# THE REPRESENTATION OF AN EVOLVING SYSTEM OF LEGAL CONCEPTS

## II. PHOTOTYPES AND DEFORMATIONS

L Thorne MeCarty
Faculty of Law and Jurisprudence
Stata University of New York at Buffalo

N. S. Sridharan
Department of Computer Science
Rutgars University

ABSTRACT

One of tha principal goals of tha TAXMAN project is to develop a theory about the structure and dynamics of legal concapts, using corporate tax law as an experimental problem domain In this paper we describe the "prototype* plus-deformation" model of legal conceptual structure a concept is represented here by a prototypical description plus a sequence of deformations of the prototype, where the deformations are selected from among the possible *"mappmga"* of one concrete description into another The *paper* focuses on the set of mappings, which is the most important component of the model because it makes manifest the basic coherence of the conceptual space The syntax and semantics of the mappings *are* described, *and* their role in the process of legal argument is suggested The formal modal is then illustrated by examples drawn from Eisner v Macomber, an early stock dividend case

## I INTRODUCTION

In an earlier paper on the TAXMAN project [I], we outlined our basic representation of the corporate tax domain [2], using a *frame-baaed* language [3] with extended facilities for the description of states, events, actions and expectations [4] A principal feature of this representation is its abstraction/expansion hierarchy a relatively abstract" conceptual structure is given *an* "expanded" representation in terms of *an* assemblage of mora concrete conceptual structures, *and* a recursive pattern-matching procedure is used to "generate" the expansions when given the abstractions, *and* to "recognize" the abstractions when given the expansions We have referred to this model of a conceptual hierarchy as a *logical template* model, and we have demonstrated that it is adequate for representing tha facts of *an* actual corporate tax case, such *at* United States v. Phettis, 257 US 156 (1921), *and* adequate also for representing the statutory rules *and* concepts which classify such cases *aa* tax-free reorganizations under Sections 368(a)(1)(B). (C) and (D) of the Internal Revenue Code [2] [I] [5] m addition, we have concluded that tha model is adequate for certain practical applications in tha legal field, as long *as* tha domain of application *ia* carefully chosen [6]

Lawyers would object, however, that the logical template modal is theoretically inadequate as a representation of a ayatam of legal concepts. First, a lawyer might say, tha moat important lagai concepts *era* "open-textured" (see [7], *pp.* 121-50) their etructure is *never* logical and precise, but amorphous and poorly defined Second, legal concepts are not static structures, but dynamic ones (see [8]): they are typically constructed and reconstructed as they *era* applied to a particular set of facts. In this paper we will describe a representation of a system of legal concepts which attempts to capture some of these characteristics. In place of the logical template model, we will represent an abstract concept by a *prototype* and a sequence of *deformations* of the prototype the prototype is *a* relatively concrete description selected from the lower-level factual network itself, and the deformations *are* selected from among the possible *mappings* of one concrete description into another We have suggested in our *earlier* papers that these "prototype-plus-deformation" structures play a crucial role *in* the proceas of legal argument, and that they contribute a degree of stability *and* flexibility to an emerging system of concapts that would not exist with the logical template structures alone [9] [1]

The idea of representing an abstract concept by a prototype and a sequence of deformations of the prototype has numerous antecedents in the literature In the philosophical literature, the idea is suggested in the "family resemblances' of Wittgenstein [10], the "paradigms' of Kuhn [11] [12], and the recent work by several authors on the logic of "natural kinds' [13]. In cognitive psychology, the idea has emerged recently in the experimental work of Rosch *and* others [14] [15] on the coding of *natural* categories There nave been several attempts to formalize these notions in the artificial intelligence literature: for example, the suggestion by Minsk y [16] that a collection of "frames'* could be connected by pairwiae "difference descriptions' into a similarity network'; or the work of Winston on tha construction of "transfer frames" [17] and the understanding of analogies [18] However, since the notion of a "prototype" has generated a great deal of controversy and confusion in the recent literature (sea, eg, [19], it would *appear* that the prototype-plua-deformation model, taken as a whole, is not yat fully understood

In this *paper* we will focus our attention on the *mappings in* the apace of descriptions, since this is the element of the prototype-plus-deformation modal which gives the conceptual representation its coherence. In Section II. wa will outline the basic formalism and introduce two important devices the MAP, which *enpreeeea* a relationship between two concrete descriptions; and the

XDN, which defines a general transformation on the space of concrete descriptions. In Section III, we will illustrate these formalisms with some examples drawn from an early corporate tax case, and we will suggest the ways in which the MAPs and the XDNs play a role in the process of legal argument.

## II THE MODEL: A STRUCTURED SPACE OF DESCRIPTIONS

We have described the basic representational mechanisms of the TAXMAN II system in our previous *papers* [I] [5], but it may be helpful to review the essential points here. To set up a domain of discourse in TAXMAN II (see generally [3]K we first construct a system of *templates* to describe the classes of objects in the domain, and a system of *relations* to express the possible relationships between these objects To describe the facts of a particular case, we generate *instances* of the templates and their associated relations in a particular *context* The main work of the TAXMAN II system is then performed by a collection of pattern-matching procedures which operate on *descriptions,* or, as they *are* frequently called, DDNs A DDN has three components: a template - list a constraint-list and a bindings-Mat, where

```
<template-list>
          ::= ((tname tvar
                    (rname svar)
                    ...          )
               ...                )

  and tvar ::= <variable-name>
      svar ::= <variable-name>
          or  (tname tvar
                    (rname svar)
                    ...          )

<constraint-list>
            ::= <logical-expr>
            or  <arithmetical-expr>

<bindings-list>
          ::= ((var (inst
                    [(var inst)
                     ...       ])
                   ...           )
               ...               )
```

For example, the following DDN would represent the ownership of more than 80 percent of the common stock of a corporation, with the corporation in question bound either to "DuPont" or to "StandardOil":

```
(DDN:
 D-80PerCentOwnership
 ((OWN O1
   (OWNER A1)
   (OWNED (SHARE P1
           (QUANTITY N1)
           (SHAREOF (STOCK S1
                           (QUANTITY N2)
                           (COMMON YES)
                           (VOTING YES)
                           (ISSUER C1))
           )))))
 ((GREATERP N1 (TIMES 0.80 N2)))
 ((C1 (DuPont) (StandardOil))))
```

In this example, we have assumed that the domain already contains a definition of the template OWN, with relations owner" and "owned"; the template SHARE, with relations "quantity" and "shareof"; and the template STOCK, with the relations as indicated.

Now suppose we wished to use the preceding DDN. but minus the bindings for the CORPORATION C1, as a representation of the *expansion* of the concept of corporate control", *as* defined in Section 368(c) of the Internal Revenue Code. We would do this by defining a new template called CONTROL, with relations "controller" and "controlled":

```
(TDN: ((CONTROL REL)
       ((CONTROLLER FN) ACTOR)
       ((CONTROLLED FN) CORPORATION)))
```

*and* then linking up the "80PerCentOwnership" description with the following *adst recti on*

```
((CONTROL CON1
          (CONTROLLER A1)
          (CONTROLLED C1)))
```

In other words, "A1 controls C1" if "A1 owns more than 80 percent of the common stock of C1 The example is a simple one, but for a discussion of several more complicated forms of the abstraction/expansion hierarchy, and an analysis of the procedures RMATCH and GMAKE which operate upon them, see [1] and [5]. As an indication of the level of complexity that we can presently handle, consider the following definition of a "B-Reorganization" in Section 368(a)(1)(B) of the Internal Revenue Code

> the acquisition by one corporation, in exchange solely for all or a part of its voting stock (or in exchange solely for all or a part of the voting stock of a corporation which is in control of the acquiring corporation), of stock of another corporation if, immediately after the acquisition, the acquiring corporation has control of such other corporation (whether or not such acquiring corporation had control immediately before the acquisition)

This example, and others like it, *are* discussed in detail in [2]

Even with a substantial elaboration of the

abstraction/expansion pairs, of course, the preceding formalism captures only the 'logical template" version of a conceptual hierarchy To generalize this representation to a system of "prototypes' *and* deformations", we will replace the single DDN expression in the abstraction/expansion hierarchy with a *structured space* of DON expressions The expansion will thus have three components (1.) an "invariant" component, consisting of a single DON. to handle the case of a concept like CONTROL, which *can* be represented with a simple logical template expansion, (2) a set of exemplars", each of which is a DDN. to handle the case of a concept which *cannot* be represented with a simple logical template expansion. and (3) a set of transformations', eech of which is a MAP. to tie the set of exemplars together into a coherent whole. The set of exemplars by itself provides only a disjunctive specification of a conceptual expansion, without giving any indication of how the various exemplars *are* related to one another, but the set of transformations is intended to supplement this disjunctive specification by expressing certain relationships among the *exemplary,* and thus imposing *an* order and a structure on the conceptual space In this way we hope to capture one of the more important characteristics of the prototype-plus-deformation model: the fact that we can designate one or more of the exemplars as the prototype, and then arrange the remaining exemplars in the order of their prototypicality

Obviously, the critical component of this definition is the MAP structure, which was discussed informally in [9] As a first approximation, s MAP has the following form

```
(MAP    <neme>
        <etmeture1> ::= DDN
        <structure2> ::* DON
        invariant>  ::» DDN
        <transformatlon> ::= MAP)
```

and it expresses the fact that the DDN in the etrueturel slot can be *mapped* into the DDN in the structure2 slot, with a specified DDN *as* the invariant *and* a specified MAP as the residual transformation For simplicity, we will assume that the MAP operates only on the template-Ilet component of the DDN, leaving the constraint-Ilet and the bindings -list unchanged With this simplification, then, we could write out a sample mapping as follows

```
(MAP M-ShareOwnership>CashOwnership
     <structure1>
          = ((OWN 01
                  (OWNER A1)
                  (OWNED (SHARE P1))))
     <structure2>
          = ((OWN 01
                  (OWNER A1)
                  (OWNED (CASH P1))))
     <invariant>
          = ((OWN 01
                  (OWNER A1)
                  (OWNED P1)))
     <transformation>
          = (MAP M-Share>Cash
                  ((SHARE P1))
                  ((CASH P1))))
```

which expresses the fairly obvious fact that the "Ownership-of-a-SHARE" can be mapped into the "Ownership-of-a-CASH", by simply mapping SHARE to CASH. For an extensive discussion of mappings of this sort, see [20].

Notice, however, that the MAP formalism, as described so far, adds nothing of substance to the domain: it merely restates the relationships that exist *implicitly* by virtue of the structure of the DDN space and the definition of the pattern-matcher We often refer to the relationship between the invariant slot and the two structure slots *here a$ an implicit vertical mapping,* since it corresponds to the vertical' generalization-specialization *hierarchy* in the space of DDN expressions (Compare here the "version spaces" of Mitchell [2I]) Analogously, we often refer to the relationship in the transformation slot *as an implicit horizontal* mapping In the preceding example, the horizontal mapping from SHARE to CASH reveals no structure of any importance, but there are numerous mappings in the corporate tax domain which depend on a continuous parameter (eg. the quantity of stock outstanding the proportionality of s stockholders ownership interest, the risk or the liquidity of *ar* investment), *and* these other horizontal *mappings* impose a partial order on the DDN space which is orthogonal to the partial order of the generalization/specialization hierarchy It turns out that the implicit mappings of the DDN space can all be *defined* in terms of a small set of primitive syntactic operations — operations which add and delete templates *and* relations; operations which move up and down the AKO hierarchies, operations which mark variables *as* identical or distinct; etc — and these operations can be arranged in a preference ordering which preserves several important aspects of the simplicity *and* the compactness of the DDN *expressions for* details, see [22]

However, these implicit MAPs are not enough In order to construct a useful system of prototypes and deformations, it is necessary to have available a set of *explicit* mappings of the *DON* space, i.e. a set of *mappings* between DDN structures which *are* not *mappabie* by virtue of their syntactic structure alone What does this mean? Aa a first *approximation,* we could simply stipulate that a certain MAP, such as the "Share>Cash" *mapping* illustrated above, had been previously defined in our domain *and was therefore* available whenever it turned out to be useful But we can often *say more* than this Frequently, we can explain a mapping between two dissimilar *DON* structures by pointing to a related *mapping* with a known invariant *and* a known transformation in a different dimension" of the DDN space For example, we might be able to describe the relationship between two events by describing the relationship between the two corresponding sequences of state descriptions Or we might be able to describe the relationship between a "Share' *and* a "Cash by describing the relationship between the rights and the obligations associated with each Sometimes these new mappings will be used to construct invariant relationships in the DDN space, *m* which case we will call them *explicit*

*vertical* mappings, and sometimes they will be used to construct *ordered* transformations in the DDN space, in which case we will call them *explicit horizontal* mappings, by analogy to the implicit mappings discussed previously. The important point, though, is that these new mappings *are* not just syntactic mappings Instead they impose an explicit semantic structure on the space of DDN expressions.

To carry out these constructions, it is clear that we need a general facility for defining new MAPs in terms of old ones, and with the appropriate expressive power. These considerations lead us to the following definitional form:

```
(XDN: <name>
      (arg1 arg2)
      (SCOPE <DDN-pattern> arg1)
      (SCOPE <DDN-pattern> arg2)
      <body> ::= <elementary-transform>
            or  <(XDNAPPLY
                  <xdn>
                  <structure1>
                  <structure2>)
                ...          >)
```

As suggested, the XDN is s *defined transformation* which takes two arguments, argl and erg2, and expresses the mapping between them in terms of the complex of related transformations listed in its body Because the arguments *are* usually complex DDN expressions, the SCOPE specification is provided to decompose them into their component parts; the given DDN-patterns are matched to the arguments, first to determine the potential applicability of the transformation itself, but also to bind the internal variables of the XDN to the components of the DDN *expressions,* in whatever detail is needed The body of the XDN can then take several forms There would be certain *elementery* transformations to provide the building blocks for the entire system: for example, one important elementary transformation would represent the implicit syntactic mapping described earlier, in which the inverient is a matching DDN expression and the transformation is a residual MAP, another set of elementary transformations would represent the arithmetical relationships in the domain. *and* so on The compound transformations would then be written as the result of *applying* an XDN to the components of argl and arg2 in this case, structure! and struoture2 refer to the sub-structures of the DDN-patterns, *as* extracted by means of the SCOPE specification A number of variations of XDN APPLY *are* possible, but the most important variant for our present purposes is the transformation defined in a related "dimension" of the DDN space We represent this transformation by filling in the structure slots with an expression of the form

```
(PROJECTION  <Projected-Dimension
                  Of DDN-Space>
             arg)
```

which, by assumption, produces the *projection* of the arguments in the specified DDN space In this case, the XDN expresses the following fact argl and erg2 can be *mapped* together because the *projections* of argl and erg2 in the specified DDN space can be *mapped* together under the application of a known *xdn.*

We will provide illustrations of these constructions in the following section of this paper. At the same time, we will suggest how the XDN formalism and the MAP formalism play a role in the construction and modification of legal concepts. Since the XDN is basically a lambda-expression, and since the MAP expresses the result of applying an XDN to a particular pair of DDN expressions, we can explain some of the procedures for conceptual construction in the prototype-plus-deformation model in terms of a directed interaction between lambda-abstraction and lambda-application. We will not develop this point in the present paper, but there will be hints of it in the discussion of the corporate tax case which follows

## III ILLUSTRATIONS: THE MAPPINGS OF *EISNER V. MACOMBER*

We will draw our illustrations from Eisner v Macomber. 252 US 189 (1920), one of the landmark decisions of the early corporate tax code The case involved the distribution of a 50 percent stock dividend Myrtle H Macomber, who originally owned 2200 shares of the common stock of Standard Oil, received a dividend consisting of 1100 additional shares of the same stock. *and* the Collector of Internal Revenue, as required by the Revenue Act of 1916. imposed a tax on the distribution The Supreme Court, however, in an opinion by Justice Pitney, held that a stock dividend of this sort was not income" within the meaning of the Sixteenth Amendment to the Constitution, and thus the tax could not constitutionally be imposed Justice Brandeis, in an elaborate opinion, dissented Several other cases were discussed as precedents in the course of the argument The distribution of a corporations cash, as in Lynch v Hornby, 247 US. 339 (1918), and the distribution of the stock of an unrelated corporation, as in Peabody v. Eisner, 247 US. 347 (1918), were situations that all parties *agreed* should be taxable On the other hand, the appreciation in the value of a stock without the actual transfer of stock certificates, a purely hypothetical case, was a situation that all parties agreed should be nontaxable The task for Justice Pitney and Justice Brandeis. then, as outlined in [9], was to construct a concept of taxable income consistent with these agreed precedents, and in such a way that the Macomber case itself would be classified as nontaxable (Pitney) or taxable (Brandeis)

We have discussed the details of the Macomber arguments in an earlier paper [23]. and we will not repeat them here. However, an important step in the analysis, for both parties, was the construction of a sequence of mappings between the decided and the undecided cases, and these mappings provide an interesting illustration of the formalisms set forth in the previous section of this *paper* For example, one of the more important points in Justice Pitney"s argument was the construction of a mapping between the Macomber case and the unrealized appreciation case: the taxpayer in Macomber now owns 3300 shares of common stock out of 75,000,000 outstanding, the argument goes, but that's the same as owning 2200 shares out of 50.000,000 outstanding, which is the situation that would have existed had there been no actual transfer of stock certificates As Justice Pitney puts it in his opinion, 252 US 189. 211 (1920). the stock dividend "does not

alter the pre-existing proportionate interest of any stockholder" The point is a simple one. but how would we formalize it?

First let *us* assume that there exists the following XDM in our domain:

```
(XDN: X-EndStateMapping
      (d1 d2)
      (SCOPE ((STATE TA varA)
              (STATE TB varB))
             d1)
      (SCOPE ((STATE TX varX)
              <segment>
              (STATE TY varY))
             d2)
      (BODY (XDNAPPLY xdn1 varA varX)
            (XDNAPPLY xdn2 varB varY)))
```

which expresses the fact that an ordered pair of state descriptions can be mapped into an indefinite sequence of state descriptions if the initial state descriptions. verA and varX, can be *mapped* together by some known mapping xdnt *and* the final state descriptions, varfi *and* varY. can be mapped together by some known mapping xdn2 Although this mapping might not be considered *an* invariant mapping in all contexts, in the corporate tax domain there *ara* strong reasons for treating it as such two sequences of transactions which reach the same *and* result ought not to be treated differently just because one of them follows a different path Let us therefore attempt to apply the "EndStateMepping" to our problem, binding a description of the unrealized appreciation case to dl and a description of the Macomber case to *dZ* Because of the definition of the unrealized appreciation case, the pattern-match inside the first SCOPE specification will bind the identical state description to varA and varB: 2200 shares out of 50,000,000 However, the pattern-match of the Macomber case, *expanded* backwards in time to a point just before the issuance of the new stock, will bind this same state description to varX. and s different state description to verY: 3300 shares out of 75.000.000 It follows, then. that the "EndSteteMapping" can be fully applied if xdnl is the identity transformation, and if xdn2 is a transformation of the following sort

```
(XDN: X-ConstantStockRatio
      (d3 d4)
      (SCOPE (<OWNership
                 of NQ1 SHAREs
                      of STOCK S1,
                   NS1 SHAREs
                      outstanding>)
             d3)
      (SCOPE (<OWNership
                 of NQ2 SHAREs
                      of STOCK S1,
                   NS2 SHAREs
                      outstanding>)
             d4)
      (BODY (EQUAL (QUOTIENT NQ1 NS1)
                   (QUOTIENT NQ2 NS2))))
```

Now it is conceivable that the XonstantStockRatio" mapping has already been defined in our domain, since it is an invariant transformation of broad applicability, *and* if this is the case then our analysis is done

However, the force of this mapping can be strengthened substantially if we project it onto s more detailed representation of the space of security interests The ownership of common stock carries with it certain rights to the "earnings", the assets" and the "control" of *a* corporation, and these rights can be represented by s system of *permission* and *obligation* schemata [91 For simplicity, let us focus our attention on the right to earnings: we represent this with a pair of descriptions expressing the fact that when s certain *situation* is satisfied the corporation is *permitted* to carry out the *action* of distributing earnings *prorata* to the common stockholders Suppose we now ask what earnings would be distributed to the owners of the shares in the structural and the structure2 slots of the XonstantStockRatio mapping above It is not too difficult to construct *a* procedure which gives us the answer we trace along the path from the permission structure, to the embedded distribution, and then to the expansion of the distribution in terms of a sequence of transfers to each stockholder The result simplified slightly, *is*

```
(DDN:
 D-PermittedTransferOfEarnings
 ((TRANS TR1
         (AGENT (CORPORATION C1))
         (OBJECT
          (CASH P2
                (QUANTITY (TIMES
                            <fraction>
                            <earnings>))
          ))
         (OLDOWNER (CORPORATION C1))
         (NEWOWNER A1))

  (OWN O1
   (OWNER A1)
   (OWNED (SHARE P1
           (QUANTITY NQ1)
           (SHAREOF (STOCK S1
                     (QUANTITY NS1)
                     (COMMON YES)
                     (ISSUER C1)))))))
 ((EQUAL <fraction> (QUOTIENT NQ1 NS1)))
 NIL)
```

which gives us the "projection' of the original' description onto the space of "permitted transfers of earnings'. But notice now that this pro/ection is identical for both the struoturel *and* the structure2 slots of the XonstantStockRatio" mapping, and thus the mapping has been shown to preserve an *invariant* in the security interest apace. Furthermore, the invariant works only for common stock, not for preferred stock, since the

distribution to preferred stockholders is normally stated in terms of the number of shares owned, instead of the fraction of ownership. Having discovered this invariant mapping, of course, we can now use it to construct an explicit invariant XDN in our domain, following the notation in the previous section of this *paper.*

A more difficult mapping to represent is the one constructed by Justice Brandeis to demonstrate that it is impossible to draw a coherent line of distinction between the stock distribution of Eisner v. Macomber and the cash distribution of Lynch v. Hornby. In his argument Justice Brandeis posits a sequence of hypothetical cases first the distribution of common stock, then *preferred* stock, then bonds; then the distribution of long-term notes, then short-term notes; *and* finally the distribution of cash. This is basically an elaboration of the "Share>Cash" *mapping* introduced in the previous section of this *paper,* but for the mapping to have any force it must be projected onto a detailed representation of the security interest space. For simplicity, let us concentrate on a portion of the mapping: from common stock, to preferred stock, to bonds A direct projection of these securities onto the space of permissions and obligations, which worked well in our previous example, fails to give us the result we want in this case, since the permission *and* obligation structures *are* not directly mappable. The common stockholders receive a permitted distribution of residual earnings, but conditional on a prior permitted distribution to the preferred stockholders, while the bondholders receive *an* obligatory distribution of a fixed amount, and there *appears* to be no way to compare these various structures except by listing all of their pairwise differences. However, a further projection, onto a space of *expectations,* reveals both a common pattern and a systematic difference

To see this, let us analyze how the corporation is expected to behave, given a capital structure with common stock, preferred stock and bonds outstanding The basic behavioral assumption is that the corporation seeks to maximize the benefits to its common stockholders, since the common stockholders hold the ultimate voting power over the corporations affairs In the case of the distribution of earnings, however, there *are,* by virtue of the system of permission and obligation schemata, two groups of claimants standing ahead of the common stockholders the bondholders and the preferred stockholders A relatively straightforward exercise in plan generation (see, e.g. [4]) will therefore demonstrate that the expected cash dividend for each class of securityholders depends on the earnings available for distribution: unless the corporation is near bankruptcy, the bondholders will receive a fixed amount; beyond that, if the available earnings exceed the bond obligation, then the preferred stockholders will receive a dividend, but only up to the limit of the preferred stock obligation; and beyond that if earnings are still available, then the common stockholders will receive the rest We can summarize this analysis with the following *expectation* structure

```
(DDN:
 D-ExpectedDistributionOfEarnings
 ((DISTRIBUTE D1
   (AGENT (CORPORATION C1))
   (OBJECT (CASH P1
                (QUANTITY (<function>
                           <earnings>))
           ))
   (OLDOWNER (CORPORATION C1))
   (RECIPIENT (SECURITY S1))))
 NIL
 NIL)
```

in which the function applied to the earnings depends in a systematic way on the identity of the "recipient security" But now, if we work with this projection of a security interest onto the apace of expectation schemata, instead of the projection onto the apace of permission and obligation schemata, we can define a mapping of the following sort

```
(XDN: X-Equity>Debt
      (d1 d2)
      (SCOPE ((STOCK S1
                     (COMMON YES)))
             d1)
      (SCOPE ((STOCK S1
                     (PREFERRED YES)))
             d2)

      (BODY (XDNAPPLY
             <xdn>
             (PROJECTION <Expected
                          Distribution
                          Of Earnings>
                         d1)
             (PROJECTION <Expected
                          Distribution
                          Of Earnings>
                         d2))))
```

where the xdn embedded inside the BODY is an elementary syntactic transformation with the DISTRIBUTION of CASH as the invariant and the mapping from (<function:COMMON> <earnings>) to (<function:PREFERRED> <earnings>) as the residual transformation The "Equity>Debt" mapping thus expresses the fact that the only difference between common stock and preferred stock in the space of expectation structures is the difference between the two functions which translate earnings into expected quantities of CASH. Another mapping is possible if we assume a reasonable probability distribution on the space of future earnings The quantities of CASH would then inherit a probability distribution, too. and it would be apparent that the common stock has a high expected return and a high riak, while the *preferre* stock has a lower expected return *and* a lower risk This is the traditional way of describing the mapping between "Equity" and "Debt. For a discussion of the importance of this mapping in the arguments of Eisner v. Macomber, and its importance in the later corporate tax cases, see [9] and [23].

## IV CONCLUSION

Although this *paper* has emphasized the differences between the "logical template" model of conceptual structure and the "prototype-pkis-deformetion* model, it is clear that both representations coexist in the corporate tax domain The patterns of reasoning observed in Eisner v Macomber persist throughout the later cases (see [24]) There *are* strong pressures to formulate legal concepts as logical templates, if possible, but there are also strong countervailing tendencies to construct mappings from prototypes, and the tension between these two modes of representation provides the foundation for some of the major strategies of legal argument We believe that the key to understanding this process is a further exploration of the structure of MAPs a*nd* XDNs. as set forth in the present *paper*

Our implementation of the TAXMAN It theory has proceeded in several stages In the first stage, we developed the logical template representations and the associated pattern-matching procedures described in [1] *and* [5] As a step toward the implementation of the prototype-plus-deformation representation, we have extended the pattern-matching procedures to deal with partial matches, and we *have added* several heuristics to produce a plausible version of a "best match" result We are now working on an implementation of the XDN formalism an interpreter for the elementary transformations has been written (see [22]). *and* an initial implementation of XONAPPLY has been tested out on the examples in the text The major work left to be done. then, is to define a search procedure in the space of MAPs which matches the observed procedures of Justice Pitney and Justice Brandeis in Eisner v Macomber Once this step is completed, we should be able to model the major components of legal reasoning in the corporate tax domain We expect our results to be of interest to workers in several areas of artificial intelligence research, and of *Qemra*\ utility in various areas of application

Acknowledgments: This research has been supported by the National Science Foundation under Grant SOC-78-11408 11978-79) and Grant MCS-79-21471 (1979-81) The authors *are* grateful to John Bresma, Richie Cantone, Donna Nagel and Julian Padget for their assistance in the implementation of the current system.

## REFERENCES

[1] McCarty, LT., and Sridharan, NS. "The Representation of an Evolving System of Legal Concepts: I Logical Templates" In *Proceedings of the Third Biennis/ Conference of the Canadian Society for Computations/ Studies of intelligence.* Victoria, British Columbia, 1960, 304-311

[2] McCarty, LT "Reflections on TAXMAN An Experiment in Artificial Intelligence and Legal Reasoning' *Hervard Law Review.* 90 (1977) 837-93

[3] Sridharan, MS. "AJMDS User Manual, Version 2", Technical report CBM-TR-89, Department of Computer Science, Rutgers University, 1978, (Version 3 forthcoming).

[4] Schmidt. C.F Sridharan. MS, and Goodson, J.L "Plan Recognition: An Intersection of Artificial Intelligence *and* Psychology' *Artificial Intelligence.* 9 (1978) 45-83.

[5] McCarty. LT., and Sridharan N.S The Representation of Conceptual Structures in TAXMAN II Part One Logical Templates", Technical report LRP-TR-4, Laboratory for Computer Science Research, Rutgers University. 1980.

[6] McCarty. LT "Some Requirements for a Computer-Based Legal Consultant" In *Proceedings of the First Annuel Netional Conference on Artificial Intelligence.* Stanford University, August, 1980, 298-300

[7] Hart. H.L.A. *The Concept of Lew.* Oxford Clarendon Press. 1961

[8] Oworkin. R. "Hard Cases' *Harvard Lew Review.* 88 (1975) 1057*1109

[9] McCarty. LT. "The TAXMAN Project Towards a Cognitive Theory of Legal Argument.' in *Computer Science end Lew: An Advenced Course,* B Nibiett, ed. Cambridge University Press. 1980. 23-43

[10] Wittgenstein. L *Philosophical Investigations.* Macmillan. 1953

[11] Kuhn T *The Structure of Scientific Revolutions.* University of Chicago Press. 1970

[12] Kuhn. T, "Second Thoughts on Paradigms." in *The Structure of Scientific Theories,* F Suppe. ed. University of Illinois Press. 1974

[13] Schwartz, S.P. *Naming, Necessity and Natural Kinds.* Cornell University Press, 1977

[14] Rosch, E. "Classification of Real-World Objects Origins and Representations in Cognition,' in *Thinking: Readings in Cognitive Science,* P.N Johnson-Laird and PC Wason. ed. Cambridge University Press. 1977. 212-22

[15] Rosch. E. *and* Lloyd, B.B *Cognition and Categorization.* Lawrence Erlbaum Associates, 1978

[16] Minsk y, M., *A Framework for Representing Knowledge,* McGraw-Hill, 1975. 211-77.

[17] Winston, *PH* "Learning by Creating and Justifying Transfer Frames." *Artificial Intelligence.* 10:2 (1978) 147-172

[18] Winston, *PH* "Learning and Reasoning by Analogy" *Communications of the ACM.* 23 12 (1980) 689-703

[19] Winograd. T. "On Primitives, Prototypes, and Other Semantic Anomalies" In *Proceedings, Second Workshop on Theoretical Issues in Natural Language Processing.* University of Illinois, 1978,

[20] Moore, J. *and* Newell, A, "How Can MERLIN Understand?/' in *Know/edge and Cognition,* L Gregg, ed., Lawrence Erlbaum Associates, 1974, 201

[21] Mitchell, TM, *Version Spaces: An Approach to Concept Learning,* PhO dissertation, Stanford University. 1978

[22] Sridharan, N.S "Transformations on Description Structures: Syntax. Semantics, and Examples from TAXMAN II". Technical report LRP-TR-12, Laboratory for Computer Science Research. Rutgers University, 1981, (forthcoming).

[23] McCarty, L.T. "A Computational Theory of Eisner v Macomber' In *Proceedings of the international Study Congress on Logic, informatics, and Law.* Florence, Italy, April, 1981, 553-95

[24] Brody, O.A "The Post-Macomber Cases in s TAXMAN II Framework A Preliminary Analysis', Technical report LRP-TR-5, Laboratory for Computer Science Research. Rutgers University, 1980