the processing of personal data within the system, and evaluate whether these policies are

23 As Thaler & Sunstein state: "good choice architects can take steps to direct people's attention to incentives" [9, p. 108].

24 In this respect, Thaler & Sunstein discuss government regulation of the cell phone market, whereby the government would require phone companies to disclose information on their fees in a comprehensible manner, as opposed to regulating prices [9, p. 102].

25 [9, pp. 8-9, 13-14, 37-39, 93].

26 [9, p. 96]. An example is a warning light in a car telling the driver that he hasn't fastened his seatbelt.

27 The data subject would interact with the software trough an end-user tool, which is not displayed in Fig. 1.

compliant with legal rules and already existing company policies (this achieves legal compliance at the second level) ;

2. Provide legal information to the data controller; data controllers are for instance reminded to take certain actions that cannot be executed by the system (think of sending a notification of processing to the local data protection authority, see Article 19 DPD) - this contributes to compliance at the first level;
3. Interpret legal norms in a legally valid manner, tailored to the specific business context. This is where the idea of choice architecture comes to play. The starting point for interpreting legal norms is the writing of company policies and the choices expressed in those policies.

The policy engine ensures compliance at the third level. If a data controller, as a machine (i.e. automated data processing) or as a human, wants to access data or perform actions on data, the engine evaluates whether the request is permitted according to the legal rules and company policies. The engine will subsequently grant or deny access to the data. I leave the policy engine aside, because, in my view, the policy engine is not the place where legal norms can be interpreted.

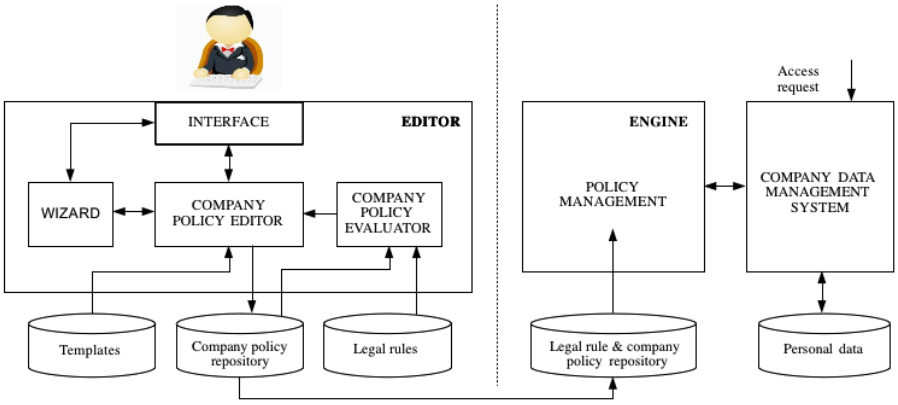

Fig. 1. ENDORSE functional architecture.

The editor consists of a number of components. The Privacy Officer specifies company policies in the company policy editor via the interface. He does this by filling in a template. An example of a template (the top line) and a fully specified company policy (the completed template) is the following:

<Actor>	<Modality>	<Action>	<Destination>	<DataObject>	<DataSource>	<Purpose>	<ProcessingGround>
Data Controller X	MAY	Transfer	(to) Germany	Name	Data subject	Processing order - shipping	Contract

The template elements can be filled in by selecting options from pull down menus. There are different sets of actions, destinations, data objects, purposes, etcetera. Based on these selections, the company policy editor can invoke the wizard, which then enables interpretation of that particular selection. An example is the ambiguity of certain processing grounds that require interpretation of a data controller. The DPD specifies a limited number of processing grounds for the data processing to be legitimate. These grounds include *inter alia* consent (Article 7(a) DPD), performance

of contract (Article 7(b) DPD) and legitimate interests of the data controller (Article 7(f) DPD). For each company policy, a data controller must specify one of these grounds. If the data controller for example selects "legitimate interest of the data controller", the wizard would be invoked because the meaning and scope of this processing ground is ambiguous. The wizard will guide the data controller in interpreting the meaning of legitimate interest. This interpretation process can be rule-based, decision-tree based, or data controllers can be guided by means of presenting plain text (legal information). Depending on the mechanism used, the wizard then either stores the outcome of the interpretation process (e.g. the data controller appropriately uses the processing ground "legitimate interest") (rule-based or decision tree based) or presents the rule manager with the conclusion in text after which the rule manager must choose to either stick with his choice or choose another processing ground.

When a company policy is finished, the rule manager can request the system to evaluate the company policy. This is done by the company policy evaluator, which checks whether the company policy is in conflict with the legal rules and already existing company policies. If the company policy is consistent with those legal rules and company policies, the company policy is stored in the company policy repository, which in turn stores compliant company policies in the legal rule and company policy repository in the runtime environment. If the policy conflicts with either legal rules or existing company policies, the editor will notify the human editor about the conflict and provide suggestions for remediation when possible. Both legal norms and company policies subsequently govern data controllers' access requests and actions on data.

The editor plays a major role in the implementation of choice architecture in compliance software. All six principles[28] of good choice architecture are represented in the architecture and process as described above.

Understand mappings

In the Wizard, the data controller is presented with contextual legal information necessary to make choices regarding legal concepts (interpretation) that fall within the legal bounds of that norm; even when the interpretation process is rule-based or decision tree based. The data controller will always have access to legal information that pertains to a particular legal concept (such as consent, or legitimate interest of the data controller). Such information includes court cases, authoritative opinions, doctrine, etcetera; all presented in a comprehensible manner when possible. The information helps the users to 'understand mappings', because relevant legal information is provided before choices are made, which in turn helps data controllers to become aware of that choice and understand its potential (legal) consequences. In

28 I will not discuss the principle of "iNcentives" in this paper. What can be said about incentives is that I consider achieving legal compliance as an incentive for data controllers to make choices, within the compliance software, that lead to legal compliance. Legal compliance could be visualized in a number of ways, by e.g. a compliance demonstration tool, which allows data controllers to query the system, and demonstrate that certain actions on data have been performed in accordance with the relevant legal rules. Another (additional) option could be a trustmark or certification mechanism.

addition, information on the possibility of sanctions or other potential legal consequences can be presented that is related to non-compliance of a certain legal provision.

Structure complex choices
Besides giving information, complex choices can be structured where necessary and possible. An example of a complex choice is interpreting "legitimate interest of the data controller". In the Netherlands, there is much case law and authoritative opinions of the Dutch data protection authority on the scope and meaning of this processing ground. Hence, to determine whether a data controller may legitimately use this processing ground, he would have to be guided in interpreting the open norm on the basis of the existing legal material. This guidance can consist of a rule-based or decision tree based dialogue implementing the legal knowledge and thus providing structure. In case of a decision tree based structure the data controller would be presented with series of multiple-choice questions that guide them towards a legally valid choice. A rule-based structure can consist of the implementation of the conditions provided by case law and authoritative opinions in the form of rules. Facts that are necessary to execute the rules and which are not yet present in the system are requested via the Wizard to the rule manager in the form of questions.

Defaults
Defaults are very powerful because people usually stick with default options [9], [10, p. 1-2]. In the development of compliance software, it is very important to be aware of this. An example of a default in the previously described compliance software architecture is an already filled in template, i.e. a default company policy, which is presented on the basis of earlier choices of the rule manager. The rule manager can for example be asked which type of service he wants to employ and he is subsequently presented with a number of choices, such as 'online shop', ' health insurance', 'social network', and so forth. Based on the selection, a set of standard company policies is presented. Because many data controllers do not have the expertise to make good, informed choices with respect to their data handling practices, such standard company policies can be very helpful. What is important from the perspective of libertarian paternalism is that people should be nudged towards the most welfare increasing option without any choice limitations, which in turn requires that Privacy Officers are clearly informed of their options to modify the standard company policies, together with all legal information necessary to make decisions on this.

Expect error and Give feedback
In the process of writing company policies and interpreting open norms or open legal concepts, Privacy Officers are presented with options that are legally valid, thereby reducing the chance of error. Think for example of (domain specific) processing purposes that are presented in a pull down menu when writing company policies, or the implementation of decision trees to structure an interpretation process. The legal validity of company policies is checked by the company policy evaluator, and the rule manager gets feedback on the (non)compliance of that policy.

4.3 Choice architecture and previous work

Within the domain of artificial intelligence and law, many attempts have been made to create systems that can disambiguate open texture by means of case based reasoning [11-13], or automated learning by analogical reasoning or neural networks [14-16]. Within these approaches, the focus is on formally representing legal knowledge in such a way that the system can reason and make decisions on the basis of this knowledge. However, from a legal (purposive) point of view, automated decision-making is not desirable with respect to the interpretation of open norms, as open norms by their nature require the norm-addressee's judgment and interpretation. Hence, a different approach is required. Schafer describes this approach as "new modesty": "[r]ather than aiming at computers that can interpret legal norms autonomously and reach a decision, computers are now mainly described as decision or argumentation support tools." [17]. Schafer proposes, in analogy with speaker dependent voice recognitions software, the use of a technique whereby software learns from the user by asking series of questions. By learning about the user's preferences the system would be able to make predictions of future choices a user would make. Using this approach to, for instance, predict interpretations of legal norms by a particular user, however, is again undesirable for the same reasons mentioned earlier. The method also presumes that the system would be used to interpret norms or rules that have been imposed by the user himself (think of a will or contractual terms). Today, no generally applicable (i.e. domain neutral) software where legal open norms can be disambiguated by human choice has been developed, as also indicated by Schafer [17]. However, for the implementation of choice architecture, inspiration can be drawn from earlier works, such as that relating to the Split-Up project, where they used a factor tree to weigh relevant factors for advising how, according to Australian family law, assets should be distributed after divorce [18], and other works on decision support systems with prevalent user involvement [19-20].

5 Conclusions

In this paper addressed some of the complexities and opportunities in developing data protection compliance software from a legal perspective. I argued that there are three legal complexities that particularly play a role in compliance software development, being the implementation of the wide scope that data protection legislation covers, legal norm conflicts and the implementation of open norms in software. The first complexity can be mitigated by also limiting the scope of compliance software; here it is important that these limitations are advertised to the software users. The issue of legal norm conflicts is best dealt with by implementing rules that can reason about the hierarchy of legal rules. Finally, I discussed how compliance with open norms can be enhanced by implementing choice architecture in compliance software, whereby data controllers are guided in choosing a legally compliant interpretation of such norms. Moreover, I indicated that defaults can be particularly helpful in guiding data controllers to enhance legal compliance.

Acknowledgements

I would like to thank Prof. Ronald Leenes and Dr. Bibi van den Berg for their most valuable suggestions and guidance in writing this paper.

References

1. Hage, J. C.: Reasoning With Rules: An Essay on Legal Reasoning and its Underlying Logic. Kluwer Academic Publishers, Dordrecht (1997)
2. Claes, E., Devroe, W., Keirsbilck, B.: Facing the limits of the law. Springer-Verlag, Berlin (2009)
3. Waismann, F.: Verifiability. In: Flew, A. (ed.) Essays on Logic and Language, vol. 7, pp. 117-144. Blackwell, Oxford (1951)
4. Boella, G., Van der Torre, L.: Substantive and procedural norms in normative multiagent systems. Journal of Applied Logic, vol. 6, p. 152 (2008)
5. Yeung, K.: Towards an Understanding of Regulation by Design. In: Brownsword, R., Yeung, K. (eds.) Regulating Technologies: Legal Futures, Regulatory Frames and Technological Fixes. Hart Publishing, Oxford (2008)
6. Koops, E.J.: The (In)flexibility of Techno-Regulation and the Case of Purpose-Binding. Legisprudence, vol. 5, no. 2, pp. 171-194 (2011)
7. Pagallo, U.: On the Principle of Privacy by Design and Its Limits: Technology, Ethics and the Rule of Law (forthcoming)
8. Kitchin, R., Dodge, M.: Code/space: Software and Everyday Life. The MIT Press, Massachusetts (2011)
9. Thaler, R.H., Sunstein, C.R.: Nudge: Improving Decisions About Health, Wealth, and Happiness. Penguin, New York (2009)
10. Tannenbaum, D.: Information asymmetries in policy defaults: Some defaults speak louder than others. Unpublished manuscript, University of California, Irvine (2010)
11. Atkinson, K. M.: Legal Case-based Reasoning as Practical Reasoning. Artificial Intelligence and Law, vol. 13, no. 1, pp. 93-131 (2006)
12. Prakken, H., Sartor, G: Modelling Reasoning with Precedents in a Formal Dialogue Game. Artificial Intelligence and Law, vol. 6., pp. 231-287 (1998)
13. Kerner, Y. H., Schild, U., Zeleznikow, J.: Developing computational models of discretion to build legal knowledge based systems. In: Proceedings of the 7th international conference on Artificial intelligence and law, pp. 206-213. ACM, New York (1999)
14. Aikenhead, M.: The Uses and Abuses of Neural Networks in Law. Computer and High Technology Law Journal, vol. 12, 31-70 (1995)
15. Lawren, S., Fong, S., Giles, C. L.: Natural Language Grammatical Inference: A Comparison of Recurrent Neural Networks and Machine Learning Methods. In: Wermter, S., Ellen, R., Scheler, G. (eds.) Connectionist, Statistical and Symbolic Approaches to Learning for Natural Language Processing. Lecture Notes in Computer Science. Springer, Berlin, pp. 33-47 (1996)
16. Bench-capon, T.: Neural Networks and Open Texture. In: Proceedings of the 4th international conference on Artificial intelligence and law, pp. 292-297. ACM, New York (1993)
17. Schafer, B.: ZombAIs: Legal Expert Systems as Representatives "Beyond the Grave". SCRIPTed, vol. 7, no. 2, p. 384 (2010)

18. Zeleznikow, J., Stranieri, A.: Modelling discretion in the Split-Up system. In: PACIS 1997 Proceedings, pp 307–320 (1997)
19. Zeleznikow, J., Nolan, J.R.: Using soft computing to build real world intelligent decision support systems in uncertain domains. Decision Support Systems, vol. 31, no. 2, pp263-285 (2001)
20. Zeleznikow, J.: Building Judicial Decision Support Systems in Discretionary Legal Domains. International Review of Computers, Law and Information Technology, vol. 14, pp. 341-356 (2000)