Angelic Environment: Support for the Construction of Legal KBS

Latifa Al-Abdulkarim
University of Liverpool
Liverpool, UK

Katie Atkinson
University of Liverpool
Liverpool, UK

Sam Atkinson
University of Liverpool
Liverpool, UK

Trevor Bench-Capon
University of Liverpool
Liverpool, UK
tbc@csc.liv.ac.uk

ABSTRACT

This paper describes a development environment for the Angelic Methodology. The environment comprises a database to store the domain theory produced by the methodology, together with an extensible set of tools which display and use the stored knowledge to support development, verification, and refinement. The environment is described and illustrated by using it to capture an analysis of the widely studied domain of property law relating to the pursuit of wild animals. The implementation we present provides an important step in moving a formal model of argument towards a tool that can be used in practice and tailored to a domain as needed, and which paves the way for widespread application of the fruits of AI and Law research to legal practice.

1. INTRODUCTION

Although AI and Law has been around for more than three decades, and there has been much interesting research [12], there has been disappointingly little penetration into legal practice. One important exception is the approach to moving from written regulations to an executable expert system based on [24], which has been developed through a series of ever larger companies: Softlaw, Ruleburst and, currently, Oracle. In the past year or so, however, there has been an unprecedented degree of interest in AI and its potential for supporting legal practice. This is evidenced by articles in the legal trade press such as *Legal Business*[1] and *Legal Practice Management*[2]; national radio programmes such as *Law in Action*[3] and *Analysis*[4] and Professional Society events, such as panels run by the Law Society of England and Wales[5]. The legal profession has never been so interested in, and receptive to, the possibilities of AI for application to their commercial activities. There are, therefore, opportunities which need to be taken.

One of the major lessons that can be drawn from Softlaw and its successors is the need for a methodology. A methodology gives some assurance to clients that their engagement with AI has some prospects of success: they are not so interested in furthering research activities as in increasing profits and the use of an established methodology can allow them to know how their particular problem will be addressed and what will be produced at the end of the process. Equally important, as the Softlaw experience also showed, is the existence of tools to support the methodology. Such tools reinforce the methodology, make it more teachable and reproducible and shorten the development time. In this paper we describe a support environment for the recently developed Angelic methodology [3], which we believe can be used to develop applications that will meet some of the current needs of the legal profession.

Section 2 briefly describes the Angelic methodology. The support tools comprise a database and an extensible set of tools built on that database. Section 3 describes the database and section 4 the currently available tools. Section 5 illustrates the use of the environment, by reference to the widely studied property law cases involving wild animals ([16], [22], [11], [23] and [25] amongst others). These cases were subjected to the Angelic methodology in [3] and we use the analysis in that paper unchanged to populate our database, allowing direct comparison. This domain is also used for the examples throughout the paper. For those unfamiliar with these cases, Section 5 begins with a description of the cases we use, which may also be useful to help explain some of the examples. We also support the dimensional analysis advocated in [2], using the dimensions identified in [13], so that facts of a case can be understood in the manner of [26]. Section 5 also provides the visualisation of a second domain, relating to the Automobile Exception of the Fourth Amendment. Section 6 offers some concluding remarks.

[1] *AI and the law tools of tomorrow: A special report.* www.legalbusiness.co.ukindex.phpanalysis4874-ai-and-the-law-tools-of-tomorrow-a-special-report. All websites accessed in January 2017.

[2] *The Future has Landed.* www.legalsupportnetwork.co.uk. The article appeared in the March 2015 edition.

[3] *Artificial Intelligence and the Law.* www.bbc.co.uk/programmes/b07dlxmj.

[4] *When Robots Steal Our Jobs.* www.bbc.co.uk/programmes/b0540h85.

[5] The full event of one such panel can be seen on youtube at www.youtube.com/watch?v=8jPB-4Y3jLg. Other youtube videos include Richard Susskind at www.youtube.com/watch?v=xs0iQSyBoDE and Karen Jacks at www.youtube.com/watch?v=v0B5UNWN-eY.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from Permissions@acm.org.

ICAIL '17, June 12 - 16, 2017, London, UK
Copyright 2017 ACM 978-1-4503-3522-5/15/06 ...$15.00.
http://dx.doi.org/10.1145/2746090.2746103.

2. THE ANGELIC METHODOLOGY

The basic Angelic Methodology was presented in [3], and evaluated in [1]. This evaluation led to the extensions to the methodology presented in [4] and [2]. The environment described in this paper is based mainly on the methodology presented in [3], but also supports some of the developments reported in [4] and [2]. In particular both the Boolean approach of [3] and the dimensional approach of [2] are supported.

The Angelic methodology is based on an Abstract Dialectical Framework (ADF) [18] and [17]. ADFs are defined in [17] as follows:

> **Definition 1**: An ADF is a tuple $ADF = < S, L, C >$ where
>
> – S is the set of statements (positions, nodes).
>
> – L is a subset of $S \times S$, a set of links.
>
> – $C = \{C_{s \in S}\}$ is a set of total functions $C_s : 2^{par(s)} \rightarrow \{t, f\}$, one for each statement s. C_s is called the acceptance condition of s.

As used in the Angelic methodology there are no cycles and the ADF effectively forms a tree running from base level factors at the bottom (the leaf nodes) to a verdict at the top (the root node). The top levels of the structure correspond to the logical model of the Issue Based Prediction system [19] and the lower levels to the abstract factor hierarchy of CATO [5], an overall structure very similar to that of CABARET [29]. Once the structure has been established, acceptance conditions are supplied for each node. These take the form of *tests*: sufficient conditions for accepting or rejecting the parent node, in terms of the status of its children. These tests are arranged in a priority order and a default is given to cover cases where none of the tests is satisfied.

The methodology of [3] is enhanced in [2] by linking the base level factors to facts through the use of dimensions [28]. Base level factors are equated with points on, and ranges in, dimensions, as proposed in [26]. This enables degrees of presence and absence of factors to be represented, and the computation can be performed by regarding the resulting expressions as formulae of fuzzy logic [31] and then computing the values for parent nodes in a manner similar to [14]. This approach was evaluated in [2] with reference to the US Trade Secrets Domain of [5] and [19]. The Angelic methodology thus draws upon three decades of AI and Law research, and has been evaluated throughly in a research context. We feel, therefore that it provides a sound foundation on which to base our practical support environment.

3. DATABASE DESIGN

At the heart of our environment is a database, designed to store and to make accessible the product of a domain analysis conducted using the Angelic methodology. This section describes the database design, which was produced using standard database design methods as found in textbooks such as [20]. The database was implemented using Oracle. Figure 1 provides an overview in the form of an entity-relationship diagram, produced from the implemented database. Subsequent subsections will describe each of the components.

3.1 Abstract Dialectical Framework Structure

The key Tables are those which hold the structure of the ADF. These are a Table for the *Nodes* (the set S of Statements in the above definition of an ADF) and a Table for the *Links* (the set L in the above definition).

3.1.1 Nodes

Nodes in an ADF are statements. These statements can be considered to be one of a number of types, as described in [4], namely Verdict, Issue, Abstract Factor, and Base Level Factor. In our Table for Nodes we have:

- Node ID. For internal use when referring to particular nodes.
- Node name. This is a short name, useful for reference in diagrams, programs and the like.
- Domain. This is the ID of the domain to which the node relates, in case the database is holding several different domains, or subdomains.
- Statement. This is a full version of the statement, useful for clarifying the intended meaning, and for use in explanations.
- Node Type. This is one of Verdict, Issue, Abstract Factor, and Base Level Factor.
- Dimension. This is used only for Base Level factors, and indicates the dimension from which the factor is taken.
- Threshold. This is again used only for Base Level Factors, and indicates the point on the dimension at which the factor should be considered present rather than absent.
- Provenance. The statements typically get introduced into the domain through statute, commentary or a precedent case. The provenance indicates where the statement originated, allowing for further investigation, or citation.

3.1.2 Links

Nodes (other than nodes of type *verdict*) are linked to parent nodes, and nodes other than those of type Base Level are linked to children. The Link Table holds the complete set of such links.

- Link ID. Used for internal reference.
- Parent Node. Holds the parent node.
- Child Node. Holds one of the children of the parent.
- Polarity. As in [5] children can be a reason for the acceptance or rejection of their parent. Polarity thus holds either '+' or '-' which can be used to label the link and so indicate its influence on the parent.

3.2 Acceptance Conditions

As in [3], the acceptance conditions take the form of a set of tests, each of which gives a sufficient condition for accepting or rejecting the node.

3.2.1 Tests

- Test ID. For internal reference
- Head Node. This the Node the acceptability of which the test is used to determine.
- Priority. The tests for a node are tried in order of a priority. This indicates the priority of the test relative to the Head Node. Priority 0 is used to denote a special test, used to compute the value of the Head Node when the non-Boolean, dimensional, approach of [2] is taken.
- Test Result. Passing the test may indicate that the head node should be accepted or rejected. This is used to indicate which of these is true of the particular test.

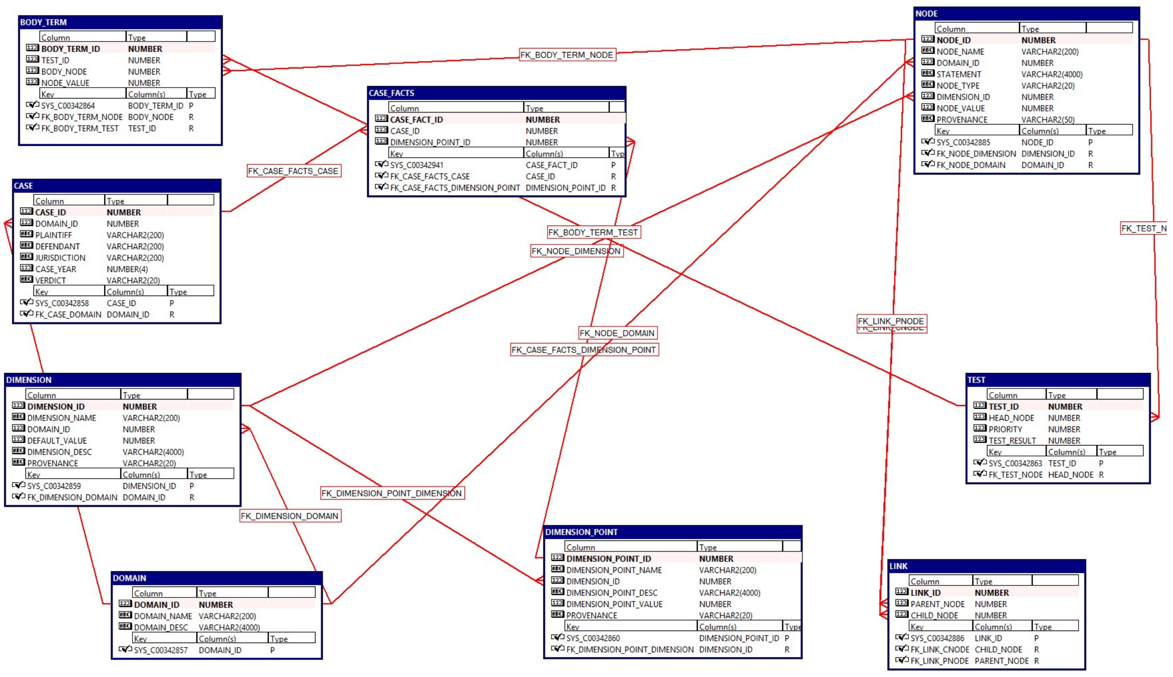


Figure 1: Entity Relationship Diagram

3.2.2 BodyTerms

The tests are conjunctions of other nodes selected from the children of the head Node. The relevant nodes are held in the Body Term Table.

- Body Term ID. For internal reference.
- Test. Holds the ID of the Test of which the body term forms a part.
- Body Node. A node used in the Test.
- Node Value. Nodes used in the test may be required either to be acceptable or to be unacceptable. This indicates which is required.

Where the Body Node is null, the test is unconditional (i.e. a default), and so the result supplies the default for the node to which the test relates.

3.3 Domain Theory

As well as the ADF structure, we need to hold information about the domain. This data is held in the following Tables.

3.3.1 Domain

This is used for the very basic information about the domain.

- Domain ID. For internal reference.
- Domain Name. A short name for quick and convenient reference.
- Domain Description. A full description of the domain.

3.3.2 Cases

This is used to hold some important details (other than the facts) about the cases used in the domain.

- Case ID. For internal reference to the case.
- Domain ID. The domain of which the case forms part.
- Plaintiff. Cases are normally referred to using the two parties to the case. This holds the plaintiff.
- Defendant. This holds the other party to the case, the defendant.
- Jurisdiction. This holds the Jurisdiction in which the case was tried. Often a domain will contain cases taken from several jurisdictions.
- Case Year. The year in which the case was decided.
- Verdict. The outcome of the case: normally either plaintiff or defendant.

3.3.3 Dimensions

We want to support the links between base level and factors and facts described in [26] and [2], and so we include information on dimensions [6].

- Dimension ID. For internal reference.
- Dimension Name. The short name for convenient reference to the dimension.
- Domain ID. The domain to which the dimension relates.

- Default Value. The dimension is taken to range between 0 and 1, and takes its value for a case from the particular point on the dimension of which the case lies. If the dimension is inapplicable to the case, or no information with respect to the dimension is available, a default value is used.

- Dimension Description. This gives a full description of the dimension if required for clarification or explanation.

- Provenance. Dimensions tend to be introduced into a domain either from the governing statute, or from commentaries, or from particular landmark cases. This says where the dimension has come from, for research or citation purposes.

3.3.4 Dimension Points

Although the dimension is in principle continuous, in practice it is useful to regard it as a series of points (or more generally ranges) moving from an extreme pro-plaintiff position to an extreme pro-defendant position. This table is use to segment the dimension.

- Dimension Point ID. Used for internal reference.

- Dimension Point Name. This is a name for the point (range) on the dimension: normally it will correspond to a base level factor in the node table.

- Dimension ID. The dimension on which this is a point (range).

- Dimension Point Description. This fully describes what is intended to be covered in the range on the dimension.

- Dimension Point Value. The dimension ranges between 0 and 1. This holds the particular point on the spectrum occupied by this dimension point or, in the case of a range, the mid-point of that range.

- Provenance. Like other elements in the domain, dimension points have particular occasions of introduction to the domain, whether in a stature, a commentary or a leading case. This origin is recorded here.

3.3.5 Case Facts

Each case can be considered as a set of facts: namely the points on the dimensions that they occupy.

- Case Fact ID. For internal reference.

- Case. A particular case.

- Dimension Point. The point of a dimension on which this case lies.

The Tables described thus hold all the information produced by an analysis following the Angelic methodology, covering: the domain theory expressed as an ADF, the domain itself and individual cases within the domain.

4. DEVELOPMENT TOOLS

This section describes the facilities to support development in the Angelic environment. The tools are intended to be extensible and developers can write programs to use information in the database in any way they find useful. In this section we describe the tools so far developed. The tools are accessed through the GUI shown in the screen shot of Figure 2. There is some basic information at the top of the screen. Tools are accessed by buttons on the left hand pane, and the larger pane is a display and working area.

4.1 Visualisation

The first tool supports visualisation of the domain. Figure 3 shows a visualisation of the wild animals domain produced using the tool. The visualistion was implemented using the vis.js Javascript library[6]. A Javascript file uses an AJAX call to execute a PL/SQL function that interacts with the Oracle database and returns JSON. The JSON is then passed to the vis.js library, which draws the graph on the canvas. We also used the Bootstrap framework[7] to help build the web-front end and assist with compatibility and portability of the tool across multiple device platforms.

The circular layout of the visualisation so produced can be compared with the more standard tree-like layout shown in, for example, Figure 4 of [3], which contains exactly the same nodes and links. Most obviously the root node, the verdict, appears in the centre rather than at the top. We have some preference for the layout shown: given the typical topology of the ADF encapsulating a domain theory, it is more compact, because the base level factors are better distributed. The various clusters of attributes are clearly shown. Overlaps are not a problem since the layout is not fixed: nodes can be dragged around the screen, bringing their linked nodes with them, so that this can be corrected to the taste of the user. This is a very useful facility for exploring the clustering of and relationships between the nodes. For large structures, it is possible to zoom in (as shown in Figure 4) and out to focus on particular areas.

Presentation in this manner suplies a useful focus for discussion with clients and also facilitates the identification of errors and omission. Even more importantly it can suggest the need for further nodes. If we consider *Capture* in the bottom right we see that it is connected to three child nodes: *NotCaught*, *HotPursuit* and *Vermin*. We may well think that these three are not best represented as siblings: *HotPursuit* and *Vermin* represent an exception to the need to have bodily caught the quarry to be deemed to have captured it. We could introduce an abstract factor, *ActivityToEncourage*, which would act as an exception, and allow for exceptions additional to the hot pursuit of vermin to be added.

4.2 Case Input

This tool facilitates the entry of the facts relating to a particular case. After creating a record for the case in the Case table, the facts can be input using the tool. For each dimension relating to the domain of the case, selectable using a drop down menu, the appropriate point of that dimension can be entered using a drop-down menu offering the set of points on that dimension as options (and a null option in case a default is required for the dimension in question).

The tool is shown in the screenshot of Figure 5.

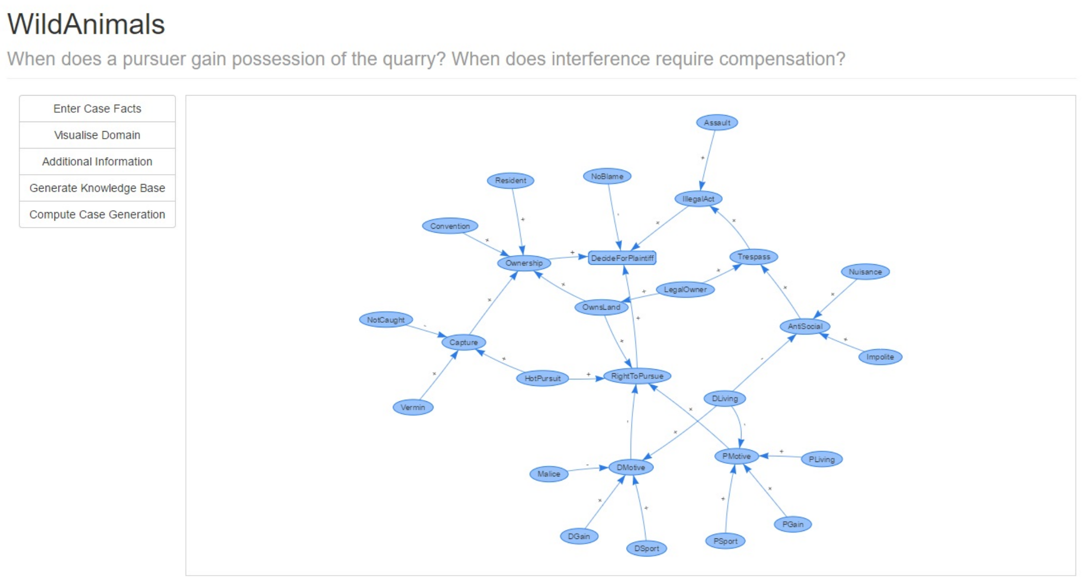
4.3 Information

The database holds a good deal of information relating to various elements such as full descriptions and provenance, which are too lengthy and detailed to be conveniently displayed as part of the visualisation. Obviously we want this information to be readily accessible, and this tool facilitates querying the database to obtain this information for particular items in the database when it is desired.

This tool will also provide links to externally held information which can give access to the original sources and commentaries on them. The cases may be in the public domain (as our the cases in our example domain), or they may be obtainable on a commercial basis from a firm of case law suppliers, or they may form part of the internal documents held by the client for the software under

[6]http://visjs.org/.

[7]http://getbootstrap.com/


WildAnimals

When does a pursuer gain possession of the quarry? When does interference require compensation?

Figure 2: GUI with Wild Animals Domain

Table 1: Test for DecideForPlaintiff

TestID	Head	Priority	TestResult
T1	N1	2	R
T2	N1	3	A
T3	N1	1	A
T4	N1	4	R

Table 2: Body Nodes for Test for DecideForPlaintiff

BodyID	Test	BodyNode	Value
B1	T1	N2	A
B2	T2	N3	A
B3	T2	N4	A
B4	T3	N5	A
B5	T4		

development.

4.4 Knowledge Base Generation

The form of the tests facilitates the generation of a knowledge base from acceptance conditions for the nodes. This enables the acceptance conditions to be tested, evaluated against a set of data and refined as necessary. The idea is to use the tests to construct a Prolog Program, where each node which is not a base level factor has its own Prolog procedure (set of clauses). The base level factors form the facts for a given case.

We will illustrate this using the acceptance conditions for the Verdict Node, Decide For Plaintiff. The expression to be resolved, as given in [3] is:

```
DecideForPlaintiff iff NOT (NoBlame)
                   AND (Ownership OR
                        (RightToPursue AND
                         IllegalAct).
```

This becomes three tests, and a default of *reject*:

```
reject if NoBlame.
accept if Ownership.
accept if RightToPursue AND IllegalAct.
reject.
```

Thus we will have four test records relating to DecideForPlaintiff as shown in Table 1. T1 and T3 have one body node, T2 has two body nodes, T4, the default has 0 body nodes. The relevant BodyTerm records are shown in Table 2.

We now express this information as Prolog clauses using the following template:

```
headnode(boolean):- bodyNode(boolean2), ...
                    body node(boolean3).
```

We replace *boolean* by “t” if the test result is A and by “f” if the test result is R. If the body node has Value = A, the relevant boolean is “t”, else it is “f”. Each test generates a clause. In the case of the default, we have no body nodes. The clauses are ordered according to the priority.

Thus for DecideForPlaintiff

```
decideForPlaintiff(f):- noBlame(t).
decideForPlaintiff(t):- ownership(t)
decideForPlaintiff(t):- rightToPursue(t),
                        illegalAct(t).
decideForPlaintiff(f).
```

The complete Prolog program will have a procedure for each of the non-base level nodes.

To determine the base level factors present in a particular case, for each base level factor in the nodes table, use the dimension associated with that base level factor in the node table to find the corresponding dimension point in case facts, and consider the value. If it is greater than or equal to *threshold*, the base level factor is a fact for the case, otherwise it is not. The set of qualifying base factors are added to the rules, and the resulting Prolog program can be executed using a standard Prolog interpreter.

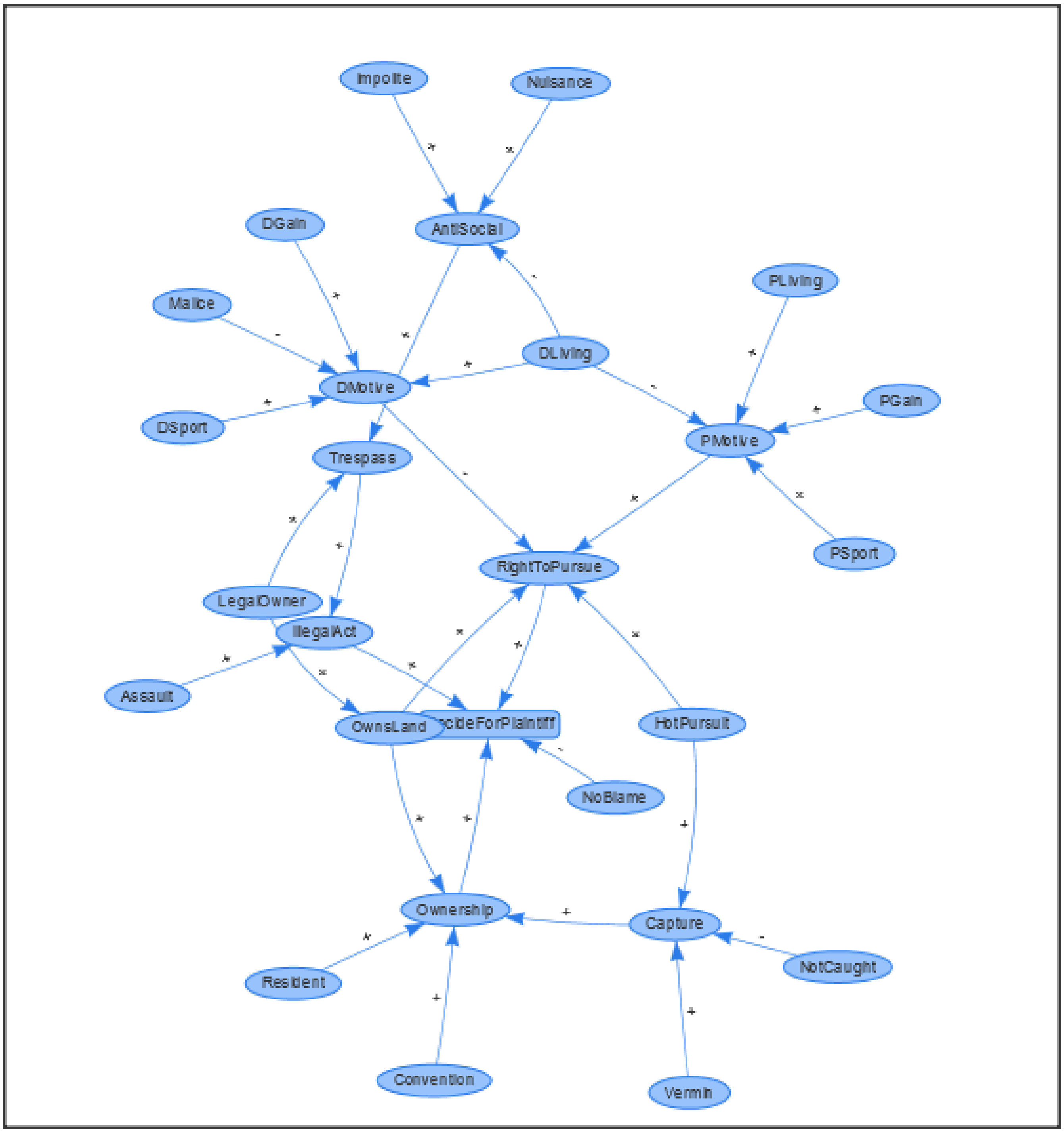
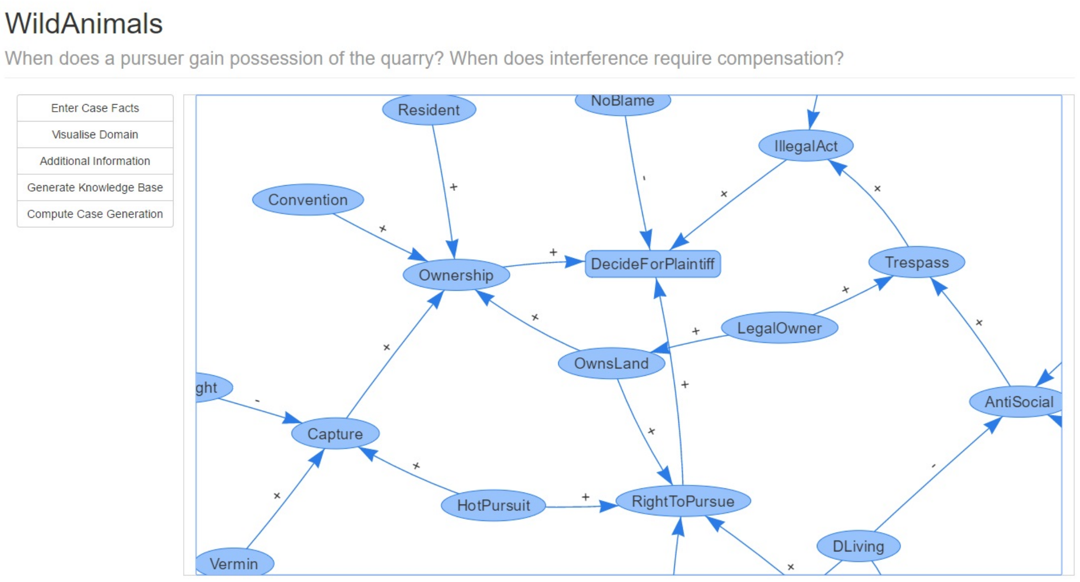
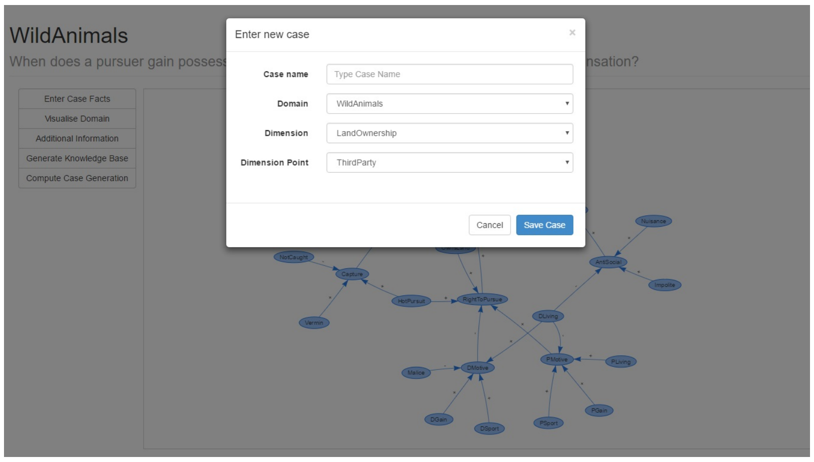


Figure 3: Visualisation of Wild Animals Domain


WildAnimals

When does a pursuer gain possession of the quarry? When does interference require compensation?

Figure 4: Zoomed-in visualisation of Wild Animals Domain


WildAnimals

When does a pursuer gain posses… nsation?

Enter new case

Case name: Type Case Name

Domain: WildAnimals

Dimension: LandOwnership

Dimension Point: ThirdParty

Figure 5: Case Entry Tool

Note that this will output a simple "true" or "false". To add explanation facilities we extend the template by adding
`write(headnode(boolean))`
to the end of every clause.

The base level factors present in a case determine the outcome for that case by supplying the set of facts for the Prolog program.

4.5 Computation when Using Dimensions

In order to allow computation with real numbers taken from the dimension points as suggested in [2] and as prototyped in [14] we proceed as follows. Having chosen a case,

1. All the base level factors are assigned the value corresponding to the dimension point in case facts for their dimension.
2. The test record with priority 0 for a node contains as test result a formula corresponding to the fuzzy logic expression of the acceptance conditions for that node. For example, for DecideForPlaintiff this will be:

   ```
   MIN(NoBlame,
       (MAX(Ownership),
           MIN(RightToPurpsue,IllegalAct)))
   ```

3. These computations are then arranged so that the program works up the tree, so that all the children have their value computed before the parent is called. Because our structure contains no cycles, this is always possible.

5. EXAMPLE APPLICATIONS

In order to test and evaluate the facilities we populated the database with information from the property law related to wild animals and from the automobile exception to the Fourth Amendment domains, both taken from [3]. Since that paper did not discuss dimensions, we drew our dimensions and dimension points for the wild animals domain from [13]. The previous screen shots shown in this paper all relate to this wild animals domain.

5.1 The Wild Animals Domain

These cases were introduced into AI and Law by [16], and are apparently commonly used in US Property law classes. which means they feature in many student work books. That paper used three cases:

- *Pierson v Post.* This was a New York case of 1805. Briefly, Post, a local landowner, was chasing a fox in the traditional manner using horse and hound on open land. As he was closing in on the fox, the fox went to ground and a local tenant farmer, Pierson, beat it to death with a fence pole. Post, annoyed that his sport had been ruined, sued. The case was found for Pierson, on the grounds that possession of the fox should only be granted once the pursuer had physical possession of the animal. The motivation for this decision was mainly that it provided a "bright line" and so gave legal clarity. Post's counsel, Livingston, had argued that because the fox was vermin and a menace to farmers, encouragement should be given to hunters, but the majority were not persuaded.
- *Keeble v Hickergill*, This was an English case dating back to 1707. Keeble earned his living by shooting ducks and selling them. He had rented some land with a pond which was frequented by ducks specifically to pursue his livelihood. His neighbour, Hickergill, had a grudge against Keeble and fired guns (on his own land, so that there was no trespass) to scare the ducks away. The court found for Keeble.
- *Young v Hitchens* was another English case. Both Young and Hitchens were fisherman, fishing for pilchards. Young had spread his net and was drawing them in. This process concentrates the fish into a small area, but takes some time. Hitchens sped into the gap before the nets were closed and was able to make off with the fish so collected. Young sued, but lost, on the grounds that the two litigants were in competition and the court did not feel able to rule as to what constituted unfair competition,

These cases have been extended by other related cases in later papers such as [9]. One much used additional case is:

- *Ghen v Rich.* This Massachusetts case of 1881 involved a whale hunter, Ghen, who harpooned a whale with a bomb lance and killed it, but was unable to secure it. Rich found the whale and sold it on. In whaling the convention is that "the iron holds the whale", and was the property of Ghen, Rich being entitled only to a finder's fee.

In [30] the case of *Popov v Hayashi* was added to these cases.This was a 2002 California case which concerned a baseball hit to secure the record number of home runs by Barry Bonds. Such baseballs have a high value, and because Bonds was a left handed pull hitter a large number of fans had positioned themselves in the right field bleachers in the specific hope of securing the record breaking ball. Popov attempted to catch it, but was frustrated by a jostling mob. The ball ran free and was picked up by Hayashi (who had played no role in the assault on Popov). In this case, because Popov had not been allowed the opportunity to complete his catch through an illegal act, but Hayashi was guilty of no illegality, the judge ruled that the ball be sold and the proceeds divided between them. This case is relevant to the animals cases since they were cited during the trial, in order to provide arguments about whether Popov should be deemed to have possessed the ball.

As mentioned above these cases are widely discussed and can be seen as a *de facto* text bed for the representation of arguments based on cases: e.g. [9], [15] and a special issue of *AI and Law* [8].

For a proper evaluation we will need to use the environment in an analysis of a variety of new and untried domains. We are in a good position to do this since we have on-going engagement with several different law firms which will give the opportunity to exercise these tools in practice[8]. On completion, however, we would hope to be allowed to report at least some of these results. This will alo provide a driver for the identification and development of additional tools.

5.2 The Automobile Exception Domain

The database for this domain is based on the ADF presented in Figure 6 of [3]. Under the Fourth Amendment, a warrant is normally needed to conduct a search of a suspect or his possessions. There are, however, several exceptions, and a series of Supreme Court cases relate to an exception for automobiles (see [10]). Essentially the issues concern finding a balance between exigency, which relates to the practical issue of whether a warrant can be obtained, or whether the evidence would simply be driven away, and privacy: expectations of privacy are considered lowered in the case of an automobile and raised in the case of a dwelling. A case which has widely been discussed in AI and Law since its introduction in [27] and its reintroduction in [7] is *California v Carney*. That case related to the search of a mobile home, which was capable of use either as a vehicle or a dwelling. The majority found for California,

[8]Unfortunately reasons of confidentiality and the constraints of blind review do not allow us to give any details relating to these activities here

because it was in use as a vehicle at the time of the search, being in a public, short stay, car park. The minority opinion argued, however, that because it was daytime in downtown San Diego, a warrant could have been obtained without risk of losing evidence. The visualisation of this domain is shown in Figure 6.

6. CONCLUDING REMARKS

In order for research ideas in AI and Law to make an impact on the legal profession, it is necessary that there is some confidence that usable applications will be delivered. Such confidence is greatly increased by a well founded methodology and a support environment, since that provides some reassurance that the problem will be tackled in a systematic and reproducible manner. While tools to support the use of ADFs do exist (e.g. DIAMOND [21]), these are very much directed at the use of ADFs in an abstract context rather than supporting development of applications in particular legal domains. This gives our tool an analyst/user focus, whereas DIAMOND is very much intended to support research on ADFs.

We believe that the tools we have described in this paper provide a number of advantages for teams following the Angelic methodology.

- The existence of the database can help drive the analysis by indicating exactly what knowledge is needed, what has been collected and what remains to be discovered. This makes the analysis more systematic, objective, transparent and repeatable.

- The information is recorded and stored in a systematic fashion facilitating exchange of information within the team, and reporting to clients.

- The tools, such as the visualisation and information tools, facilitate the exploration and validation of the analysis, presentation of results, and suggest areas for refinement.

- Other tools facilitate the collection, entry, storage and use of test data, and the execution of the test data to identify problems and the need for potential refinements.

We believe that the work described in this paper will provide invaluable assistance in further applications of the Angelic methodology, and that opportunities to extend the tool set will be identified as a result of using the environment. We see the development of such support tools as essential if the current interest in AI that is being shown in the legal profession is to bear real fruit in terms of the widespread adoption of the techniques developed in AI and Law research.

7. REFERENCES

[1] L. Al-Abdulkarim, K. Atkinson, and T. Bench-Capon. Evaluating the use of abstract dialectical frameworks to represent case law. In *Proceedings of the 15th International Conference on Artificial Intelligence and Law*, pages 156–160. ACM, 2015.

[2] L. Al-Abdulkarim, K. Atkinson, and T. Bench-Capon. Angelic secrets: bridging from factors to facts in us trade secrets. In *Proceedings of Jurix*, 2016.

[3] L. Al-Abdulkarim, K. Atkinson, and T. Bench-Capon. A methodology for designing systems to reason with legal cases using abstract dialectical frameworks. *Artificial Intelligence and Law*, 24(1):1–49, 2016.

[4] L. Al-Abdulkarim, K. Atkinson, and T. Bench-Capon. Statement types in legal argument. In *Proceedings of JURIX 2016*, pages 3–12, 2016.

[5] V. Aleven. *Teaching case-based argumentation through a model and examples*. PhD thesis, University of Pittsburgh, 1997.

[6] K. Ashley. *Modelling Legal Argument: Reasoning with Cases and Hypotheticals*. Bradford Books/MIT Press, Cambridge, MA, 1990.

[7] K. D. Ashley, C. Lynch, N. Pinkwart, and V. Aleven. A process model of legal argument with hypotheticals. In *Proceedings of Jurix 2008*, volume 189, pages 1–10, 2008.

[8] K. Atkinson. Introduction to special issue on modelling Popov v. Hayashi. *Artificial Intelligence and Law*, 20(1):1–14, 2012.

[9] T. Bench-Capon. Representation of case law as an argumentation framework. In *Proceedings of Jurix 2002*, pages 103–112, 2002.

[10] T. Bench-Capon. Relating values in a series of Supreme Court decisions. In *Proceedings of Jurix 2011*, pages 13–22. Citeseer, 2011.

[11] T. Bench-Capon. Representing Popov v Hayashi with dimensions and factors. *Artificial Intelligence and Law*, 20(1):15–35, 2012.

[12] T. Bench-Capon, M. Araszkiewicz, K. Ashley, K. Atkinson, F. Bex, F. Borges, D. Bourcier, P. Bourgine, J. G. Conrad, E. Francesconi, et al. A history of ai and law in 50 papers: 25 years of the international conference on ai and law. *Artificial Intelligence and Law*, 20(3):215–319, 2012.

[13] T. Bench-Capon and F. Bex. Cases and stories, dimensions and scripts. In *Proceedings of Jurix 2015*, pages 11–20, 2015.

[14] T. Bench-Capon and T. Gordon. Two tools for prototyping legal cbr. In *Proceedings of Jurix 2015*, pages 177–178, 2015.

[15] T. Bench-Capon and G. Sartor. A model of legal reasoning with cases incorporating theories and values. *Artificial Intelligence*, 150(1-2):97–143, 2003.

[16] D. H. Berman and C. D. Hafner. Representing teleological structure in case-based legal reasoning: the missing link. In *Proceedings of the 4th international conference on Artificial intelligence and law*, pages 50–59. ACM, 1993.

[17] G. Brewka, S. Ellmauthaler, H. Strass, J. P. Wallner, and S. Woltran. Abstract dialectical frameworks revisited. In *Proceedings of the Twenty-Third international joint conference on Artificial Intelligence*, pages 803–809. AAAI Press, 2013.

[18] G. Brewka and S. Woltran. Abstract dialectical frameworks. In *Proceedings of the Twelfth International Conference on Principles of Knowledge Representation and Reasoning*, pages 102–111. AAAI Press, 2010.

[19] S. Bruninghaus and K. D. Ashley. Predicting outcomes of case based legal arguments. In *Proceedings of the 9th international conference on Artificial intelligence and law*, pages 233–242. ACM, 2003.

[20] T. M. Connolly and C. E. Begg. *Database Solutions: A step-by-step guide to building databases*. Pearson Education, 2004.

[21] S. Ellmauthaler and H. Strass. The DIAMOND system for computing with abstract dialectical frameworks. In *Proceedings of COMMA 2014*.

[22] T. F. Gordon and D. Walton. Pierson vs. Post revisited - A

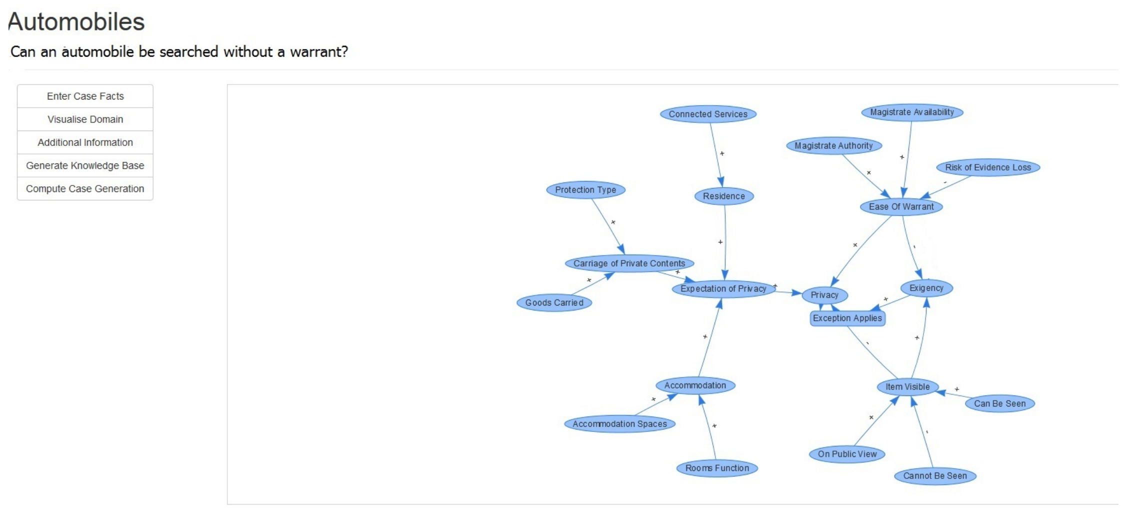


Figure 6: Visualisation of Automobile Exception Domain

reconstruction using the Carneades argumentation framework. In *Proceedings of COMMA 2006*, pages 208–219, 2006.

[23] T. F. Gordon and D. Walton. A Carneades reconstruction of Popov v Hayashi. *Artificial Intelligence and Law*, 20(1):37–56, 2012.

[24] P. Johnson and D. Mead. Legislative knowledge base systems for public administration: some practical issues. In *Proceedings of the 3rd international conference on Artificial intelligence and law*, pages 108–117. ACM, 1991.

[25] H. Prakken. Reconstructing Popov v. Hayashi in a framework for argumentation with structured arguments and Dungean semantics. *Artificial Intelligence and law*, 20(1):57–82, 2012.

[26] H. Prakken, A. Wyner, T. Bench-Capon, and K. Atkinson. A formalization of argumentation schemes for legal case-based reasoning in ASPIC+. *Journal of Logic and Computation*, 25(5):1141–1166, 2015.

[27] E. L. Rissland. Dimension-based analysis of hypotheticals from supreme court oral argument. In *Proceedings of the 2nd international conference on Artificial intelligence and law*, pages 111–120. ACM, 1989.

[28] E. L. Rissland and K. D. Ashley. A note on dimensions and factors. *Artificial Intelligence and law*, 10(1-3):65–77, 2002.

[29] D. B. Skalak and E. L. Rissland. Arguments and cases: An inevitable intertwining. *Artificial intelligence and Law*, 1(1):3–44, 1992.

[30] A. Wyner, T. Bench-Capon, and K. Atkinson. Arguments, values and baseballs: Representation of Popov v. Hayashi. In *JURIX*, volume 165, pages 151–160, 2007.

[31] L. A. Zadeh. Fuzzy sets. *Information and control*, 8(3):338–353, 1965.