From Real-World Regulations to Concrete Norms for Software Agents – A Case-Based Reasoning Approach

Tina Balke[1], Paulo Novais[2], Francisco Andrade[3], Torsten Eymann[4]

[1]University of Bayreuth, Chair of Information Systems Management
Bayreuth, Germany, tina.balke@uni-bayreuth.de
[2]Universidade do Minho, DI-CCTC, Braga, Portugal,
pjon@di.uminho.pt
[3]Universidade do Minho, Escola de Direito,
Braga, Portugal, fandrade@direito.uminho.pt
[4]University of Bayreuth Chair of Information Systems Management
Bayreuth, Germany, torsten.eymann@uni-bayreuth.de

Abstract. When trying to use software agents (SAs) for real-world business and thereby putting them in a situation to operate under real-world laws, the abstractness of human regulations often poses severe problems. Thus, human regulations are written in a very abstract way, making them open to a wide range of interpretations and applicable for several scenarios as well as stable over a longer period of time. However, in order to be applicable for SAs, regulations need to be precise and unambiguous. This paper presents a case-based reasoning approach in order to bridge the gap between abstract human regulations and the concrete regulations needed for SAs, by developing and using a knowledge base that can be used for drawing analogies and thereby serves as reference for "translating" abstract terms in human regulations.

Keywords: Software Agents, Case-Based Reasoning, Electronic Contracting, Dispute Resolution

1. Introduction

Intelligent inter-systemic electronic contracting is a specific way of forming contracts by electronic means in such a way that contracts are concluded and perfected exclusively by the actuation and interaction of intelligent and autonomous informatics devices capable of autonomous, reactive and proactive behavior, of reasoning, of learning through experiences, of modifying their own instructions and, last but not least, of making decisions on their own and on behalf of others (AI and Law) [35]. In this form of contracting, an important role is played by intelligent software agents (SAs). And these may be fictioned as tools controlled by humans or faced as subjects of electronic commerce, they may be seen as legal objects or as legal subjects [4, 5]. Yet, in any case, it is important to legally consider their own and autonomous will [6]. Thus, within the last years the vision of autonomous software agents conducting inter-systemic electronic contracts on behalf of their principals in the Internet has gained wide popularity and scientists have published a wide number of papers with possible application scenarios [24]. However, when thinking about these scenarios one needs to keep in mind, that the Internet (as an extension of the real-word) and all its users are affected by real-world regulations. Consequently, SAs that act on behalf of their human owners are subject to real-world regulations as well [12]. Neglecting the question of how legal acts by SAs should be interpreted, nevertheless the problem arises that SAs as actors in the Internet need to understand the legal context in which they are acting. Hence when performing legal acts for their principals, SAs need to understand the corresponding human regulations [18] in order to be able to assess when and under which circumstances a regulation is violated and when not and what punishment might follow. One possible relevant issue is the mere consideration of rules and sanctions, especially when considering the communication platforms and the relations between SAs and platforms: if SAs don't abide by the rules, probably they may be put out of the platform and, eventually, they might even be totally destroyed or "murdered" [7]. But another important issue, especially when considering the will of the SA in legal relations, has to do with the consideration of legal rules and the possibility that SAs actually know them and adopt certain standards of behavior according to the legal rules. However, is it reasonable to expect that SAs behave in accordance with legal rules? [13]

This will be especially relevant in situations of on-line dispute resolution, which results in the moving of already traditional alternative dispute resolution "from a physical to virtual place" [11]. This allows the parties not just the ease of litigation, but mainly a simple and efficient way of dealing with disputes, saving both "temporal and monetary costs" [26]. Several methods of Online Dispute Resolution (ODR) may be considered, "from negotiation and mediation to modified arbitration or modified jury proceedings" [21].

Anyway, regardless of the method to be adopted, we must confront ourselves with the existence of different ODR systems, including legal knowledge based systems appearing as tools that provide legal advice to the disputant parties and also "systems that (help) settle disputes in an online environment" [17]. Yet, it is undoubtful that Second Generation ODR in which ODR systems might act "as an autonomous agent" [32] are also on the edge of becoming a way of solving disputes. In considering this possibility, it is not our purpose to question the Katsch vision of the four parties in an ODR process: the two opposing parties, the third party neutral and the technology that works with the mediator or arbitrator [25]. But here, it must be assumed a gradual tendency to foster the intervention of SAs, acting either as decision support systems (DSS) [11] or as real electronic mediators [32]. Surely, this latest role for SAs would imply the use of artificial intelligence techniques through case based reasoning (CBR) and information and knowledge representation. "Models of the description of the fact situations, of the factors relevant for their legal effects allow the agents to be supplied with both the static knowledge of the facts and the dynamic sequence of events" [32]. Of course, representing facts and events would not be sufficient for a dispute resolution, the SA in order to perform actions of utility for the resolution of the dispute also needs to know not only the terms of the dispute but also the rights or wrongs of the parties [32], and to foresee the legal consequences of the said facts and events. Actually, we may well have to consider the issue of software agent really understanding law or, in the way the Dutch doctrine has been discussing about legal reasoning by software agents and its eventual legal responsibility: "are law abiding agents realistic?" [13]

The problem that arises when SAs are to operate under real world conditions is that human regulations are usually written in a quite abstract way and are often open to interpretation [22]. The main reason for this is to cover a large number of cases with the same legal text and to keep regulations stable over a longer period. Thus if being formulated in an abstract way, the same legal text can be applied to several scenarios and only its interpretation needs to be adapted [39]. For instance, German regulations on the obligation in kind, e.g. obligations of a seller who has not sold a specific item, but an item of a certain kind are as follows: (§243 German Civil Code (BGB) [1]):

(1) A person who owes a thing defined only by class must supply a thing of average kind and quality.
(2) If the obligor has done what is necessary on his part to supply such a thing, the obligation is restricted to that thing.

In this case "*average kind and quality*" and "*what is necessary*" are abstract terms/actions that (on purpose) are not properly defined, so that the number of accepted ways for the debitor to fulfill his obligation(s) in kind can be extended without changing existing laws. Furthermore, the study of law itself is not a natural science but is based on hermeneutics where coherence and context are used to solve a given problem. Thus, in the example the fulfillment is linked to the contextual circumstances, leaving more room for interpretation on both sides.

As mentioned earlier, this abstraction and possibility of multiple interpretations that is positive for humans pose severe problems when trying to implement them for SAs where meaning should be precise and unambiguous. In order to tackle this problem, this paper will present a cased-based reasoning (CBR) approach, in which a context depended knowledge-base is set up that can be used for terminological interpretations and comparisons by the SAs. In detail the paper is structured as follows: in order to lay the foundations for the CBR approach, related work dealing with the question of representing knowledge and regulations for SAs will be presented and compared to CBR in chapter 2. Afterwards, in chapter 3.1 CBR and its six steps will be illustrated in more detail. Last but not least, in chapter 3.2 the CBR model will be used to analyze the example just mentioned in the last paragraph. The paper will close with a short summary and conclusion.

2. Related work

After briefly explaining the problem of "translating" abstract human regulations for SAs, in this chapter the related work will be presented. Therefore existing approaches to represent information and rules shall be analyzed. As however, a multiplicity of ways to represent information and regulations exists so far, this paper tries to classify them into 4 categories - namely *rule-based systems*, *ontologies*, *semantic webs* and *case-based reasoning systems* [20] - and will analyze the categories respectively.

2.1 Rule-Based Systems

As the name already indicates, rule-based systems are composed of a finite number of rules. These rules normally can be formulated as conditional clauses of the following form:

IF condition *A* holds, *THEN* it can be concluded that statement *B* is true as well. (If *A* then *B*.)

Thereby the "if"-part of the rule is called proposition or left hand side whereas the "then"-formulation is referred to as conclusion or right hand side. Besides these rules, the knowledge base in rule-based systems consists of facts. Facts, in general, are elements that can be described by a finite amount of discrete values [3]. The coherences between the elements are represented by rules. Both components, the rules and the elements, form the abstract knowledge of the rule-based system.

In order to apply the abstract knowledge to a new context, such as in the case of the context-depended "obligations in kind" mentioned in chapter 1, a detailed context description (i.e. concrete or case-specific knowledge) as well as an inference mechanism are required. Depending on the application, the inference mechanism can either be applied data-driven (forward-linked) or goal-oriented (backward-linked). In the first case, the case specific knowledge is used as initial point for the reasoning process. Starting from the fulfilled assumptions, the rules are used to infer about the truth of the concluding rules. Subsequent, the deduced facts on their part are used as initial points for the further inference process. In contrast, the goal-oriented approach uses the opposite conclusion-direction. Thus, the final situation is taken as initial point and all rules are checked by moving backwards, like in a decision tree where starting from the top-node all subjacent edges and nodes are verified (see figure 1).

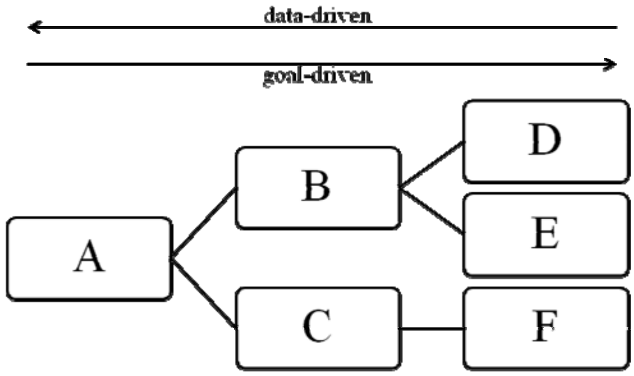

Figure 1. The tree structure of rule-based systems

When judging the applicability of rule-based systems for the "translation"-problem mentioned in the introduction it has to be noticed, that although they foster a well structured analysis, they do not seem applicable. One reason for this is that in rule-based systems all possible situations (or facts) and rules need to be known in advance, leaving not only the problem of pre-definition, but this invokes such a large number of propositions and rules that need to be defined (if one wants to map everything for the SA) that the systems consistency and transparency are more than in danger.

2.2 Ontologies

Another method discussed in literature to move from abstract human regulations to concrete ones for SAs are ontologies (see [39] for example), as their formulation and usage enables programmers of SAs to separate the knowledge of a system (including the terminological knowledge) and the processes. As a consequence of this separation the knowledge can be analyzed, processed and expanded independent of the processes and can be used by SAs for communication purposes. Thereby all knowledge that needs to be used for the communication of SAs needs to be completely represented by the ontology. An ontology itself is a description (like a formal specification of a program) of the concepts and relationships that can exist for an agent or a community of agents. Thus, in the ontology, the individual communication elements correspond to language constructs that are arranged according to a standardized, predetermined form. Besides this integrative form of the communication elements the content of the messages is restricted as well [23]. Although this restriction seems delimiting, it nevertheless ensures that the communication partners use a certain common vocabulary and understand the same terms. This is comparable to the human language: a reasonable communication is only possible if all persons participating associate the same meaning with the same terms. For SAs the establishment of a common ontology means that abstract terms, although having a number of meanings in human interpretations, can be translated to a specific terms that are understood by all SAs the same way, solving the problem of making abstract terms understandable for SAs. Although this idea sounds reasonable and might be applicable for very specific scenarios, as the rule-based systems it brings along complexity problems as soon as these specific scenarios are left. Thus, although ontologies offer standardized text constructs that might be used for negotiation, often these are not being used in the specifications and negotiations (e.g. for reasons of the lack of adaptability of the ontological terms to new situations), but free-text fields are used instead. This however, makes ontologies disadvantageous for bridging the gap between abstract human regulations and specific ones for SAs and illustrates the need for a better concept to solve the problem.

2.3 Semantic Nets

The last group of methods of solution that shall be discussed in this paper - besides CBR approaches - are semantic nets, which were first invented for computers by Richard H. Richens of the Cambridge Language Research Unit in 1956. A Semantic net is net, which represents semantic relations between the concepts. This is often used as a form of knowledge representation. It is a directed or undirected graph consisting of vertices, which represent terms and concepts, and edges that represent the relations between the terms [38] (see figure 2 for example).

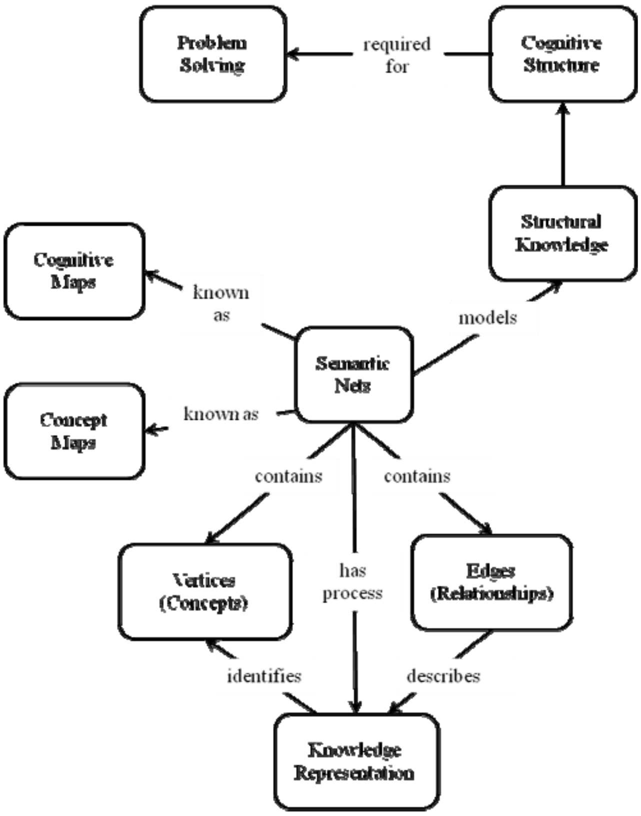

Figure 2. Semantic Nets

By using semantic nets for concepts and terminologies, SAs are given the capability to understand and process freely drafted texts by referring to the components of the nets and their structure to one another. Although this solves one problem occurring when applying ontologies, several further problems remain. Thus, although semantic nets are appropriate for specifying fuzzy terms that consist of several elements (i.e. items with vague component specifications), it is difficult to construct semantic nets that help to define single terms that are hardly divisible such as the term "average" when referring to the kind and quality when dealing with obligations in kind.

3. Cased-based reasoning

As a result of the limitations of the approaches presented so far, this paper will present a mechanism that overcomes these limitations and helps to solve the translation problem introduced in chapter 1: the CBR approach. The fundamental idea of this approach is not to try to "translate" abstract terms directly, but - as done in hermeneutics - to use coherence and context to address the problem [8]. Thereby it is assumed that similar cases normally tend to have similar solutions and similar terms normally tend to have similar meanings, even if they emerge against different backgrounds. Consequently the knowledge gained from solving earlier translation problems can be used as a first approximation when new translation problems appear [36]. This idea of cases that are used for drawing analogies is very well known in legal practice [9] and therefore has the advantage of being [10] widely discussed and reasoned about. A concrete case of case-based reasoning at least consists of a description of the problem (i.e. the abstract terms) and the solution found therefore (i.e. the translation in a specific context). In addition the solution to the problems can be associated with a quality assessment or justifications why a specific solution was chosen for a specific case. The individual cases are stored in a knowledge base which can be resorted to when a new problem arises.

3.1 The 6 steps of Case-Based Reasoning

The six step CBR process model that will be used in this paper was first presented by Roth-Berghofer and Iglezakis [34] who expanded the often cited CBR model of Aamodt and Plaza [2]. The model consists of the six steps retrieve, reuse, revise, retain, review and restore that are integrated into two separate phases, the application and the maintenance phase (see figure 3).

Retrieve. Given a target problem, in the first phase of the model, similar cases[1] that are relevant for solving the new problem are retrieved cases from memory. A case consists of a problem, its solution, and, typically, annotations about how the solution was derived. For example, suppose an agent wants to buy a specific complex grid service (that uses CPU time, disk space and memory for its calculations) in the name of his principal. So far, however he has never bought such a service before and is no familiar with the vocabulary applied. Thus, being a novice in this area, the most relevant experience he can recall is one in which he successfully bought some virtual disk space, i.e. a resource that the service he wants to buy now consists of [19]. The procedure he followed for buying the disk space, together with the justifications for decisions made along the way, constitutes the agent's retrieved case.

Reuse. After the retrieval of similar cases, these solutions from the previous cases have to be mapped to the target problem. This is done in the reuse-phase. The mapping itself may involve adapting the solution as needed to fit the new situation. In

[1] For more information about how to retrieve similar cases and to draw analogies between them see [29] or [14] for example. They, for example, propose to use a memory that organizes experiences (cases) based on generalized episodes. These structures hold generalized knowledge describing a class of similar episodes. An individual experience is indexed by features which differentiate it from the norms of the class (those features which can differentiate it from other similar experiences). As a new experience is integrated into memory, it collides with other experiences in the same generalized episode which shares its differences. This triggers two processes. Expectations based on the first episode can be used in analysis of the new one (analogy). Similarities between the two episodes can be compiled to form a new memory schema with the structure just described (generalization) [28].

the grid service example, this would for example mean that the agent must adapt his retrieved solution to focus on complex services instead of "simple" resources.

Revise. Having mapped the previous solution to the target situation, the next step is to test the new solution in the real world (or a simulation) and, if necessary, revise it. Suppose the agent adapted his grid resource solution by adding the costs for the individual resources up in order to have an idea about the price for the service. After this, he discovers that the aggregated costs for the individual resources are much higher than the costs for the complex service and he offered the seller of the service too much money for it, as his cost calculation did not account for this interrelation - an undesired effect. This suggests the following revision: concentrate on market prices when trying to calculate the costs for a service and do not aggregate the costs of the individual resources instead.

By finishing the revision, the application phase (i.e. the actual problem solving) itself can be closed[2]. However for a CBR system to function properly the knowledge base that it is based on, needs to be sustained. This is done in the maintenance phase which consists of the three sub-phases retain, review and restore.

Retain. After the solution has been successfully adapted to the target problem, together with the resulting experience, it should be stored as a new case in the memory i.e the knowledge base. The agent, accordingly, records his newfound procedure for buying grid services, thereby enriching his set of stored experiences, and better preparing him for future grid service transactions. A second purpose of the retain step is to modify the similarity measures by modifying the indexing structures. However, modifications like this should only be implemented in case-based reasoning if it is possible to track the changes or better measure the impact of those changes.

[2] At first glance, CBR (and especially its application phase) may seem similar to the rule-induction algorithms of machine learning as it starts with a set of cases or training examples and forms generalizations of these examples, albeit implicit ones, by identifying commonalities between a retrieved case and the target problem. The key difference, however, between the implicit generalization in CBR and the generalization in rule induction lies in the point when the generalization is made. A rule-induction algorithm draws its generalizations from a set of training examples before the target problem is even known; that is, it performs eager generalization. In contrast, CBR starts with the target problem and delays implicit generalization of its cases until testing time.

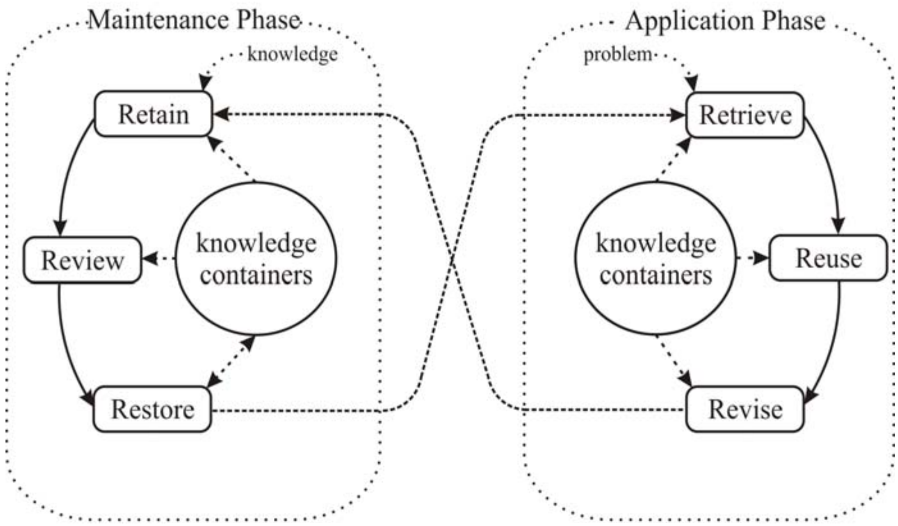

Figure 3. The six steps in CBR

Review. The review step considers the current state of the knowledge containers and assesses their quality. For this purpose appropriate measures need to be found. In literature two fields of corresponding kinds of measures can be distinguished: syntactical measures (i.e. measures that do not rely on domain knowledge) like minimality, simplicity, uniqueness, etc. [33], and semantical measures (i.e. measures using domain knowledge) which check whether the cases are (still) relevant for example [37].

Restore. Finally, the last phase comes into play in case in the review phase it was identified that the quality level of the cases is not as desired. In this case measures to lift the quality level above the critical value are suggested and if approved are being implemented [34].

After having had a look at the CBR model and its six steps in general, in the next chapter, the model shall be applied to the obligation in kind example given in the introduction in order to show the CBR potentials for helping to make abstract terms understandable for SAs. Thereby special focus will be on the potential prerequisites and problems within the six steps as well as potential solutions to these.

3.2 Applying the Case-Based Reasoning Approach

After explaining the general CBR approach, the question arises how it can help with "translation" abstract legal terms for SAs. Therefore the example given in the introduction (concerning the "obligations in kind") shall be recalled. One example where this regulation applies is the domain of cloud computing. The term cloud computing describes the idea that similar to other services - such as electrical power, the telephone, gas or water, in which the service providers seek to meet fluctuating customer needs, and charge for the resources based on usage rather than on a flat-rate basis - IT-services are sold over the Internet [15]. Examples of such IT-services are storage space, server capacity, bandwidth or computer processing time. Cloud computing envisions that in contrast to traditional models of web hosting where the web site owner purchases or leases a single server or space on a shared server and is charged a fixed fee, the fixed costs are substituted by variable costs and he is charged upon how much he actually uses over a given period of time. The negotiation of the cloud services is performed by SAs that automatically react to changes in the resource needs and buy the additional resources needed. The contracts thereby do not concentrate on specific resources (e.g. a specific part of a certain server as storage space or a specific processor that shall be used for the calculations) but feature obligations in kind (i.e. only the general "storage" service, etc. is fixed in the contracts). The reason for this is that the service suppliers try to optimally use their capacity and therefore allocated and reallocate all services continuously depending on the total demand in the network. That's why in cloud computing contract normally service-packages are offered, leading to problems in the comparability for software agents. This problem is intensified by the fast development in the IT sector, leading to a steady increase in the possible component that can be used for a cloud service.

So how could CBR help to solve this translation problem, i.e. how can SAs learn to reason about very general legal terms such as "average kind and quality" and "what is necessary", etc.? To start the explanation, we would like to recall the general CBR-idea: namely the usage of coherence and context to address. As mentioned in chapter 3.1 it thereby is assumed that similar cases normally tend to have similar solutions and similar terms normally tend to have similar meanings, even if they emerge against different backgrounds. This means that in order to be applicable for the "translation"-example, the SA needs a knowledge base that is filled with at least a few cases. If no similar cases exist, the SA first of all needs to be trained, meaning that it has to pass the decision to his principal who then makes that decision and gives the result to the SA who then is able to fill his knowledge container. As the cases are the fundamental elements of CBR and everything else is based upon them, the case-definition is a first very important step to look at. For practical reasons, normally all cases have a particular name, a set of empirical circumstances or facts, and an outcome representing the results of the problem for the decision, solution or classification it poses [16]. These characteristics of a case are then written down in a systematical structured way, such as in form of tables or vectors, etc. Looking at the cloud example, the set of facts might include the original contract formulations (including the related juristic paragraphs and their formulations), the services requested delivered and some quality criteria of the services (e.g. availability or speed), whereas the outcome description could comprehend in how far the measured quality criteria represent the expected ones and whether any difference can be attribute to the obligation in kind. Once, a knowledge based with a few cases exists, the reasoning process can be started, i.e. the SA has to find a similar case and needs to go on by analyzing which decisions were made in this case and why. A very general scheme for the deduction step was presented by Ashley [9]:

Start: Problem description.

```
A: Process problem description to match terms in case
database index.
B: Retrieve from case database all candidate cases
associated with matched index terms.
C: Select most similar candidate cases not yet tried.
If there are no acceptable candidate cases, try
alternative solution method, if any, and go to F.
Otherwise:
D: Apply selected best candidate cases to analyze/solve
the problem. If necessary, adapt cases for solution.
E: Determine if case-based solution or outcome for
problem is successful.
If not, return to C to try next candidate cases.
Otherwise:
F: Determine if solution to problem is success or
failure, generalize from the problem, update index
accordingly and Stop.
```

Based on this general algorithm, in literature five paradigmatic approaches comparing the existing knowledge base with new cases can be found; these are: statistically-oriented, model-based, planning / design-oriented, exemplar-based, and adversarial or precedent-based approaches[3].

Out of these five, for the cloud example, the model-based paradigm is of special interest, as this paradigm, cases are examples explained in terms of a theoretical model of the domain task. Thus, if the SA is confronted with a new case, it has to determine, if the past explanations (e.g. of the legal terms) apply [30]. Similar cases in the cloud computing-"translation" example might for example be transactions about IT services that included §243 of the German Civil Code which the SA has concluded before. Starting from these similar cases, in the next step, the SA is to analyze the similarities between his new problem and the old cases. Thereby he has to include the context of the cases in its reasoning. Finally, if a decision is made concerning the interpretation or the translation of the new terms, the mapping needs to be tested in reality. This can either be done by the software agent sending its decision to its principal for validation purposes or by closing the deal and waiting for the outcome (which is then checked against the expected outcome). Finally, after the "translation"-problem is being solved and the outcome is clear in a next step, the quality of the new solution needs to be assessed. This is either done by comparing the achieved result with the expected one or by transferring the evaluation to the principal who can make more elaborated decisions. Afterwards the SA can decide whether to include this new case in the knowledge base or not. Normally it will choose to do so if the new case expands its knowledge base in a sensible way, e.g. if it has not stored any cases concerning the vocabulary of §243 of the German Civil Code before. This knowledge adaptation is completed by maintaining the knowledge base. Thus in the legal context it might happen that a paragraph or a law is changed or interpreted differently in the course of time.

[3] For a detailed description of the paradigms see [9].

4. Conclusions

As mentioned in the introduction, when wanting to move to electronic environments where intelligent software agents not only conclude contracts on behalf of their human owners but also may participate in dispute resolution, many challenges need to be overcome. One of them is the problem of the abstractness of human regulations. The paper presented several approaches that can be found in literature (e.g. ontologies, etc.) trying to tackle the problem, which however have several drawbacks and consequently may not be the best choice. That is why the paper presented the CBR reasoning concept and explained how it could help to solve the problem. In contrast to many other approaches, CBR has the advantage of being applicable even to the new problems to be solved (e.g. the understanding of new abstract terms)[4] if the problem is badly structured or described incompletely, if the knowledge base starts with a relatively small number of cases or if the rules between the different components are not all known [27, 31]. For this reason and due to its relative simplicity, in the view of the authors, it is well suited for addressing the "translation"-challenges lying ahead and should be researched in more detail.

5. Acknowledgements

The work of Paulo Novais and Francisco Andrade described in this paper is included in TIARAC - Telematics and Artificial Intelligence in Alternative Conflict Resolution Project (PTDC/JUR/71354/2006), which is a research project supported by FCT (Science & Technology Foundation), Portugal.

5. References

1. German civil code (bgb). DTV-Beck, September 2008. 62nd edition.
2. Aamodt and E. Plaza. Case-based reasoning: Foundational issues, methodological variations, and system approaches. AI Communications, 7(1):39-59, 1994. IOS Press.
3. Abraham. Rule-based expert systems. In P. H. Sydenham and R. Thorn, editors, Handbook of Measuring System Design, pages 909-919. John Wiley & Sons, 2005.
4. F. Andrade, P. Novais, J. Machado, and J. Neves. Contracting agents: legal personality and representation. Intelligence and Law, 15(4):357-373, 2007. ISSN 0924-8463.
5. F. Andrade, P. Novais, J. Machado, and J. Neves. Intelligent contracting: Software agents, corporate bodies and virtual organizations. In Establishing The Foundation of Collaborative Networks, volume 243, pages 217-224. Springer Boston, 2007.
6. F. Andrade, P. Novais, and J. Neves. Divergence between will and declaration in intelligent agent contracting. In ICAIL 2007 - Eleventh International Conference on Artificial Intelligence and Law, Stanford University, Stanford, California, USA, June 4-8 2007, pages 289-290. ACM Press, 2007. ISBN 978-1-59593-680-6.

[4] Although CBR reasoning can be applied if only a small knowledge base is available, the more cases it can build on the better it tends to work.

7. M. Apistola, F. M. T. Brazier, O. Kubbe, A. Oskamp, J. E. J. Prins, M. H. M. Schellekens, and M. B. Voulon. Migrating agents: Do sysadmins have a license to kill? In Proceedings of the 3rd International SANE Conference (SANE 2002), 2002.
8. K. D. Ashley. Arguing by analogy in law: A case-based model. In D. H. Helman, editor, Analogical Reasoning, pages 205-224. Kluwer Publishers, 1988.
9. K. D. Ashley. Case-based reasoning and its implications for legal expert systems. Artificial Intelligence and Law, 1(2-3):113-208, 1992.
10. K. D. Ashley. An ai model of case-based legal argument from a jurisprudential viewpoint. Artificial Intelligence and Law, 10:163-218, 2002.
11. E. Bellucci, A. Lodder, and J. Zeleznikow. Integrating artificial intelligence, argumentation and game theory to develop an online dispute resolution environment. In ICTAI-2004 - 16th IEEE International Conference on Tools with Artificial Intelligence, pages 749-754, 2004.
12. G. Boella, L. van der Torre, and H. Verhagen. Introduction to the special issue on normative multiagent systems. Autonomous Agents and Multi-Agent Systems, 17:1-10, 2008.
13. F. M. T. Brazier, O. Kubbe, A. Oskamp, and N. J. E. Wijngaards. Are law-abiding agents realistic? In Proceedings of the workshop on the Law of Electronic Agents (LEA2002), pages 151-155, 2002.
14. M. H. Burstein. A model of learning by analogical reasoning and debugging. In Proceedings of the National Conference on Artificial Intelligence, Washington, D. C., 1983.
15. N. G. Carr. It doesn't matter. Harvard Business Review, pages 41-49, May 2003.
16. Chorley and T. Bench-Capon. Agatha: Using heuristic search to automate the construction of case law theories. Artificial Intelligence and Law, 13:9-51, 2006.
17. De Vries, R. Leenes, and J. Zeleznikow. Fundamentals of providing negotiation support online: the need for developping batnas. In Proceedings of the Second International ODR Workshop, pages 59-67, Tilburg, 2005. Wolf Legal Publishers.
18. F. Dignum. Agents, markets, institutions, and protocols. The European AgentLink Perspective, pages 98-114, 2001.
19. T. Eymann, M. Reinicke, W. Streitberger, O. Rana, L. Joita, D. Neumann, B. Schnizler, D. Veit, O. Ardaiz, P. Chacin, I. Chao, F. Freitag, L. Navarro, M. Catalano, M. Gallegati, G. Giulioni, R. C. Schiaffino, and F. Zini. Catallaxy-based grid markets. Multiagent and Grid Systems, 1(4):297-307, 2005. IOS Press.
20. O. Geibig. Agentenbasierte Unterstützung Öffentlicher Ausschreibungen von Bauleistungen unter Verwendung von Methoden der Künstlichen Intelligenz. PhD thesis, Universität Duisburg-Essen, 2008.
21. J. Goodman. The pros and cons of online dispute resolution: an assessment of cyber-mediation websites. Duke Law and Technology Review, 4, 2003.
22. D. Grossi and F. Dignum. From abstract to concrete norms in agent institutions. In Lecture Notes in Computer Science, volume 3228. Springer, 2005.
23. T. R. Gruber. Toward principles for the design of ontologies used for knowledge sharing. International Journal of Human-Computer Studies, 43(5-6):907-928, November 1995. Academic Press, Inc., Duluth, MN, USA.
24. R. H. Guttman, A. G. Moukas, and P. Maes. Agent-mediated electronic commerce: a survey. The Knowledge Engineering Review, 13(2):147-159, 1998.
25. E. Katsch and J. Rifkin. Online dispute resolution - resolving conflicts in cyberspace. Jossey-Bass Wiley Company, 2001.
26. L. Klaming, J. Van Veenen, and R. Leenes. I want the opposite of what you want: summary of a study on the reduction of fixed-pie perceptions in online negotiations. - "expanding the horizons of odr". In Proceedings of the 5th International Workshop on Online Dispute Resolution (ODR Workshop'08), pages 84-94, 2008.

27. J. Kolodner. Case-Based Reasoning. Morgan Kaufmann Publishers, San Mateo, 1993.
28. J. Kolodner, R. Simpson, and K. Sycara. A process model of case-based reasoning in problem solving. In Proceedings of the Ninth International Joint Conference on Artificial Intelligence, pages 284-290, 1985.
29. J. L. Kolodner and R. L. Simpson. Experience and problem solving: a framework. In Proceedings of the Sixth Annual Conference of the Cognitive Science Society, pages 2-9, Boulder, CO., 1984.
30. P. Koton. Using Experience in Learning and Problem Solving. PhD thesis, MIT, 1988.
31. D. Leake. Cbr in context: The present and future. In D. Leake, editor, Case-Based Reasoning: Experiences, Lessons, and Future Directions, pages 1-30. AAAI Press / MIT Press, 1996.
32. G. Peruginelli and G. Chiti. Artificial intelligence in alternative dispute resolution. In Proceedings of the workshop on the Law of Electronic Agents (LEA 2002), 2002.
33. T. Reinartz, I. Iglezakis, and T. Roth-Berghofer. Review and restore for case based maintenance. In E. Blanzieri and L. Portinale, editors, Advances in Case-Based Resoning, pages 247-259. Springer, 2000.
34. T. Roth-Berghofer and I. Iglezakis. Six steps in case-based reasoning: Towards a maintenance methodology for case-based reasoning systems. In H.-P. Schnurr, S. Staab, R. Studer, G. Stumme, and Y. Sure, editors, Professionelles Wissensmanagement: Erfahrungen und Visionen (Proceedings of the 9th German Workshop on Case-Based Reasoning (GWCBR)), pages 198-208. Shaker-Verlag, 2001.
35. S. Russell and P. Norvig. Artificial Intelligence - A Modern Approach. Prentice Hall, 2002.
36. R. Schank. Dynamic Memory: A Theory of Learning in Computers and People. Cambridge University Press, 1982.
37. Smith and M. Keane. Remembering to forget: A competence.preserving case deletion policy for case-based reasoning systems. In Proceedings of the 13th International Joint Conference on Artificial Intelligence, pages 377-382, 1995.
38. J. F. Sowa. Semantic networks. In S. C. Shapiro, editor, Encyclopedia of Artificial Intelligence. Wiley, 1987.
39. J. Vázquez-Salceda, H. Aldewereld, D. Grossi, and F. Dignum. From human regulations to regulated software agents' behavior. Artificial Intelligence and Law, 16(1):73-87, 2008. Kluwer Academic Publishers.