Using a Multi-Layered Approach to Representing Tort Law Cases for Case-based Reasoning

Barbara B. Cuthill
National Institute of Standards and Technology
Division 872, Bldg 225, Room B266
Gaithersburg, Maryland 20899
email: bcuthill@swe.ncsl.nist.gov

and

Robert McCartney
Dept. of Computer Science and Engineering
University of Connecticut
Storrs, CT 06269
email: robert@cse.uconn.edu

ABSTRACT

This paper presents a multi-layered case representation for addressing the problem of comparing and indexing cases in a case-based reasoning system using both the facts and the underlying themes associated with those cases. In many domains, experts discuss problems in terms of multiple layers of abstract principles, classifying problems by conflicts, strategies, or themes important to the domain. For example, when lawyers compare cases, they compare the facts, each side's interpretation of the facts, each side's arguments, and disagreements about those arguments. Representing the case as a single flat frame does not support reasoning about the underlying themes in the case because it does not represent interconnections among the facts and themes of the case. Instead a case-based reasoner should use a multi-layered representation including both these facts and interconnections. This representation has implications for much of the case-based reasoning process including case comparison, selection and retrieval mechanisms. The paper will describe the CHASER case-based reasoning system which uses a multi-layered case representation approach to reason about tort law cases.

1.0 Introduction

A *case-based reasoner* compares problems in new situations to selected past experiences or *cases* and uses the results of that comparison to address those problems. Useful past cases are similar to the current situation but the similarity may be surface (in the facts) or deep (based on some underlying theme). Therefore, one problem in case-based reasoning is that of representing and using the significant elements of a case whether they are facts, underlying themes or some combination of the two. Possibly portions of several past cases, rather than entire past cases, may be useful for addressing the problems in the new situation. Exactly which portions or what criteria to use in judging the relative significance of the facts and themes present in a case should depend on the domain theory (Hafner, 1990; McCarty, 1991).

Our solution to this problem is a *multi-layered* case representation. Multi-layered representations represent the facts of the case and several types of abstractions from the facts. An abstraction is any theme or principle the domain theory uses to classify or compare problems. Types of abstractions include interpretations of or viewpoints on the facts, conflicts between interpretations, abstractions of those conflicts and underlying themes.

This representation makes using domain theory to analyze, compare and select cases easier by supporting reasoning about the application of domain theory to specific cases. A multi-layered case representation allows the case-based reasoner to represent portions of a case as examples of several different domain abstractions. The case-based reasoner can isolate the portions of the case used to support a specific abstraction and compare them to portions of another case. The case-based reasoner can reflect the domain theory's partition of the case elements by their abstraction from the facts or their relationship to a particular theme. The case-based reasoner can also use the abstractions classifying the underlying problems in the case as indices for case retrieval allowing it to select related cases based on the underlying problem in the new case.

This paper presents background to the case representation problem (Sec. 2); CHASER (Cuthill, 1992), a case-based reasoning system in the tort law domain which uses a multi-layered case representation (Sec. 3); an example of CHASER's operation (Sec. 4) and conclusions

about the advantages of a multi-layered case representation drawn from this research (Sec. 5).

2.0 Background to the Problem

The research discussed here addresses the problem: How can a case-based reasoner select, compare, and index cases using the significant features of the domain whether they are surface facts or underlying themes of the domain theory. The goal of selecting, comparing, indexing and partitioning cases using underlying themes requires the solution system to use the same abstractions to characterize cases involving different facts if the domain theory does under the same circumstances. In general, the significant features of a case are the problem, the domain theory's characterization of the problem, and the components of the case supporting that characterization. While the representation and use of multiple abstraction levels of knowledge is not new to AI or case-based reasoning (for example, Sacerdoti, 1973; Alterman, 1986; Kolodner, 1987), we are proposing to use a multi-layered representation as a means both to organize and partition the cases and to accurately reflect the accepted structures in the domain theory.

2.1 Tort law domain

While there are many possible domains for this research, we chose the tort law domain because it provides good examples of experts discussing cases in terms of defined abstractions. Torts include a broad class of harmful actions involving compensation for deaths, maimings, and mental anguish from sources such as medical malpractice and product liability. Lawyers use precedent cases to support a desired resolution of a legal dispute drawing parallels between the reasoning used in a previous decision and events in the current case. This use of cases stems from the common law tradition obliging judges to explain rulings that apparently deviate from past application of legal principles.

Our case representation reflects tort law domain theory as found in judicial opinions and legal text books (Prosser, 1988). Lawyers describe tort cases using the *facts*, the *duty* defining the plaintiff and defendant's legal relationship, the *cause-of-action* defining the plaintiff's interpretation of the facts, the *defenses* or the defendant's responses, the *issues* or the conflicts between the interpretations, and the *holdings* or the judge's decisions on issues. The duty, cause-of-action, defense, issue and holding are the *constructs* of tort law.

2.2 Past approaches to case representation

There have been two major approaches to representing legal cases for case-based reasoning: representing the entire case as a single frame and providing a formal representation for all the facts of the case. Each approach has its strengths and we draw on both to arrive at a representation for the structure of legal cases. HYPO (Ashley, 1991; Rissland and Ashley, 1988) is a good example of a single frame representing a case with a selection of linear features or *dimensions*. This representation is convenient for comparing and organizing entire cases. Cases are close to the degree that they match on these features. CABARET (Rissland and Skalak, 1990) has demonstrated the usefulness of this representation in hybrid reasoning.

Alternatively, McCarty (1989) and Branting (1991) emphasize the importance of representing all the facts of the new situation. McCarty (1989) addressed the problem of creating a knowledge representation language allowing common sense categorization and the application of legal reasoning rules. His criteria for a solution are that the language be formal, have a compositional syntax, a precise semantics, and a well-defined inference mechanism. McCarty provides his solution to this problem in his *Language for Legal Discourse* (LLD). He uses variations of the LLD in TAXMAN (McCarty, 1977) and EPS II (Sholbohm and McCarty, 1989). Branting uses a variation of this approach in GREBE representing decisions in legal cases with an *event network* illustrating the *functional dependencies* among facts. This representation provides a structure for comparing clusters of facts within cases.

3.0 CHASER

CHASER demonstrates the usefulness of representing and comparing structures of cases as domain experts would. It uses a multi-layered approach to analyze the facts in a new situation, provide the potential legal cases arising from it and identify the closest precedent cases to the issues present in those cases.

3.1 Overview of CHASER's Functions

CHASER is a system based on a multi-layered case representation (depicted in Figure 1) performing two functions associated with lawyers: issue spotting and precedent selection. Issue spotting is the process of identifying the conflicts between legal arguments a court must address in deciding a legal case. Issue spotting requires the creation of potential legal cases consisting of the facts, components of legal arguments derived by applying legal principles to the facts, legal arguments built from those components, and the legal issues generated by the conflicting legal arguments. CHASER builds each layer of the case representation by applying the domain theory to the facts and the previous layers generating an increasingly abstract picture of the problems addressed in the case.

Precedent selection is the process of identifying the best past cases for supporting a particular resolution of a legal issue. To be a useful precedent a past case must address the same issue as that found in the new case, the judge's reasoning in the past case must be applicable to the new case, and the issue should have resulted from a similar conflict in legal interpretations. Precedent selection can use the multi-layered structure of legal cases to identify closely

related past cases since the closer the legal constructs and facts of the past case are to those of the new case, the stronger the argument based on the past case.

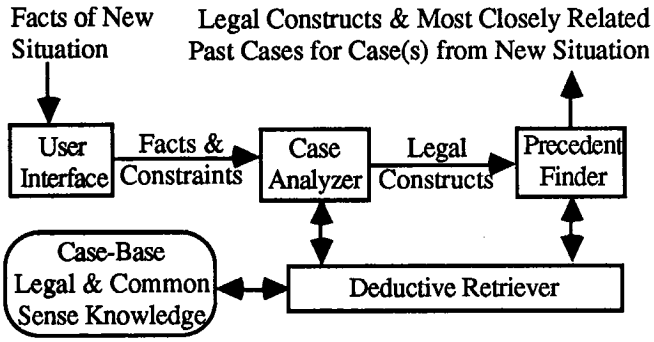

Figure 1: CHASER

CHASER accepts the facts of a new situation in predicate calculus form. The user can specify only those plaintiffs and defendants interesting to him or allow CHASER to generate all possible tort cases from the new situation. The case analyzer accepts the facts and constraints and performs issue spotting providing the legal constructs of the tort cases arising from the new situation. The precedent finder uses the constructs and disputed facts as indices to find the closest past cases addressing the legal issues. The precedent finder can select only the single closest case or continue searching as long as the user continues to request more cases. The case analyzer and the precedent finder use a knowledge base of past legal cases, legal knowledge, and world knowledge available through a deductive retriever. The legal knowledge consists of rules describing the circumstances in which a legal principle applies. CHASER's outputs are the legal constructs for the new cases and the closest past cases for resolving the legal issues and judging the strengths and weaknesses of the arguments provided.

CHASER's current case-base consists of 35 cases. Each case is a multi-layered structure consisting of between 30 and 100 propositions representing the facts and at least one duty, cause-of-action, defense, issue and holding.

3.2 Case Representation

To analyze new situations and find relevant past cases, CHASER uses a case representation allowing it to classify the case's constructs, relate the constructs to supporting facts, classify cases and compare cases or portions of cases. CHASER represents new and past cases as structures with six layers: facts, duty, cause-of-action, defense, issue and holding. Figure 2 illustrates these layers.

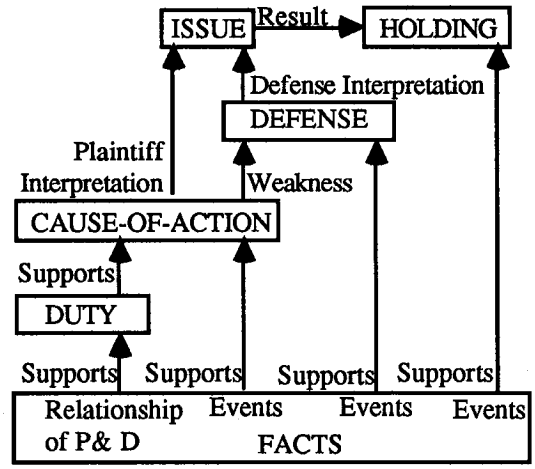

Figure 2: Layers in the Case Representation

Each layer of the case builds on the previous layers and uses the facts of the case to support the application of the legal principle contained in the construct for that layer. The duty construct refers to some of the facts to support applying a specific duty. The cause-of-action construct refers to the duty and the facts to support applying a specific cause-of-action. The defense refers to an element in the cause-of-action and, possibly, the facts to support applying a defense. The issue refers to the challenged element in the cause-of-action and the defense to support the construction of a specific legal issue. The holding refers to the issue and the facts in the case to support a specific conclusion. Each construct partitions the facts into those supporting the application of the construct.

The basis of the case representation is the representation of the facts; facts are also the basis for domain reasoning in the tort law domain. CHASER uses a predicate calculus notation to represent the facts in a situation. The legally relevant items and events include actions, actors, objects, relationships among actors and objects, actors' knowledge, and temporal and causal relationships among actions, states, and state-changes. CHASER represents facts in a language similar to LLD (McCarty, 1989). Like LLD, CHASER treats events and relationships as separate objects to facilitate reasoning about important case elements. Since state-changes such as changes in the health or physical integrity of actors and objects are critical to tort law, these are reified allowing the representation of causal and temporal relationships between events and state-changes.

CHASER represents tort law constructs as frames. Each frame is a general template with type constraints on the slots and can be instantiated for a particular fact situation using legal knowledge encoded as rules. CHASER organizes legal classifications, therefore legal constructs, in a taxonomic structure similar to (Hafner, 1987) and based on accepted jurisprudential classifications (Prosser, 1988). For example, Figure 3 shows part of the cause-of-action classification hierarchy. The hierarchical organization of these frames allows CHASER to easily relate corresponding frames on the basis of their common

taxonomic ancestry. The precedent retrieval process uses this taxonomic information.

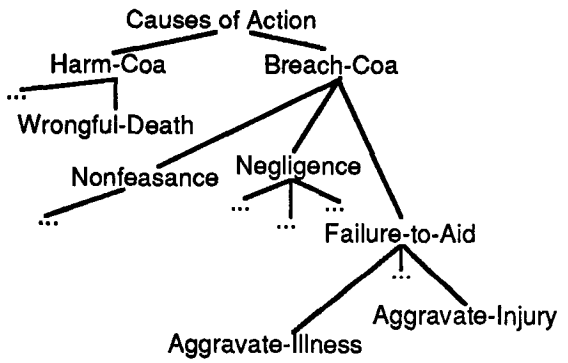

Figure 3: Partial Cause-of-Action Network

CHASER represents the legal knowledge used to generate the frames as rules. We term these *legal interpretation rules*. The rules connect two types of definitions of legal principles, explicit definitions of legal principles represented using propositions and sections of past cases used to support the application of legal principles. Figure 4 illustrates a legal rule defining harm. Explicit definitions clearly indicate a fact or group of facts meeting the definition of a legal principle. Typically, these are common sense definitions. In Figure 4, death is an explicitly defined type of harm. The second type of definition for a legal principles uses the legal principle to index unusual instances of the principle found in the case base. In Figure 4, Haines v. Temple University Hospital (Post, 1986) contains an unusual type of harm, loss of psychic abilities, and the rule for finding harm indexes this case to define this harm. The rule compares only those facts from the past case supporting the specific legal principle or classification to the new case. These definitions reflect the need to find supporting precedents for the application of legal principles in unusual cases but not in typical cases.

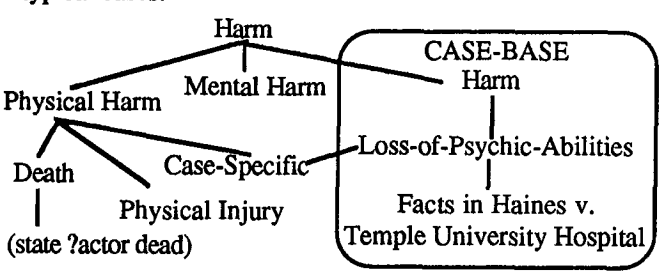

Figure 4: Illustration of Rule Defining Harm

CHASER includes knowledge of 6 types of legal duties, 12 types of causes of action, 10 types of defenses and 8 types of issues. The case-base contains at least one example of each.

3.3 Case Analyzer

CHASER's case analyzer accepts a description of the facts of a new situation with any user constraints on the plaintiff and defendant and provides a description of the potential resulting tort cases. The input description of the facts is in predicate calculus form. The output description contains the potential plaintiffs, defendants, causes of action, duties, defenses, and issues characterizing each possible case derivable from the facts. Figure 5 depicts the output.

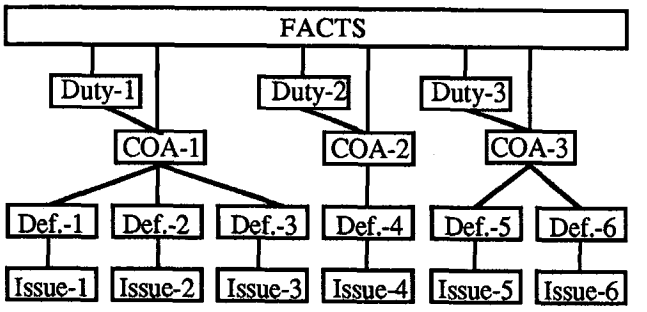

Figure 5: Output of the Case Analyzer

CHASER's case analysis mechanism is consistent with the issue spotting method that Gibbons (1990) describes lawyers as using for new cases. Gibbons' lawyer first identifies who suffered an injury. The lawyer then traces the events causing the injury backward to find a series of potential defendants. Then the lawyer considers how each defendant might respond to the plaintiff and what the likely resulting legal questions or issues are. CHASER mimics this method by applying legal interpretation rules using an algorithm derived from Gibbons (1990) to supply the information to create the legal constructs for the new case. See Figure 6 for the algorithm and the rules applied to fill in the constructs.

CHASER builds a case representation by building and classifying legal constructs at each step of the issue spotting process using the legal interpretation rules. CHASER applies the rules to the facts of the case and the previously generated legal constructs. Each rule compares the facts to explicit definitions of and sections of past cases used as examples of legal principles. Like CABARET (Rissland and Skalak, 1989), CHASER tests explicit definitions first and uses cases when the rules run out (Gardner, 1987). For example, if someone suffered a physical injury, CHASER does not seek unusual examples of harm. However, if no facts meet the explicit definitions, CHASER compares unusual examples of facts supporting a legal principle to the new situation.

The rule guided comparison of past cases and the new situation used in issue spotting considers only those facts in the past case actually cited to support the application of a legal principle. CHASER compares facts by checking if the propositions for the facts contain the same properties or ones that are potentially specializations of each other. CHASER presents the facts of the previous case used to justify the application of a legal principle and the corresponding facts in the current case.

The case analyzer both generates the legal constructs used in the multi-layered case representation and takes advantage of that representation when determining what classifications apply to the constructs in the new case.

Algorithm	Slot/FRAME Filled	Rule Applied
For each potential plaintiff,		
Identify harmed actor	Plaintiff	Common Sen. Harm Def.
Identify harm action	Harm	Causation Def.
Until no more actions or acceptable defendant		
Identify next action leading to harm	Causation	Common Sen. Causation
Id. actor responsible	Defendant	Legally Responsibility
	Breach	Breach Def.
Id. actor responsible for 1st actor	Defendant	Legal Responsibility
Id.duty involved	DUTY	Duty Def.
Is defendant acceptable?		Ask User
For each cause of action, Find defenses challenging		
Duty	DEFENSE	Duty-Defense
Breach	DEFENSE	Breach Defense
Causation	DEFENSE	Causation-Defense
Harm	DEFENSE	Harm-Defense
Facts	DEFENSE	Common-Sense Contradiction
Any examples of similar cases?	DEFENSE	Check Cases
For each defense, identify		
Type of defense	Defense-Int./ ISSUE	
CoA Element challenged	Challenging/ DEFENSE	
Support for CoA element	Plaintiff-Int. ISSUE	
Classify issue	Issue-type	Issue Def.

Figure 6: Algorithm for Analyzing Cases

3.4 Precedent Retrieval

CHASER's precedent retriever selects the closest past cases to a specific issue in the new case using the relative importance of the constructs as assigned in the domain theory. In the tort law domain, the actual issue is the most important element since it is the problem to resolve. The competing arguments giving rise to it are next, since if the lawyers argued the new case and precedent case the same way, the cases are closer examples of the same problem. Similarly, if the causes-of-action refer to the same duty, the match between the arguments is stronger. Finally, if the disputed facts are the same, the cases are very similar. However, if lawyers argued two very similar fact situations differently, the different resulting arguments and issues will prevent one case from being a useful precedent for resolving the issue in the other.

The precedent retriever receives the legal constructs describing the problem. Each construct provides an index into a hierarchy of legal principles. These hierarchies relate the legal principles to each other and organize the past cases. By partitioning the case-base using the most abstract description of the problem, then partitioning the remaining closest matches using each successively more specific classification down to the disputed facts, CHASER partitions the case-base until it arrives at the closest precedent case(s). The last step of the process selects the most recent case from among the equally close cases. If at any point the combination of classifications produces no related past cases, CHASER generalizes the last classification considered using the hierarchies of legal principles and continues partitioning the case-base. Figure 7 illustrates partitioning the case base.

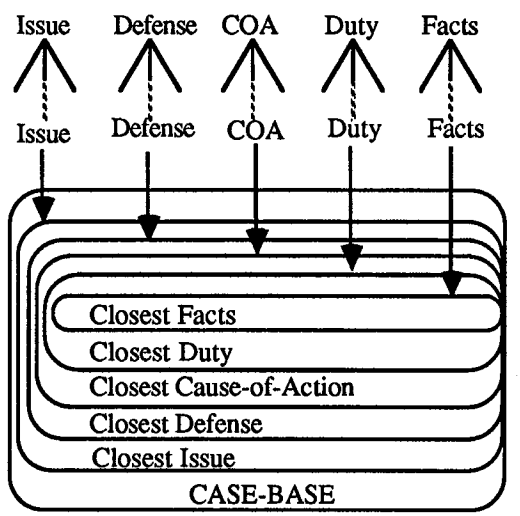

Figure 7: Illustrating Precedent Retrieval

This method preserves the relative importance of the various indices by generalizing the indices describing the details of the problem first thereby preserving the classification of the overall problem. This corresponds to varying the context or surface facts before reclassifying the underlying problem.

After finding the closest precedent case, CHASER can continue searching the case-base at the user's discretion for the next most closely related cases by varying the last index applied to the case-base.

4.0 Example

This section provides an extended example of CHASER processing a new situation. The example case is *Hicks v. L.S. Ayres* (Prosser, 1988). The facts of the case

are that when six-year old Albert Hicks' mother took him to the L.S. Ayres store, he got his hand caught in an escalator which the manager initially refused to stop causing further injury to Albert's hand.

CHASER's case analyzer receives the facts of the case with no constraints on who the plaintiff and defendant should be. CHASER begins identifying the components of the cause-of-action by applying the legal interpretation rule for harm. The definition for *physical injury* includes Albert's injuries making him the potential plaintiff. CHASER then applies legal interpretation rules for the *cause* of the harm, who was responsible (the *defendant)* and the *breach* action or defendant's action causing the harm. CHASER applies common sense causal rules to trace the events back from the injury finding the manager's failure to stop the escalator responsible for aggravating the injury, and the store the potential defendant because it is responsible for the manager's action. Next CHASER applies the legal interpretation rules for identifying the plaintiff and defendant's *relationship* and the applicable *duty* finding Albert and the store in a business host-invitee relationship with the store owing Albert the duty of a business host. Last, the legal interpretation rule classifying the cause-of-action uses the components to find this cause-of-action matches the definition of an *aggravate-injury* cause-of-action.

If the store is a satisfactory defendant, CHASER applies legal interpretation rules defining challenges to each component of the cause-of-action. CHASER finds only one defense the store can use to challenge the cause-of-action. This is the defense the store actually used, that stopping the escalator was not within the scope of store's duty to Albert because no past cases required a store to stop an escalator. After generating the defense layer of the case representation by instantiating the defense frame, CHASER uses the classifications for the cause-of-action and defense to arrive at the classification for the issue. CHASER identifies the challenge to the classification of the defendant's action as a breach of his duty as the underlying dispute. The resulting issue is: What was the extent of the store's duty to Albert Hicks? Figure 8 illustrates the output of the case analyzer.

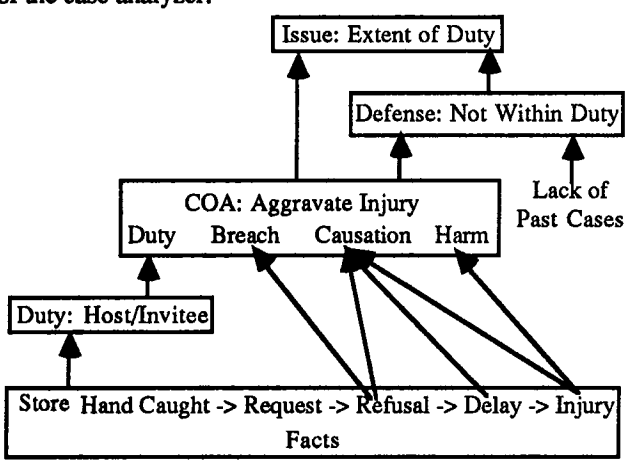

Figure 8: Hicks v. L.S. Ayres Represented

After the precedent finder receives the case representation, CHASER partitions the case-base using the initial classifications provided beginning with the issue. However, the combination of issue, defense and cause-of-action yields no cases. CHASER generalizes the cause-of-action from *aggravate-injury* to *failure-to-aid*. This classification yields several cases; however, further partitioning the case-base using the next layer of the case representation, the duty, yields one case, *Depue v. Flatau*. CHASER returns this case to the user who can ask for the search to continue or stop at that point. Figure 9 illustrates precedent selection in *Hicks v. L.S. Ayres*.

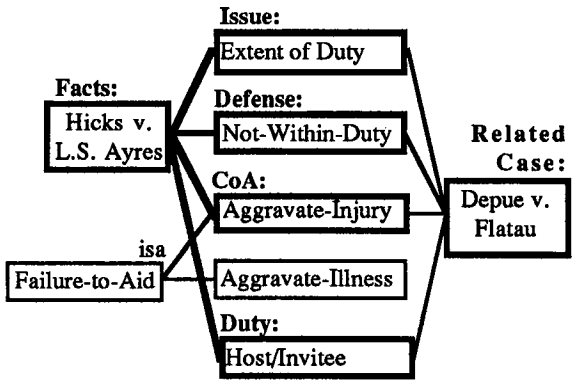

Figure 9: Illustrating Precedent Selection

5.0 Conclusions

This research has explored the use of a multi-layered case representation in a case-based reasoning system. In addition to reflecting how the domain experts discuss cases, a multi-layered representation allows for reasoning about how the domain theory applies to a case. Since the different abstractions classify specific case elements, a multi-layered case representation can isolate the elements supporting a particular abstraction creating a definition of that abstraction. These different classifications for different portions of the case allow for the representation of different viewpoints on the facts. A multi-layered case representation also makes indexing the case-base easier since each layer of information can provide a separate index ordered according to its significance in describing the overall problem or the piece of the problem under consideration. Knowing the importance of the elements of a case and the abstraction hierarchy for each element type can provide a method for systematically retrieving cases in the order of their similarity.

One advantage of such a multi-layered representation is that it allows the case-based reasoner to incorporate the methodology of the domain theory in analyzing, selecting and comparing cases. The domain of this research, tort law, formalizes discussion of abstractions in the domain as discussion of legal principles. Legal experts, not just tort lawyers, tend to use a layered representation of legal cases as their domain model of a case. Our case-based reasoner, CHASER represents and uses the domain model most legal experts use for reasoning about and from cases in the tort law domain.

Future work in this area could generalize the multi-layered case representation to support reasoning about other domains. Any domain incorporating multiple conflicting views of the facts to arrive at the underlying problem is a suitable one for this representation.

References

Alterman, Richard. 1986. An Adaptive Planner. In Proceedings of the American Association for Artificial Intelligence. Menlo Park, Calif.: American Association for Artificial Intelligence.

Ashley, Kevin D. 1991. *Modeling Legal Argument: Reasoning with Cases and Hypotheticals.* Cambridge, MA: MIT Press.

Branting, L. Karl. 1991. Reasoning with Portions of Precedents. In Proceedings of the Third International Conference on AI and Law, 145-154. New York: ACM SIGART.

Cuthill, Barbara B. 1992. Situation Analysis, Precedent Retrieval, and Cross-Context Reminding in Case-Based Reasoning, Technical Report #CSE-TR-92-3 (PhD Diss.). University of Connecticut.

Gardner, Anne. 1987. *An Artificial Intelligence Approach to Legal Reasoning.* Cambridge, MA: MIT Press.

Gibbons, Hugh. 1990. *The Death of Jeffery Stapleton: Exploring the Way Lawyers Think.* Concord, NH: Franklin Pierce Law Center. (Manuscript)

Hafner, Carole. 1987. Conceptual Organization of Case-Law Knowledge Bases. In Proceedings of the First International Conference on AI and Law, 35-42. New York: ACM SIGART.

Hafner, Carole. 1990. An Integrated Model of Deep Structure and Surface Structure in Legal Reasoning. In Workshop Notes of the AAAI Workshop on AI and Legal Reasoning. Menlo Park, Calif.: American Association for Artificial Intelligence.

Kolodner, J., R. Simpson and K. Sycara. 1985. A Process Model of Case-Based Reasoning in Problem Solving. In Proceedings of the Ninth International Joint Conference on Artificial Intelligence. Menlo Park, Calif.: International Joint Conferences on Artificial Intelligence, Inc.

McCarty, L. Thorne. 1977. Reflections on TAXMAN: an Experiment in Artificial Intelligence and Legal Reasoning. *Harvard Law Review* 90. 837-893.

McCarty, L. Thorne. 1989. A Language for Legal Discourse, I: Basic Features. In Proceedings of the Second International Conference on AI and Law, 180-189. New York: ACM SIGART.

McCarty, L. Thorne. 1990. AI and Law: How to Get There from Here. In Workshop Notes of the AAAI Workshop on AI and Law. Menlo Park, Calif.: American Association for Artificial Intelligence.

Post staff. 1986. "Judge Rejects Damage Award to Psychic" in *Washington Post*, Aug. 9, p. A7.

Prosser, William L., John W. Wade, and Victor E. Schwartz. 1988. *Cases and Materials on Torts.* Westbury, NY: Foundation Press, Inc..

Rissland, Edwina L. and Kevin D. Ashley. 1988. Credit Assignment and the Problem of Competing Factors in Case-Based Reasoning. In Proceedings of the Case-Based Reasoning Workshop, 327-344. Defense Advanced Research Projects Agency.

Rissland, Edwina L. and David B. Skalak. 1989. Combining Case-Based Reasoning and Rule-Based Reasoning: a Heuristic Approach. In Proceedings of the Eleventh International Joint Conference on Artificial Intelligence, 524-530. Menlo Park, Calif.: International Joint Conferences on Artificial Intelligence, Inc.

Sacerdoti, E.D. 1973. Planning in a Hierarchy of Abstraction Spaces. In Proceedings of the Third International Joint Conference on Artificial Intelligence. Menlo Park, Calif.: American Association for Artificial Intelligence.

Sholbohm, David and L. Thorne McCarty. 1989. EPS IIA: Estate Planning with Prototypes. In Proceedings of the Second International Conference on AI and Law, 1-10. New York: ACM SIGART.