WORKING PAPER

CANDID DESCRIPTION OF COMMERCIAL AND FINANCIAL CONCEPTS: A FORMAL SEMANTICS APPROACH TO KNOWLEDGE REPRESENTATION

Ronald M. Lee

December 1981
WP-81-162

International Institute
for Applied Systems Analysis
A-2361 Laxenburg/Austria

IIASA

CANDID DESCRIPTION OF COMMERCIAL AND FINANCIAL CONCEPTS: A FORMAL SEMANTICS APPROACH TO KNOWLEDGE REPRESENTATION

Ronald M. Lee

December 1981
WP-81-162

INTERNATIONAL INSTITUTE FOR APPLIED SYSTEMS ANALYSIS
2361 Laxenburg, Austria

ABSTACT

The formal language CANDID is presented as a knowledge representation formalism for artificially intelligent decision support systems. The language is specifically oriented to representation of concepts in finance, commerce and administration. Later parts of the paper demonstrate the application of CANDID to the explication of corporate entities and contractual objects, as well as to various concepts in elementary finance.

ACKNOWLEDGMENTS

A very special thanks to Miyoko Yamada for a superb job of text editing and equation setting.

CONTENTS

INTRODUCTION		1
Part I:	SYNTAX AND FORMAL SEMANTICS OF CANDID	11
Part II:	FORMAL DESCRIPTION OF ECONOMIC ACTORS AND OBJECTS	75
Part III:	CANDID DESCRIPTION OF FINANCIAL CONCEPTS	99
REFERENCES		131

CANDID DESCRIPTION OF COMMERCIAL AND FINANCIAL CONCEPTS: A FORMAL SEMANTICS APPROACH TO KNOWLEDGE REPRESENTATION

Ronald M. Lee

INTRODUCTION

There is a growing interest in Decision Support Systems (DSS) research to incorporate the techniques and methods of Artificial Intelligence (AI), especially the areas of so-called knowledge-based expert systems. (See for instance, the increasing emphasis on AI in the DSS texts by Keen and Scott-Morton (1978) Fick and Sprague (1980), and Bonczek, Holsapple and Whinston (1981).)

Expert systems are characterized by the ability to do non-deterministic, qualitative deductions on a knowledge-base about some particular problem domain. Some of the best know examples are: the MYCIN system for bacterial infection diagnosis and therapy (Shortliffe 1976), the DENDRAL system which computes structural descriptions of complex organic chemicals from their mass spectograms and related data (Buchanan and Feigenbaum 1978), and the MACSYMA system for mathemaical formula manipulation (Martin and Fateman 1971).

A fundamental issue in the development of expert systems is the is the formalism for representing the contents of the knowledge base. The robustness of this formalism obviously determines the range of phenomena that can be discriminated and the types of deductions that can be performed on these descriptions. In AI, a variety of such formalisms have been proposed (see Brachman and Smith (1980) for a survey). These divide (roughly) into those using graphical schemes (called 'semantic nets') and those based on symbolic logic. For reasons which will become clear later on, the orientation here utilizes the notation of symbolic logic.

As argued in the above cited literature, a DSS might also usefully incorporate such knowledge-based inferencing techniques to 'intelligently' assist in decision making in some particular problem area.

Our purpose in this paper is to present a knowledge representation formalism, called CANDID, which is specifically oriented to typical DSS applications, focusing on the representation of concepts in administration, commerce and finance.

However, there is a certain difference in the requirements and priorities of a knowledge representation language for DSS's as proposed to expert systems. In an expert system one attempts to completely capture the expertise related to a given task. In a DSS, one typically addresses problems of greater complexity where at best only a partial formalization of the problem domain is possible. Hence, a DSS seeks to *aid* rather than replace the decision maker.

This raises an important methodological issue regarding the development of formalisms for the representation of knowledge in these systems.

In artificial intelligence, somewhat as in applied mathematics, a primary emphasis is placed on deductive capability and efficiency, leaving the modeling capacity of the formalism as a secondary priority. Thus, it often happens that computational tools are developed, and refined, while their application remains a craft, e.g., of an operations researcher or a knowledge engineer. Under this approach, if one can describe a problem in the appropriate formalism, a computational solution is automatic.

However, in decision support systems the philosophy is to attempt to go beyond the range of problems having completely structured computational solutions, and attempt to address areas which may be only partially formalizable (at current levels of understanding). This raises the thorny issue of how we can attempt to describe these more complicated problem domains without resorting to subjective discourse (also known as 'handwaving'). The challenge for problem domain description in decision support contexts is therefore the apparent contradiction of finding formal methods for describing only partially formalizable phenomena.

There is a key, however, in the interpretation of the word "formal." Here we make use of a distinction from meta-logic (see, e.g., van Fraasen (1971)) between the *formal semantics* of a notation and its *logical axiomatization*. The formal semantics of the notation (what is usually called a *formal language)* provides an unambiguous denotation or object of reference for each symbol and combination of symbols allowed in the notation's syntax. Denotations are generally described in set theoretical terms, where the sets are *sets of objects*, such as the set of people, the set of geographical location or sets of times. These sets may also coincidentally be symbolic objects such as numbers alphabetic letters, but these too are considered to be in a referential relationship to the symbols in the notation.

Two expressions in the notation are said to be *semantically equivalent* if they denote the same objects. Hence semantic equivalence can only be verified by reference to these external sets.

A logical axiomatization, on the other hand, involves a set of transformations, called *inference rules*, which have the claim that if the inputs to these rules (called *premisses)* are true expressions, then the output (called the *conclusion)* will also be a true expression. An important point is that these inference rules make use of purely syntactic information only. More broadly, if truth values are considered among the sets of objects that may be denoted, an inference rule asserts that *if* its input expressions, satisfying certain syntactic criteria, have a certain denotation, then its output expression will have a certain other denotation. However, the denotations themselves are not examined.

Application of an inference rule is called a *deduction*, and if one expression can be derived from on or more others by possibly many applications of these rules, it is said to be *deducible* from the other expressions. A set of axioms of the formal language is a set of expressions from which all other (valid) expressions may be deduced.

A *logic* for a formal language comprises the sets of inference rules and axioms. A logic is complete for the formal language if deducibility can be made to coincide with semantic equivalence.

The relevance of this discussion to the methodological problem posed for decision support systems should become clear if we associate the concepts

a. 'formal language' with 'knowledge representation scheme'

b. 'formal semantics' with 'modeling capability' (of the representation scheme).

c. 'deducibility' with 'computability.'*

As argued above, a methodology appropriate for decision support systems is one that places priority on modeling or what we might alternatively call formal description. This amounts to development of a representational scheme (formal language) with an explicit and unambiguous syntax and formal semantics. This formal semantics is described in terms of manipulations of sets of objects (some of which may be symbolic).

The division of labor between a human user and the computer decision support system for a particular problem domain described in such a formal language can now be described in rigorous terms: the potential of the DSS in this problem domain is precisely the range of deduciblity covered by the inference rules.

*Strictly speaking, 'computable' should be translated as 'efficiently deducible'—i.e., including an algorithm for applying the inference rules which halts in a reasonable amount of time.

The purpose of this work is thus one of "explication," Carnap's term for the task of "making more exact a vague or not quite exact concept used in everyday life or in an earlier stage of scientific or logical development."*

The conceptual vocabulary we seek to 'explicate' is the special terminology of commerce and finance: in particular the descriptive terminology found in accounting reports, financial and commercial contracts and administrative databases.

Part of this terminology deals with the particular class of goods and services involved—e.g., household products, transportation, energy. This is what some organization theorists (e.g., Woodward (1978)) have called the *technology* of the organization, i.e., in a very broad sense, what the organization knows how to do that distinguishes it from other organizations. For this aspect, our calculus provides a general framework within which these application dependent concepts can be developed.

On the other hand, there is a large number of concepts that are independent of the technology involved. This may be roughly divided into concepts relating to the organization's *bureaucratic structure*, i.e., its system of authority, and the organization's *contractual structure*, i.e., its commitments to other parties. (This is only an approximate distinction; bureaucratic structure will later be re-cast as set of interrelationships between contracts to employees.)

* Carnap, R. ***Meaning and Necessity***, Chicago, Univ. of Chicago Press, 1947, p.7 cited in Cresswell (1973), p.3.

The goal in CANDID is to explicate these types of concepts—e.g., what is a contract? what is common stock? what is an asset? a liability? what distinguishes a proprietorship, an partnership and a corporation?

Our goal here is therefore one of *description* rather than normative prescription.

We should note that the goal in accounting is also one of description of similar phenomena. However the objectives here are in fact complementary to those in accounting. Accounting is concerned with the *valuation* of these phenomena (in monetary terms). Our goal is the description of these phenomena independent of such valuations. (This work, insofar as it succeeds, therefore offers a descriptive foundation for accounting theory.)

Our goal, therefore, is to reduce this conceptual vocabulary to a set of primitive concepts about which there is no ambiguity. (The relationship to DSS knowledge bases is discussed in more detail in the next section.)

What we so far lack is a criterion for when we have arrived; put otherwise, why is the informal terminology presently in use not sufficient? Our reply is based on the philosophical work of Strawson (1959), who examines the necessary frame of reference needed for consensual understanding of objects and concepts. His conclusion is that the underlying basis for such understanding is its location in a spatial temporal framework.

For our purposes, this will be interpreted as an domain of discourse consisting of physical objects (having mass), including of course people, existing in the present or past.

One problem that immediately arises is the individuation of such objects, especially in the case of granualar substances and liquids or gases. As a simplification, which is realistic in most commercial contexts, we will assume these to be located in a container which can in turn be individuated and uniquely identified in time and space.

The question arises why we limit this domain of discourse to objects in the past and present, and not include the continuation of these objects, as well as other objects, in the future. Our response is that while a given spatial coordinate--at a future point in time *can* only be occupied by one physical object, we do not know whether or not it *does*. Thus the future will appear in CANDID as a framework of *possibility*, whereas the past and present constitute a framework of *fact*.

If we consider only the physical products and activities of an organizations, its explication in this domain of discourse would be relatively straightforward (though perhaps tedious).

However, the financial and bureaucratic concepts present a profound challenge. Consider the elementary concept of money. Cash is of course a physical object, but that is probably among the least interesting of its aspects. Similarly, a bond or a common stock is represented by a paper certificate, but again the real import of this object is something beyond that.

On the bureaucratic side, consider: what is a corporation? Is it the collection of its assets? No, for the corporation owns its assets and is therefore separate from them. Is it the collection of its employees? No, for the corporation contracts with its employees for their work, and is therefore separate from them. Is it the collection of its stockholders? No, for the corporation is owned by its stockholders, hence separate from them. What is it then?

Within this arises the issue of organizational authority. What is meant that x has authority over y? This is surely quite different than a simple physical relationship.

These are the sorts of phenomena we are attempting to explicate in the CANDID calculus.

The remainder of this paper is divided into three parts. In Part I, the syntax and formal semantics of the CANDID language are developed. In Parts II and III we illustrate how CANDID can be applied to the description of financial and commercial phenomena. In Part II, the entities, that is the principal actors and objects of economic activity, are considered. In Part III, various elementary concepts of finance are explicated using CANDID.

PART I: SYNTAX AND FORMAL SEMANTICS OF CANDID

CONTENT

A.	The Language L_1	13
B.	Re-interpretation of Predicates	18
C.	Many Sorted, Type Theoretic Languages	21
D.	Lambda Abstraction	23
E.	Operations, Definite Reference	25
F.	Summary of the Language L_T	28
G.	Character Strings, Labels	32
H.	Numbers and Measurement	34
I.	Time, Realization, Change	37
J.	Possible Worlds, Intensions	45
K.	Summary of the Language, IL	50
L.	Action	57
M.	Models, Deontic operators	59
N.	Summary of the Language CANDID	67

This part presents form the syntax and formal semantics of the language we have called CANDID, originally described in Lee (1980).

In the discussion which follows, the reader is presumed to be familiar with the first order predicate calculus (FOPC), which we take as our starting point. For background, we suggest that the text by Kalish, Montague and Mar (1980). The extensions to this which comprise CANDID are drawn chiefly from Montague's "intensional logic" (Montague 1973, Dowty 1978), and von Wright's "deontic logic" (von Wright 1965, 1967 and 1968), with minor influence from the temporal logic of Rescher and Urquhart (1971). The presentation given here is a model theoretic one. Background on model theory is giving in Dowty (1978) and Kalish et al. (1980) mentioned above. Deeper coverage is provided in van Fraasen (1971) and Chang and Keisler (1973).

The CANDID language as described here loosely follows the development of Montague's Intensional Logic as presented in Dowty (1978), augmented with the operators of von Wright's Deontic Logic. The principle differences up to the language IL (Intentional Logic) are as follows:

— addition of operations and the definite reference operator, ι

— omission of the tense operators, P and F (past and future)

— addition of the sets C (character strings) and N (numbers) in the model.

— recognition of time (designated as the set T rather than J) within the object language; addition of the operator R for temporal realization (adapted from a similar notation by Rescher and Urguhart (1971).

The language IL is then extended to include the connectives and operators of von Wright's deontic logic with the following modifications:

- addition of an agent place in the I connective.
- re-interpretation of contingent permission and obligation.
- addition of operators for contractual obligation and permission, and the connective OE (or else).

General Notational Conventions

Throughout this paper we will describe a series of formal languages of increasing complexity. The formal language itself will be called the *object language*, whereas its description is done via a ***metalanguage***.

Object Language—Constants

In the object language, *constant names* will be strings of upper or lower case Roman letters or digits or dashes, beginning with a capital letter. These will designate individuals in the domain. Later, the object language is extended to include symbolic entities, i.e., character strings and numbers. These may be designated directly in the language, without the intermediate device of a constant name. Character string constants will be shown between double quotes, e.g., "string," and numeric constants will have the usual Arabic notation, with an optional embedded decimal point, e.g., 1, 2, 3. For consistency these designations will be treated as names for themselves. Thus the general notation for constants is that they begin with a capital letter, digit or double quote.

Object Language—Variables

Variables will be denoted as one or more lower case letters, with an optional subscript; e.g., x, y, z_1, z_2.

Metalanguage

In the metalanguage, constants will be represented using the Greek characters, α, β, γ, Φ, Ψ. Variables will be designated in the metalanguage by the characters μ and ν.

I-A. THE LANGUAGE L_1

L_1 is a fairly standard version of a first order predicate calculus with equality.

Syntax of L_1

Basic Expressions

Constants:

Individual Constants: will be denoted as a capital letter followed by one or more lower case letters, e.g., A, B, Tom, Dick, Harry.

Individual Variables: are denoted as one or more lower case letters with an optional subscript, e.g., x, y, z_1, z_2.

Predicate constants: are denoted as one or more capital letters, e.g., P, Q, RED. Each predicate has associated zero or more *places*. (A zero place predicate is called a *proposition.)*

Terms:

A *term* in L_1 is an individual variable or an individual constant.

Formation Rules of L_1

A well formed formula (wff) of L_1 is defined recursively as follows:

1. If Φ is a predicate of n places, $(n \geq 0)$ and $\alpha_1, \ldots, \alpha_n$ are terms, then $\Phi(\alpha_1, \ldots, \alpha_n)$ is a wff.

2-6. If Φ and Ψ are wffs, then so are:

2. $\sim\Phi$

3. $\Phi \,\&\, \Psi$

4. $\Phi \vee \Psi$

5. $\Phi \rightarrow \Psi$

6. $\Phi \leftrightarrow \Psi$

7-8. If μ is a variable and Φ a wff, then:

7. $\forall\mu\, \Phi$ is a wff

8. $\exists\mu\, \Phi$ is a wff

A variable μ is *bound* in a formula Φ iff it occurs in Φ within a subformula of the form $\forall\mu\, \Phi$ or $\exists\mu\, \Phi$; otherwise the variable is *free* in Φ.

A *sentence* is a wff containing no free variables.

Semantics of L_1

A model for L_1 is an ordered pair <D, F> such that D (the *universe of discourse)* is a non-empty set and F (the *interpretation function) is a function assigning a denotation to* each constant of L_1 (i.e., to individual constants and predicate constants). The set of possible denotations of individual constants is D. The set of possible denotations of one place predicates is ρ(d) (where ρ is the power set of D, i.e., the set of all subsets). The set of possible denotations for an n place predicate is $\rho(D^n$ where $D^n = \{<d_1, \ldots, d_n> \mid d_1 \in D, \ldots, d_n \in D\}$.

The set of possible denotations for a 0 place predicate (proposition), is the set {True, False}.

An *assignment of values to variables* (or *value assignment)* g is any function assigning a member of D to each variable of L_1. $Den_{M,g}$ (α)" is the abbreviation for "denotation of α with respect to M ang g" "true wrto M,g" abbreviates "true with respect to a model M and value assignment g."

Denotations of Basic Expressions of L_1
(relative to a model <D, F> and value assignment g)

1. If μ is an individual variable of L_1, then $Den_{M,g}(\mu) = g(\mu)$.

2. If α is a (non-logical) constant of L_1, then $Den_{M,g}(\alpha) = F(\alpha)$.

Truth Conditions for Formulae of L_1
(relative to a model <D, F> and Value Assignment g)

1. If Φ is an n place predicate and $\alpha_1, \ldots, \alpha_n$ are terms, then $\Phi(\alpha_1, \ldots, \alpha_n)$ is true wrto M,g iff $Den_{M,g} <\alpha_1, \ldots, \alpha_n) \in Den_{M,g}\, \Phi$.

2. If Φ is a wff, then $Den_{M,g}$ $(\sim\Phi)$ = true iff $Den_{M,g}$ (Φ) = False

3-6. If Φ and Ψ are wffs, then

3. $Den_{M,g}[\Phi \ \& \ \Psi]$ = True iff $Den_{M,g}(\Phi)$ = True and $Den_{M,g}(\Psi)$ = True

4. $Den_{M,g}[\Phi \vee \Psi]$ = True iff either $Den_{M,g}(\Phi)$ = True or $Den_{M,g}(\Psi)$ = True.

5. $Den_{M,g}[\Phi \rightarrow \Psi]$ = True iff either $Den_{M,g}(\Phi)$ = False or else $Den_{M,g}(\Psi)$ = True

6. $Den_{M,g}[\Phi \leftrightarrow \Psi]$ = True iff either (a) $Den_{M,g}(\Phi)$ = True and $Den_{M,g}(\Psi)$ = True or (b) $Den_{M,g}(\Phi)$ = False and $Den_{M,g}(\Psi)$ = False

7. If Φ is a formula and μ is a variable, then $Den_{M,g}(\forall\mu\Phi)$ = True iff for every value assignment g' such that g' is exactly like g except possibly for the individual assignment to μ by g', $Den_{M,g'}(\Phi)$ = True.

8. If Φ is a formula and μ is a variable then $Den_{M,g}(\exists\mu\Phi)$ = True iff there is some value assignment, g', such that g' is exactly like g except possibly for the value assigned to us by g' and $Den_{M,g'}(\Phi)$ = True.

Truth Conditions for Formulae of L_1 Relative to a Model M

1. A formula Φ of L_1 is ***true with respect to M*** if for all value assignments g, $Den_{M,g}(\Phi)$ = True

2. A formula Φ of L_1 is *false with respect to M* if for all value assignment g, $\mathrm{Den}_{M,g}(\Phi)$ = False.

Note: If a formula Φ is a sentence or proposition (i.e., with no free variables), then it will turn out that $\mathrm{Den}_{M,g}(\Phi)$ = True with respect to M and all value assignments, g (hence true with respect to M by 1. above) or else $\mathrm{Den}_{M,g}(\Phi)$ = False with respect to M and all value assignments (hence false with respect to M and all value assignments (hence false with respect to M, by 2. above). It can never be true with respect to M and some value assignments and false with respect to other value assignments.

However, if Φ has one or more free variables, then it may be true with respect to some assignments and false with respect to others. In this case its truth or falsity is simply undefined by the above rules.

I-B. RE-INTERPRETATION OF PREDICATES

In the preceding section, a one place predicate was regarded as denoting a subset of the domain D. Hence, for a term α and a predicate Φ, $\Phi(\alpha)$ is true (denotes True) if and only if the thing α denotes is an element of the set denoted by Φ.

Similarly for n-place predicates, Φ is viewed as an n-place relation on D, and is true of n terms, $\alpha_1, \ldots, \alpha_n$ iff the n-tuple of entities they denote is an element of the relation denoted by Φ.

This interpretation will now be modified slightly. Consider first the case of a one place predicate. Suppose we had a domain, D, consisting of five individuals as follows:

$$D = \{D1, D2, D3, D4, D5\}$$

and a predicate, P, whose denotation is as follows:

$$Den(P) = \{D1, D4, D5\}$$

Here, P is true (denotes True) of the individuals in this set and is false (denotes False) of the individuals not in this set.

These denotations of True and False can be made explicit by describing the *characteristic function* of P. This is a function that maps any individual in D to True or False, according to whether it is in the subset of D denoted by P. For instance, the characteristic function in this case is the set of pairs:

<D1, True>

<D2, False>

<D3, False>

<D4, True>

<D5, True>

The information conveyed here is essentially that of the previous subset plus the interpretation of elementhood conveying the truth of the predicate applied to its argument. However here, this interpretation is conveyed directly.

That is, let us henceforth view a one place predicate as denoting the characteristic function of the set of elements for which it is true.

Then the denotation of the predicate applied to an argument is simply the result of functional application of this argument to the characteristic function, i.e., if Φ is a one place predicate and α is a term, then

$$Den_{M,g}(\Phi\alpha) = Den_{M,g}(\Phi)(Den_{M,g}(\alpha)).$$

For instance, in the above example, if α is D1, then Den($\Phi\alpha$) = True if α is D2, then Den($\Phi\alpha$) = False.

We might similarly extend this so that two place predicates denoted sets of triples, mapping two individuals to a truth value, and that n place predicates denoted n+1 tuples mapping n individuals to a truth value.

However, it will provide more flexibility later on if we regard a two place predicate not as a function of two arguments mapping to truth values, but as a function of one argument mapping to another function of one argument which maps to a truth value.

Thus a predicate of any number of places is considered to denote a function of only one argument whose result is either another function or a truth value. (The idea of functions which have other functions as values may seem strange—except perhaps to LISP programmers. Its motivation will become clear when we introduce lambda abstraction.)

To incorporate this new interpretation, the language L_1 is modified as follows:

Replace formation rule 1 with

Syn.1a. If Φ is a one place predicate and α is a term, then $\Phi\alpha$ is a wff.

Syn.2a. If Φ is an n place predicate and α is a term, then $\Phi\alpha$ is an n-1 place predicate.

Replace semantic rule 1 with:

Sem.1. If Φ is an n place predicate ($N \geq 1$) and α is a term, then $\mathrm{Den}_{M,g}(\Phi\alpha) = \mathrm{Den}_{M,g}(\Phi)(\mathrm{Den}_{M,g}(\alpha))$.

Note that the previous notation $\Phi(\alpha_1, \alpha_2, \ldots, \alpha_n)$ now takes the form $\Phi(\alpha_1)\ (\alpha_2)\ \ldots\ (\alpha_n)$. The former notation will still be used on occasion to abbreviate the latter however.

I-C. MANY SORTED, TYPE THEORETIC LANGUAGES

A *many- sorted* formal language is one that assumes there is a non-empty set I whose members are called *sorts*. For each sort i, there are variables V_1^i, V_2^i,... that belong to sort i. Also for each sort i there is a (possibly empty) set of constant symbols of sort i.

For each n > 0 and each n-type $<i_1, \ldots, i_n>$ of sorts, there is a (possibly empty) set of predicates, each of which is said to be of sort $<i_1, \ldots, i_n>$. For each sort i there is a universal and existential quantifier, $\forall_i$ and $\exists_i$. A many sorted logic can be embedded in a first order predicate calculus and therefore does not have any more power (Enderton 1972).

A many sorted approach will prove valuable later when we extend the domain of the formal language to include in addition to entities (whose designation we leave imprecise until later), character strings, numbers and times.

The purpose of a many sorted language is to coordinate references among the several domains of discourse representing each sort. As noted above, these references remain "first-order," i.e., only individual and properties and relationships of individuals (indicated by predicates) are represented in the language.

Recall that in the previous section we modified the interpretation of a predicate so that it no longer denoted a set or relation on a domain, but rather characteristic functions of such sets.

Rather than orient the formal language towards the first order framework of a multi-sorted language, we will instead continue the development begun in the last section and introduce a more general framework that includes the multiple domain features of a multi-sorted language. Such a language is called a ***higher-order type-theoretic*** language (the name derives from origins in Russell's simple theory of types.)

Basically, a type is like a sort as described above, except that a type may be not only a class of individuals (like a sort), but classes of higher order objects (e.g., sets, sets of sets) as well. So far, the elementary types we have discussed are: individuals in the domain, designated as type "e" (for "entity"), and truth values, which we designate as type "v" (from Latin, ***veritas;*** the obvious abbreviation "t" is reserved for time, which appears later).

The set of types, called Type, is defined recursively as follows:

(1) e is a type

(2) v is a type

(3) if a and b are any types, then <a,b> is a type.

The members of Type are labels of categories. The notation ME_a, (the ***meaningful expressions*** of type a) denotes the set of expressions of type a itself.

By way of example

a formula or proposition is of type v.

a one place predicate is of type <e,v>.

a two place predicate is of type <e,<e,v>>.

~ is of type <v,v>.

connectives (&, V, →, ↔) are all of type <v,<v,v>>

I-D. LAMBDA ABSTRACTION

Using set notation, a set may be defined extensionally, listing its elements, e.g.,

$$A = \{A1, A2, A3\}$$

or "intensionally," by means of some predicate that selects from the domain a subset of individuals, e.g.,

$$B = \{x \mid P(x)\}$$

is the set of all individuals satisfying P.

This brace notation thus provides the means for constructing higher order sets from a predicate.

But, by our first interpretation of predicates, they themselves denoted sets, e.g., as Den(P). Thus substituting,

$$B = \{x \mid x \in Den(P)\}$$

or

$$B = Den(P)$$

In our revised interpretation, however, the denotation of P was modified to be the characteristic function of this set. The device for referring to this in the object language is the so-called *lambda operator*, λ.

Thus, for P a one place predicate,

$$\lambda x P(x)$$

is the set of ordered pairs of the form $<e,v>$, one pair for each individual in the domain, which assigns True or False if P is true or false of that individual respectively.

While we have introduced lambda in terms of individuals and one place predicates, it can in fact be generalized to apply to expressions and variables of any type.

This involves the following additions to the syntactic and semantic rules:

Syn. If $\alpha \in ME_a$ and μ is a variable of type b, then $\lambda\mu a \in ME_{<b,a>}$.

Sem. If $\alpha \in ME_a$ and μ is a variable of type b, then $Den_{M,g}$ $(\lambda\mu\alpha)$ is that function h from D_b into D_a such that for all objects k in D_b, h(k) is equal to $Den_{M,g'}(\alpha)$, where g' is that variable assignment exactly like g except for the possible difference that $g'(\mu) = k$.

Note that lambda abstraction takes the role of set definition and functional aplication the role of set membership in the *object languages* we are developing, whereas traditional set concepts are used in the *metalanguage* definitions.

In later parts, where we illustrate the use of CANDID with examples, it will occassionally be convenient to revert back to traditional set notation because of its familiarity. For this reason, we incude the following additional definitions in the *object language*.

For a predicate Φ and variable μ,

$$\{\mu \mid \Phi(\mu)\} ::= \lambda u[\Phi(u)]$$

$$\mu \in \Phi ::= \Phi(u)$$

To repeat, while set notation may thus be used in the *object language*, its interpretation is in terms of lambda abstraction and characteristic functions. In the *metalanguage* definitions, set notation is used in the normal way.

I-E. OPERATIONS, DEFINITE REFERENCE

The expressions discussed so far have all been of the type e or <a,v> where a is some other, possibly complex, type.

An *operation* is an expression of the form <a,b> where b is an elementary type other than v. At the current level of the language, the only expressions that qualify are of the form <a,e>, i.e., expressions which result in an individual, when applied to an argument. Indeed, an individual constant may be regarded as a 0-place operation.

Operations may thus serve as arguments to predicates, e.g., for the predicate "ITALIAN," the operation, "Father,"

ITALIAN(Father(John))

asserts that John's father is Italian.

Note: to aid readability in the examples, we adapt the following practice for constant names: constants denoting individuals (individual constants and operations) are given names beginning with a capital letter, followed by lower case. Other constants, including predicates, are given names all in upper case.

Operations serving as arguments to predicates is included in the definition of functional application given in the preceding section; i.e., the argument to a functor of type <a,b> may be any meaningful expression of type a, this includes operations as well as variables and constants.

For instance, in the above example

$$\text{John} \in ME_e$$
$$\text{Father} \in ME_{<e,e>}$$
$$\text{ITALIAN} \in ME_{<e,v>}$$

so that, by functional application

$$\text{Father(John)} \in ME_e$$
$$\text{ITALIAN(Father(John))} \in ME_v.$$

(The quantifiers $\forall$ and $\exists$ as well as the lambda operator, λ, are confined by definition to variables only.)

Note that by combining an operation with a predicate of equality we can define a corresponding predicate:

$$\text{FATHER}(x,y) \quad ::= \quad y=\text{Father}(x)$$

A new operator, the so-called ***descriptive*** or ***iota operator***, ι will allow us to make definitions in the other direction.

This operator has the following syntactic and semantic rules:

Syn. If $\Phi \in ME_{<a,v>}$ and μ is a variable of type a, then $\iota\mu\Phi \in ME_a$.

Sem. For $\Phi \in ME_{<a,v>}$ and μ a variable of type a, if for some constant c,

$Den_{M,g}\forall\mu[\Phi(\mu) \leftrightarrow \mu=c]$ = True, then $Den_{M,g}(\iota\mu\Phi\mu) = c$.

Note that by this definition, the expression $\iota\mu\Phi\mu$ has a denotation only if Φ is true of just one individual; otherwise—i.e., if Φ is true of no individuals or more than one individual—the denotation of $\iota\mu\Phi\mu$ is undefined.

Expressions of the form "$\iota\mu\Phi$" are read "the unique μ such that Φ." Iota is thus an *operation forming* operator. For example, the earlier operation Father(x), could be formed from the predicate FATHER(x,y) as follows:

$$\lambda x \text{ Father}(x) ::= \lambda x \iota y \text{ FATHER}(y,x)$$

Comment: By way of comparison $\lambda\mu\Phi\mu$ denotes the set (or rather characteristic function thereof) of individuals satisfying Φ. This may, coincidentally, be a set with only one element (characteristic function with only one domain value mapping to True), or indeed it may be the null set.

$\iota\mu\Phi\mu$, on the other hand, denotes a single individual if it denotes at all.

I-F. SUMMARY OF THE LANGUAGE L_T

The language L_t incorporates the features discussed thus far:

Syntax of L_t

The set of *types* of L_t is the set defined as follows:

1). e is a type

2) v is a type

3) if a and b are types, then <a,b> is a type.

The basic expressions of L_t consists of

1) for each type a, the set of (non-logical) *constants* of *type a*, denoted Con_a. (Names for particular constants follow the conventions defined earlier: all constants names begin with a capital letter. Names of constants which refer to entities, have the remainder spelled in lower case; all other constants have names spelled entirely in upper case.

2) for each type a, the set of *variables of type a*, denoted Var_a. (Names for variables are as before, i.e., lower case letters with an optional numeric subscript.)

3) for each type a, the set of *terms of type a*, denoted $Term_a$, are defined recursively as follows:

   — if $\alpha \in Con_a$ then $\alpha \in Term_a$

   — if $\alpha \in Var_a$ then $\alpha \in Term_a$

— if $\beta_1, \ldots, \beta_n$ are terms of type $x_1,\ldots,x_n$ respectively and Φ is an operation of type $<x_1,\ldots,<x_n,a>>$ then $\Phi(\beta_1, \ldots, \beta_n \in$ $Term_a$.

— if $u \in Var_a$ and $\Phi \in ME_{<a,t>}$, whose only unbound variable is μ, then $\iota\mu\Phi \in Term_a$.

Formation Rules of L_t

The set of meaningful expressions of type a, denoted "ME_a", for any type a (the well formed expressions for each type) is defined recursively as follows:

1. For each type a, every variable and constant of type a is in ME_a.
2. If $\alpha \in ME_{<a,b>}$ and $\beta \in ME_a$, then $\alpha(\beta) \in ME_b$.
3. If $\alpha \in ME_a$ and μ is a variable of type b, then $\lambda\mu\alpha \in ME_{<b,a>}$.
4. If α and β are terms of type a, then $[\alpha=\beta] \in ME_v$.
5. If $\Phi \in ME_{<a,t>}$ and $\mu \in Var_a$ then $\iota\mu\Phi \in ME_a$.

6-10. If Φ and Ψ are in ME_v, then so are:

6. $[\sim\Phi]$
7. $[\Phi \mathbin{\&} \Psi]$
8. $[\Phi \vee \Psi]$
9. $[\Phi \rightarrow \Psi]$
10. $[\Phi \leftrightarrow \Psi]$

11-12. If $\Phi \in ME_v$ and μ is a variable (of any type) then

11. $[\forall \mu \Phi] \in ME_v$.

12. $[\exists \mu \Phi] \in ME_v$.

Semantics of L_t

Given a non-empty set D (the domain of *individuals* or *entities)*, the set of possible denotation of meaningful expressions of type a (abbreviated D_a) is given by the following recursive definition

(1) $D_e = D$

(2) D_v = {True, False}

(3) $D_{<a,b>} = D_b^{D_a}$ for any types a and b.

(the notation of the form Y^X is the set of all possible functions from the set X into the set Y.)

A *model* for L_t is an ordered pair <D, F> such that D is as above and F is a function assigning a denotation to each constant of L_t of type a from the set D_a.

An *assignment of values to variables* (or simply a *variable assignment) g is a function assigning to each variable* $\mu \in Var_a$ denotation from the set D_a, for each type a.

The denotation of an expression α of L_t relative to a model M and variable assignment g is defined recursively as follows:

1. If α is a constant, then $Den_{M,g}(\alpha = F(\alpha)$.

2. If α is a variable then $Den_{M,g}(\alpha) = g(\alpha)$.

3. If $\alpha \in ME_{<a,b>}$ and $\beta \in ME_a$, then $Den_{M,g}(\alpha(\beta)) = Den_{M,g}(\alpha)(Den_{M,g}(\beta))$. (i.e., the result of applying the function $Den_{M,g}(\alpha)$ to the argument $Den_{M,g}(\beta)$).

4. If $\alpha \in ME_a$ and $\mu \in Var_b$, then $Den_{M,g}(\lambda\mu\alpha)$ is that function h from D_b into D_a such that for all objects k in D_b, h(k) is equal to $Den_{M,g'}$, where g' is that variable assignment exactly like g except for the possible difference that $g'(\mu) = k$.

5. If α and β are terms of type α, then $Den_{M,g}(\alpha=\beta)$ is True iff $Den_{M,g}(\alpha)$ is the same as $Den_{M,g}(\beta)$.

6. For $\Phi = ME_{<a,t>}$ and $\mu \in Var_a$, if for some constant, c, $Den_{M,g}\ \forall\mu\ [\Phi\mu \leftrightarrow \mu=c]$ = True, then $Den_{M,g}(\iota\mu\Phi\mu] = c$. (Otherwise the expression $\iota u\Phi u$ has no denotation defined.)

7-11. For Φ and Ψ in ME_t

7. $Den_{M,g}(\sim\Phi)$ = True iff $Den_{M,g}(\Phi)$ = False

8. $Den_{M,g}[\Phi\ \&\ \Psi]$ = True iff $Den_{M,g}(\Phi)$ = True and $Den_{M,g}(\Psi)$ = True.

9. $Den_{M,g}[\Phi \vee \Psi]$ = True iff $Den_{M,g}(\Phi)$ = True or $Den_{M,g}(\Psi)$ = True.

10. $Den_{M,g}[\Phi \rightarrow \Psi]$ = True iff either $Den_{M,g}(\Phi)$ = False or else $Den_{M,g}(\Psi)$ = True.

11. $Den_{M,g}[\Phi \leftrightarrow \Psi]$ = True iff either a) $Den_{M,g}(\Phi)$ = True and $Den_{M,g}(\Psi)$ = True or b) $Den_{M,g}(\Phi)$ = False and $Den_{M,g}(\Psi)$ = False.

12. If $\Phi \in ME_t$ and μ is a variable, then $Den_{M,g}(\forall\mu\Phi)$ = True iff for all g' such that g' is exactly like g except possibly for the value assigned to μ, $Den_{M,g'}(\Phi)$ = True.

13. If $\Phi \in ME_t$ and μ is a variable, then $Den_{M,g}(\exists\mu\Phi)$ = True iff there is some g' exactly like g except possibly for the value assigned to μ and $Den_{M,g'}(\Phi)$ = True.

I-G. CHARACTER STRINGS, LABELS

We now introduce a new elementary type, called a *character string*, abbreviated by the type name, c. The set of types (Type) is therefore extended as follows:

e is a type

c is a type

v is a type

if x and y are types, then <x,y> is a type.

The set of elementary characters is the set Char where

$$Char = \{A, B, \ldots, Z, 0, 1, \ldots, 9, ., -\}$$

This character set is sufficient for our purposes here. It can be extended as needed to include e.g., lower case letters, special character markings such as accents, circumflex, cedilla, tilde, or completely different alphabets such as Cyrillic or Greek.

The set C of character strings is the set of n-place relations defined on Char, i.e.,

$$C = \bigcup_{n=1}^{\infty} \text{Char}^n$$

where Char^n is Char x Char x Char..., n times.

A character string constant is therefore an n tuple $<\alpha_1, \alpha_2, \ldots, \alpha_n>$ where $\alpha_i \in C$. This will henceforth be abbreviated

$$"\alpha_1 \alpha_2 \cdots \alpha_n"$$

i.e., a character string constant is a string of characters from the above set C listed between double quotes.

Various computer languages, such as SNOBOL, provide a rich vocabulary of predicates and operations on strings. Here we make use of only the bare minimum of such a vocabulary, namely predicate of equality which is defined for all types in the calculus. Again this could be extended as needed for diverse applications.

Here the principle interest in character strings is with operations of the form <e,c>. This is a mapping from an entity to a character string, what we call a *label*. Example of labels are:

Last-Name(x) = "SMITH"

First-Name(y) = "JOHN"

Corp-Name(z) = "GENERAL MOTORS"

Vehicle-Number(a) = "N33E76"

Social-Security-Number(b) = "474-52-4829"

As is probably evident from these examples, a label is an association of a character string with an individual for identification purposes only. Labels may or may not provide unique identification, as the above examples illustrate.

I-H. NUMBERS AND MEASUREMENT

Another elementary type is now added, that of *numbers*, which we take to be the real numbers. The set of numbers is designated as N, and the elementary type, number, is abbreviated n. The set of types is now extended as follows:

e is a type

c is a type

n is a type

v is a type

if x and y are types, then <x,y> is a type.

Numeric constants are denoted in the common way as a string of Arabic digits, with an optional imbedded decimal point and an optional leading sign, e.g., 0, 1.2, -3.7.

The one-place predicate I (i.e., of type <n,x>, designates the set* of integers.

As for all types, the predicate "=" is assumed. Further, a linear ordering, indicated by the predicate "<" is assumed. Based on these, plus negation, the other numeric inequalities are easily derived. The notation is as follows, for α and β terms of type n:

$\alpha = \beta$	α equals β
$\alpha < \beta$	α less than β
$\alpha \leq \beta$	α less than or equal to β

* Technically, the characteristic function thereof.

$\alpha > \beta$ α greater than β

$\alpha \geq \beta$ α greater than or equal to β

$\alpha \neq \beta$ α not equal to β

These predicates are all of type <n,<n,v>>.

The following operations, of type <n<n,n>> are also assumed:

For α and β terms of type n:

$\alpha + \beta$ addition

$\alpha - \beta$ subtraction

$\alpha * \beta$ multiplication

α / β division

$\alpha ** \beta$ exponentiation.

Our principle interest in numbers in CANDID is as they are related to entities (and later, times).

An operation of type <x,t>, where x is a term of type e, is called a *measurement function:* i.e., it is a mapping from the entities to the numbers (or a subset thereof). For instance,

$$\text{Height-in-meters}(x) = 6.5$$

indicates that x is 6.5 meters tall.

In the theory of measurement, a measurement is generally taken to involve a so-called *measurement operation* and a *measurement standard*. Measurement standards are the sorts of objects maintained by e.g., the National Bureau of Standards in Washington D.C., which have some special property against which other objects are to be gauged. Thus a particular rod is regarded as the standard meter for the country. (A more picturesque example: the roundish stone on the front of St. Stephan's cathedral in the center square of Vienna was used in medieval times as a standard for the size of a loaf of bread.)

A measurement operation is the procedure by which another object is compared to the standard. This procedure may be direct, e.g., by aligning the object against the standard meter, or indirect, through the use of intermediating measurement devices (rulers measuring tapes, etc.) which are ultimately compared to the standard.

In the formal language, a measurement operation is regarded as a (formal) operation, while a measurement standard is an individual constant. For instance, we may modify the last example to be:

$$\text{Height}(x,\text{Meter}) = 6.5$$

Here Height is a measurement operation and Meter is a measurement standard. Note that measurement operations are numeric terms and thus may appear as arguments to other numeric predicates and operations. E.g., to assert measurement unit convertibility from inches to centimeters:

$$\forall x\ \text{Height}(x,\text{Cm}) = 2.5 * \text{Height}(x,\text{Inch})$$

where Cm and Inch are measurement standards.

I-I. TIME, REALIZATION, CHANGE

Another elementary type is now added, consisting of elementary points in *time*. The set of times (past present and future) is denoted T, and its corresponding type, t.

The set of types is thus extended as follows:

e is a type

c is a type

n is a type

t is a type

v is a type

if x and y are types, then <x,y> is a type.

Equality, "=", and "<", a linear ordering, are assumed as predicates on T. With the aid of negation and disjunction, other temporal relations are defined in a straightforward way. If α and β are terms of type t, these have the following interpretation:

$\alpha = \beta$	α is the same time (point) as β
$\alpha < \beta$	α is earlier than β
$\alpha \leq \beta$	α is earlier or equal to β
$\alpha > \beta$	α is later than β
$\alpha \geq \beta$	α is later or equal to β
$\alpha \neq \beta$	α is not equal to β

Lastly, the predicate NEXT, indicates adjacent points in time.

$$\mathrm{NEXT}(\alpha,\beta) ::= \alpha < \beta \ \& \ \forall u \ (u \neq \alpha) \ \& \ (u \neq \beta) \rightarrow \sim(\alpha < u < \beta).$$

In many cases, our interest is not with points in time, but rather time intervals or *spans*. A time span is the set of points between and including two time points. This is provided by the operation Span, of type <t,<t,<t,v>>>:

$$\text{Span} ::= \lambda x\ \lambda y\ \lambda z\ [(z \geq x)\ \&\ (z \leq y)].$$

For two time points, α and β,

$$\text{Span } (\alpha(\beta)$$

is the set (technically, characteristic function) of points between and including these two points. Further, for a third time point, γ,

$$\text{Span}(\alpha)(\beta)(\gamma)$$

evaluates True or False depending whether γ is between α and β or not. (Note: as we have defined it, Span can also be used to select an interval of numbers.) Conversely, it is often convenient to go in the opposite direction to obtain the beginning and end points of a time span:

$$\text{Beg} ::= \lambda x\ \iota y\ \exists z\ [x = \text{Span}(y,z)]$$

$$\text{End} ::= \lambda x\ \iota z\ \exists y\ [x = \text{Span}(y,z)]$$

Thus, for a time span α, Beg(α) is its beginning point, End(α) is its ending point.

It is also occasionally useful to express that one time span is contained within another. We call this PT (for part)

$$\text{PT} ::= \lambda x\ \lambda y\ [\text{Beg}(x) \geq \text{Beg}(y)\ \&\ \text{End}(x) \leq \text{End}(y)]$$

Thus for two time spans α and β, PT(α)(β) says that β begins after α and ends before it.

As noted, Span(α)(β) results in (map to) a set of time points of type <t,v>. Many of these time spans have familiar labels, as provided by the Gregorian calendar, e.g., 28 February, 1981 and 10 July, 1984 are two individual day time spans, February 1981 and July 1984 are two individual month time spans and 1981 and 1984 are two individual year time spans. Reference to the time span constants labeled by the Gregorian calendar will be provided by three operations:

Date	of type	<n,<n,<n,<t,v>>>>
Mo	of type	<n,<n,<t,v>>>
Yr	of type	<n,<t,v>>.

That is, each of these maps (three, two or one) numbers to time spans, where months are specified by an integer 1-12. Thus the operation Date imitates the informal notation, e.g., 28/2/81. the time spans mentioned above would thus be,

Date(28,2,1981)

Date(10,7,1984)

Mo(2,1981)

Mo(7,1984)

Yr(1981)

Yr(1984).

Further, we often want to apply numeric measurement to time spans. For this we use the measurement operation Dur (for duration).

Thus for a time span α, a temporal measurement standard β and a number γ,

$$\mathrm{Dur}(\alpha,\beta) = \gamma$$

is read that the duration of α in terms of β is γ.

The choice of measurement standards is however somewhat problematic in the case of time spans. Standards such as Second, Minute and Hour pose no particular problems since these are precisely determined based on a particular physical phenomenon (e.g., movement of a standard pendulum, molecular vibrations of quarts). Generally for commercial purposes however we have need of larger size units, e.g., days, months, years.

Following the procedure recommended earlier, suppose we chose one particular month to serve as our standard—say, January 1981. Then, the duration of a year, e.g., 1983, in terms of this standard month would be:

$$\mathrm{Dur}(\mathrm{Yr}(1983),\mathrm{Mo}(1,1981)) = 11.77.$$

If, however, we take the next month as our standard, i.e., February, 1981, we would have:

$$\mathrm{Dur}(\mathrm{Yr}(1983),\mathrm{Mo}(2,1981)) = 13.04$$

Neither of these accords with the popular usage that a year comprises twelve months.

A similar, though slightly less serious problem arises in the choice of a standard year, since leap years do not have the same number of days as other years.

Indeed, even the choice of a standard day has potential difficulties, since the length of the last day in a century is slightly longer than the rest.

This however seems to be a tolerable level of inaccuracy. Thus, we may take as our standard, call it Day, any of the non end of the century days or equivalently, define it in terms of standard hours, minutes or seconds. Thus, for example,

$$\mathrm{Dur}(\mathrm{Mo}(1,1981),\mathrm{Day}) = 31$$

$$\mathrm{Dur}(\mathrm{Mo}(2,1981),\mathrm{Day}) = 28$$

$$\mathrm{Dur}(\mathrm{Date}(1,1,1984),\mathrm{Day}) = 1$$

We next consider the association of times to entities. For this we adopt a notation suggested by Rescher and Urquhard (1971) where for a time point, α, and a formula Φ

$$R(\alpha)\Phi$$

is read that Φ is "realized" at time α. E.g., if Φ is the formula "it is raining," this expression would be true at certain times, false at others.

Including this in our formal language would obviously require a syntactic rule like:

Syn. If α is a term of type t and $\Phi \in ME_v$, then $R(\alpha)\Phi$ is in ME_v.

However, the inclusion of the R operator will lead us to revise our semantic format somewhat. Like character strings and numbers, time points are merely another sort added to the object language. Viewed this way, the R operator is simply a functional application, i.e.,

$$R(\alpha)\Phi ::= \Phi(\alpha)$$

(This would of course assume that a variable ranging over time points was lambda abstracted within Φ.)

However, in order to make various needed discriminations in our semantic rules, we prefer to take a different tack: in addition to including time in the object language, we will also include it in our metalanguage.

That is to say, time is not only another sort or type within the object language, but will also figure as an additional dimension on which the denotation depends in the metalanguage. Or, one may regard it as though there were actually two times involved: those referred to within the expression, and the time of the expression itself.

In order to make the separation clear, we will use variables beginning with "t" in the object language to stand for times. In the metalanguage we will indicate times as "j". (This latter maintains a notational convention begun by Montague.) Thus, where we formerly wrote $Den_{M,g}\Phi$, we will now write $Den_{M,j,g}\Phi$. The semantic rule for R is therefore as follows:

Sem. For α a term of type t, and $\Phi \in ME_{v}$, $Den_{M,j,g}R(\alpha)\Phi$ = True iff for some j', j' = α and j' < j, $Den_{M,j',g}(\Phi)$ = True.

Some explanation might be in order. Here, and henceforth, j will be the time with the expression in question is interpreted, i.e., when the denotation is evaluated (in computer terms, the time when the database is queried). $R(\alpha)\Phi$ is true at this time if and only if Φ is true at some earlier time, α. Note that if α refers to some *future* time, i.e., $\alpha > j$, then the denotation of the expression $R(\alpha)\Phi$ remains undefined by this semantic rule.

Several further realization operators will prove useful. They are defined as follows. For a time span α, and a formula Φ:

$$RT(\alpha)\Phi ::= \forall t\ \alpha(t) \rightarrow R(t)\Phi$$

Reading: Φ is "realized throughout" time span α. Note: since time spans were defined as characteristic functions, the expression "$\alpha(t)$" evaluates True if time point t is in α.

$$RD(\alpha)\Phi ::= \exists\mu[PT(\mu,\alpha)\ \&\ RT(\mu)\Phi]$$

Reading: Φ is "realized during" time span α, i.e., it is realized throughout some sub-interval of α.

For a time point, γ, and a formula Φ,

$$RB(\gamma)\Phi ::= \exists\mu\ [\mu \leq\ \&\ RD(Span(\gamma,\mu))\Phi]$$

The above realization operators are "state oriented," i.e., they indicate something to be true at a particular point or span of time.

Another construct will allow us to describe *change*. One could describe change using the above constructs, e.g.,

$$R(t_0)\alpha\ \&\ R(t_1)\beta$$

where t_0 and t_1 are succeeding moments in time. However, often we will want to describe changes *generically*, without reference to the specific time when it occurred. For this we adopt a notation of von Wright (1965), where

$$(\alpha\ T\ \beta)$$

is read "α and then β." Here in CANDID, this will be defined essentially as a lambda abstraction on the preceding formula:

$$(\alpha \mathrm{T} \beta) ::= \lambda t_0 \exists t_1 \mathrm{NEXT}(t_0, t_1) \;\&\; \alpha(t_0) \;\&\; \beta(t_1).$$

It will be remembered that the set E was defined as consisting of physical objects existing in the past or present. However, it is often necessary to indicate just *when* a particular object is in exists. For that we need to adopt the predicate:

$$\mathrm{EXISTS}(\mu).$$

With the aid of the preceding realization operators, we can indicate whether an object existed at a particular time, e.g.,

RT(Yr(1980)) EXISTS(John)

indicates that John existed throughout the year 1980 (he may also have existed at other times as well). Birth and death are designated respectively as

$$\sim\mathrm{EXISTS}(x) \mathrm{\ T\ } \mathrm{EXISTS}(x)$$

$$\mathrm{EXISTS}(x) \mathrm{\ T\ } \sim\mathrm{EXISTS}(x).$$

One may then question how this differs from the existential quantifier, $\exists$ which is sometimes read as "there exists." Rescher and Urquhart (1971) offer the interpretation that the predicate, EXISTS, is one of "temporal existence." In our case this is merely a question of convenient translations of the two symbols. The existential quantifier refers to the inclusion of some individual in the model. The existence predicate, however, refers to relationship between this individual and points or spans of time.

I-J. POSSIBLE WORLDS, INTENSIONS

In the last section, we generalized the notion of denotation to depend not only on the model M = <D, C, N, T, F> and an assignment of values to variables, g, but also on the location of the expression in a time dimension.

In this section we generalize one final time on the notion of denotation, making it in addition dependent on its location in a so-called *possible world*. This concept has had a rich and not uncontroversial recent history in logic, philosophy and linguistics. The early Wittgenstein (1921) saw this as the key to the formalization of natural languages (later in life, after an immense following was pursuing his earlier work, he reversed this clain, (Wittgenstein 1953)).

Kripke (1963) used the concept of possible world to create a formal semantics for modal logic. On the one hand, mathematical logicians, e.g., Chang and Keisler (1973), Kalish et al. (1980), equate the notion with a model for a formal language (at least at the level of first order languages). On the other hand, linguists and philosophers, e.g., Cresswell (1973), Rescher (1975), seem to regard possible worlds more broadly, as a sort of gedanken experiments, not limited by the vocabulary of the language.

Our usage of possible worlds here will be more on the mathematical side, i.e., that a possible world is an alternative model.

Following Montague's notation, the collection of possible worlds will be designated by the set I, whose individuals are written as i, i', etc. in the metalanguage. Apart from the model M and assignment g, the denotation of an expression therefore depends on its location in a possible world, i, and a time j. The pair <i,j> is called an ***index***.

In our last formal summary, i.e., of the language L_t, the model consisted of the domain, D of individual entities, and F an interpretation function on D interpreting the predicate and operation constants as relations and functions on D. Since then we added the additional sets C (character strings), N (numbers) and T (times) to the model.

Our use of character strings and numbers was essentially an alternative to introducing more predicate names, e.g., Height(x) = 20 might be viewed as an abbreviation of HEIGHT-IS-20(x), and Last-Name(x) = "SMITH" might abbreviate LAST-NAME-SMITH(x).

Time, on the other hand, introduced a dimension on which the truth value denotations of an expression depended. I.e., for an expression Φ, $Den_{M,j,g}\Phi$ = True or False depending, inter alia, on the time j. Here M = <D, C, N, T, F>. The only one of these sets that varies with time is D, i.e., the set of individuals existing at or before time j. Correspondingly, the interpretation function, F, will also depend on the time, j, since while F includes relations in all the sets, the relations involving D will vary.

Thus, it is essentially only the pair <D,F> that vary with j. Here the changes in <D,F> as j increases might be viewed as all "due to natural causes," e.g., individuals are born and die, and single and sets of individuals change their properties.

The aspects of the model that vary between different possible worlds are also confined to the pair <D,F>. Here, however, the differences in <D,F> between one possible world and another are arbitrary. (There is no notion of adjacency between possible worlds as there is with times, since worlds are not ordered under "<", hence there is no basis for graduating differences.) Indeed, which we will continue to discuss the pair <D,F> as depending on a possible world i, though in an arbitrary way, in fact a possible world is *equivalent* to some arbitrarily chosen domain and interpretation function, i.e., some <D',F'>.

Thus, possible worlds and points in time determine a coordinate system on which <D,F> depends. Graphically, we might represent this for two possible worlds, i_1 and i_2, and three times, j_1, j_2, j_3, as follows:

$$\begin{array}{cccccccc}
 & & | & & | & & | & \\
i_1 & \text{------} & \langle D,F\rangle_{1,1} & \text{------} & \langle D,F\rangle_{1,2} & \text{------} & \langle D,F\rangle_{1,3} & \text{------} \\
 & & | & & | & & | & \\
i_2 & \text{------} & \langle D,F\rangle_{2,1} & \text{------} & \langle D,F\rangle_{2,2} & \text{------} & \langle D,F\rangle_{2,3} & \text{------} \\
 & & | & & | & & | & \\
 & & j_1 & & j_2 & & j_3 &
\end{array}$$

The purpose, for Montague, of this device is to explicitly represent what philosophers call the *intension* (spelled with an "s") of an expression. (Thus Montague's calculus is called "Intensional Logic").

Very briefly, it has long been recognized that the usual concept of denotation is insufficient to capture what we consider its complete meaning. (In informal usage, this residual part of meaning is often called its *connotation*. Intension and extension, as used here, are more technical terms corresponding to connotation and denotation, respectively.) Frege

(1893) captured the problem succinctly in his famous example of Morning Star and Evening Star: the two phrases denote the same thing, but they have somewhat different uses, hence different connotations or intensions.

More to the point of our interests is the problem of so called *opaque contexts*. In English these appear with such verbs as "believe,," "think," "imagine," etc. followed by the relative pronoun "that." (In Latin based languages these are the class of subjunctive constructions.) Consider the following example.

Let P = "the world is flat"

Q = "the moon is made of green cheese"

Suppose that an individual, John, believes P, i.e.,

BELIEVES(John,P).

The problem is that this would lead us to infer

BELIEVES(John,Q)

since A and B denote the same thing, namely False. However, we intuitively find it unacceptable to infer that if someone believes one false thing, he/she then believes every false thing.

As relates to the applications of CANDID, this same problem of opaque contexts arises in all types of commercial and financial contracts: if someone contracts to do *some* thing, that they are then obligated to do *every* thing.

The mechanism that Montague proposes to avoid this is his *intension operator*, $^\wedge$.

Effectively, this operates as an implicit lambda abstraction on indices (possible world, point in time pairs). Thus for instance if $\Phi \in ME_v$, $^\wedge\Phi$ refers to the set of tuples of the form $<<i,j>,v>$, i.e., evaluating the proposition Φ at every index. Cresswell (1973:23-24) offers an intuitive motivation of what this provides

> If we think for a moment of the job a proposition has to do we see that it must be something which can be true or false, not only in the actual world but in each possible world. Suppose for the moment that we could "show" a person all possible worlds in turn. This is of course impossible, but try to imagine it anyway. We want to know whether two people are thinking of the same proposition. So we ask them, as we show them each (complete) possible world. "Would the proposition you are thinking of be true if that was the way things were?" If their answers agree for every possible world there is at least the temptation to suppose that they have the same proposition in mind. Or to put it another way, if the set of worlds to which *A* says "yes" is the same as the set of worlds to which *B* says "yes" we can say that *A* and *B* have the same proposition in mind. So why not simply identify the proposition with the set of worlds in question? As a first approximation therefore we shall say that a proposition is a set of possible worlds.

Thus, with reference to our previous example, we would avoid the erroneous substitution by writing

$$\text{BELIEVES}(\text{John}, {}^\wedge P)$$

Since there are conceivable possible worlds in which P is true and Q false, or vice versa, $^\wedge P \neq {}^\wedge Q$, even though both P and Q are false in the actual world.

The converse of the intension operator is written "$^\vee$"; i.e., $^\vee\Phi$ is the application of the intension Φ to the actual world. Hence,

$$^\vee[^\wedge\Phi] = \Phi$$

This latter notation will however be of lesser importance for our applications.

As seen in the above discussion, intension and extension are interrelated concepts. Further, extension corresponds to what we have heretofore called denotation. In keeping with the terminology of Montague (and Dowty), we will switch to the abbreviation "Ext" (for extension) rather than "Den" in our semantic rules. Correspondingly, the new abbreviation, "Int" (for intension) will be introduced.

Let us now summarize the formal language as it stands thus far.

I-K. SUMMARY OF THE LANGUAGE IL

Syntax of IL

Corresponding to each type a, the intension of that type will be a new type, written <s,a> (where s stands for "sense"—Frege's original term for "intension," which was introduced by Carnap.) The "s" may be read as an abbreviation for the <i,j>. Hence <s,a> abbreviates <<i,j>,a>. The set of types (i.e., Type) is defined recursively as follows:

e is a type

c is a type

n is a type

t is a type

if a and b are any types, then <a,b> is a type

if a is any type, then <s,a> is a type.

The basic expressions of IL consist of:

1) for each type a, the set of *constant of type a*, denoted Con_a. (Names for particular constants follow the conventions described earlier.)

2) for each type a, the set of *variables of type a*, denoted Var_a. (Names for variables are as before.)

3) *terms of type a*, denoted $Term_a$, defined recursively as follows:

   — if $\alpha \in Con_a$ then $\alpha \in Term_a$.

   — if $\alpha \in Var_a$ then $\alpha \in Term_a$

   — if $\beta_2, \ldots, \beta_n$ are terms of types $x_1,\ldots,x_n$ respectively and Φ is an operation of type $<x_1,\ldots,<x_n,a>>$ then $\Phi(\beta_1, \ldots, \beta_n) \in Term_a$.

   — if $\mu \in Var_a$ and $\Phi \in ME_{<a,t>}$, whose only unbound variable is μ, then $\iota\mu\Phi \in Term_a$.

The set of meaningful expressions of type a, denoted ME_a, is defined recursively as follows:

1. Every term of type a is in ME_a.

2. If $\alpha \in ME_{<a,b>}$ and $\beta \in ME_a$, then $\alpha(\beta) \in ME_b$.

3. If $\alpha \in ME_a$ and μ is a variable of type b, then $\lambda\mu\alpha \in ME_{<b,a>}$.

4. If α and β are both in ME_a, then $[\alpha=\beta] \in ME_v$.

5. If $\Phi \in ME_{<a,t>}$ and $\mu \in Var_a$ then $\iota\,\mu\,\Phi \in ME_a$.

6-10. If Φ and Ψ are in ME_v, then so are

6. $[{\sim}\Phi]$

7. $[\Phi \,\&\, \Psi]$

8. $[\Phi \vee \Psi]$

9. $[\Phi \rightarrow \Psi]$

10. $[\Phi \leftrightarrow \Psi]$

11-12. If $\Phi \in ME_v$ and μ is a variable (of any type) then

11. $\forall\mu\,\Phi \in ME_v$

12. $\exists\mu\,\Phi \in ME_v$

13. If $\mu \in Var_t$ and $\Phi \in ME_v$ then $R\,\mu\,\Phi \in ME_v$

14. If Φ and Ψ are in ME_v, then $\Phi\,T\,\Psi \in ME_v$

15. If $\Phi \in ME_a$ then ${}^{\wedge}[\Phi] \in ME_{<s,a>}$

16. If $\Phi \in ME_{<s,a>}$ then ${}^{\vee}\Phi \in ME_a$.

Semantics of IL

A model for IL is the ordered tuple M = <D, C, N. T, W, F> where D, C, N, T and W are non-empty sets, that assigns an appropriate denotation to each (non-logical) constant of IL relative to each pair <i,j>, for $i \in W$ and $j \in T$. (Thus "$F(<i,j>,\alpha) = \gamma$" asserts that the denotation of the constant α in the possible world i at time j is the object γ.)

The set of *possible denotations* of type a, written D_a, is define as follows (where a and b are any types):

$D_e = D$

$D_c = C$

$D_n = N$

$D_t = T$

$D_v = \{True, False\}$

$D_{<a,b>} = D_b^{D_a}$

$D_{<s,a>} = D_a^{WXT}$ (where WXT is the set of all world, time point pairs, i.e., the set of all indices <i,j>).

Semantic Rules

The semantic rules of IL define recursively for any expression α, the *extension* of α with respect to model N, $i \in W$, $j \in T$ and value assignment g, written $Ext_{M,i,j,g}(\alpha)$ as follows:

1. If α is a non-logical constant then $Ext_{M,i,j,g}(\alpha) = F(\alpha)$ (<i,j>), (i.e., the extension of α at <i,j> is simply the result of applying the intension of α, which is supplied by F, to <i,j>).
2. If α is a variable, then $Ext_{M,i,j,g}(\alpha) = g(\alpha)$
3. If $\alpha \in ME_a$ and μ is a variable of type b, then $Ext_{M,i,j,g}$ $(\lambda\mu\alpha)$ is that function h with domain D_b such that for any object x in that domain, $h(x) = Ext_{M,i,j,g'}(\alpha)$, where g' is that value assignment exactly like g with the possible difference that $g'(\mu)$ is the object x.
4. If $\alpha \in ME_{<a,b>}$ and $\beta \in ME_a$, then $Ext_{M,i,j,g}(\alpha)(Ext_{M,i,j,g}(\beta))$ (i.e., the result of applying the function $Ext_{M,i,j,g}(\alpha)$ to the argument $Ext_{M,i,j,g}(\beta)$).

5. If α and β are in ME_a, then $Ext_{M,i,j,g}(\alpha=\beta)$ is True if and only if $Ext_{M,i,j,g}(\alpha)$ is the same as $Ext_{M,i,j,g}(\beta)$.

6. If $\Phi \in ME_t$, then $Ext_{M,i,j,g}(\sim\Phi)$ is True if and only if $Ext_{M,i,j,g}(\Phi)$ is False, and $Ext_{M,i,j,g}(\sim\Phi)$ is False otherwise.

7. If Φ and Ψ are in ME_v, then $Ext_{M,i,j,g}[\Phi \ \& \ \Psi]$ is True if and only if both $Ext_{M,i,j,g}(\Phi)$ and $Ext_{M,i,j,g}(\Psi)$ are True.

8. $Ext_{M,i,j,g}[\Phi \vee \Psi]$ is True if and only if either $Ext_{M,i,j,g}[\Phi]$ is True or $Ext_{M,i,j,g}[\Psi]$ is True. $Ext_{M,i,j,g}[\Phi \rightarrow \Psi]$ is True if and only if either $Ext_{M,i,j,g}[\Phi]$ is False or $Ext_{M,i,j,g}[\Psi]$ it True.

10. $Ext_{M,i,j,g}[\Phi \leftrightarrow \Psi]$ is True if and only if either both $Ext_{M,i,j,g}[\Phi]$ and $Ext_{M,i,j,g}[\Psi]$ are both True or are both False.

11. If $\Phi \in ME_v$ and μ is a variable of type e, then $Ext_{M,i,j,g}(\forall\mu\Phi)$ is True if and only if $Ext_{M,i,j,g}(\Phi)$ is True for all g' exactly like g except possibly for the value assigned to μ.

12. If $\Phi \in ME_v$ and μ is a variable, then $Ext_{M,i,j,g}(\exists\mu\Phi)$ is True if and only if $Ext_{M,i,j,g'}(\Phi)$ is True for some value assignment g' exactly like g except for the value assigned to μ.

13. For α a term of type t, and $\Phi \in ME_v$, $Ext_{M,i,j,g}(R\ \alpha\ \Phi)$ = True iff for some j', j'=α and j'<j, $Ext_{M,i,j',g}(\Phi)$ = True.

14. For Φ and Ψ in ME_v, $Ext_{M,i,j,g}(\Phi\ T\ \Psi)$ = True iff $Ext_{M,i,j',g}(\Phi)$ = True and $Ext_{M,i,j'',g}(\Psi)$ = True for some j' and j'' such that j' immediately precedes j''.

15. If $\alpha \in ME_a$, then $Ext_{M,i,j,g}(^\wedge\alpha)$ is that function h with domain WXT such that for all <i',j'> in WXT, h(<i',j'>) is $Ext_{M,i',j',g}(\alpha)$.

16. If $\alpha \in ME_{<s,a>}$, then $Ext_{M,i,j,g}({}^{V}\alpha)$ is $Ext_{M,i,j,g}(\alpha)$ (<,i,j>) (i.e., the result of applying the function $Ext_{M,i,j,g}$ (α) to the argument <i,j>).

Additional Primitive and Defined Predicates and Operations

For Domain C (character strings)

None

For Domain N (numbers)

1. type: <n,<n,v>>

$[\alpha < \beta]$	(primitive)
$[\alpha \leq \beta]$ ::=	$[\alpha < \beta] \vee [\alpha = \beta]$
$[\alpha > \beta]$ ::=	$\sim[\alpha \leq \beta]$
$[\alpha \geq \beta]$ ::=	$[\alpha > \beta] \vee [\alpha = \beta]$
$[\alpha \neq \beta]$ ::=	$\sim[\alpha = \beta]$

2. type: <n,<n,n>>

$[\alpha + \beta]$	(primitive)
$[\alpha - \beta]$	(primitive)
$[\alpha * \beta]$	(primitive)
$[\alpha / \beta]$	(primitive)
$[\alpha ** \beta]$	(primitive)

For Domain T (times)

1. type: $<t,<t,v>>$

   $<, \leq, >, \geq, \neq$ (defined as for numbers,)

   $NEXT(\alpha,\beta) ::= \alpha<\beta \;\&\; \forall t[[T\neq\alpha \;\&\; [t \neq \beta] \rightarrow {}^{\sim}[\alpha < t < \beta]]$

2. type: $<t,<t<t,v>>$

   $Span ::= \lambda x\, \lambda y\, \lambda z\, [z >= x \;\&\; z \leq y]$

3. type: $<<t,v>,t>$

   $Beg ::= \lambda x\, \iota y\, \exists z\, [x = Span(y,t)]$

   $End ::= \lambda x\, \iota z\, \exists y\, [x = Span(y,z)]$

4. type: $<<t,v><<t,v>,v>$

   $PT ::= \lambda x\, \lambda y\, [Beg(x) \geq Beg(y) \;\&\; End(x) \leq End(y)]$

5. type: $<n,<n,<n,<t,v>>>>$

   $Date(\alpha,\beta,\gamma)$ (primitive)

6. type: $<n,<n,<t,v>>$

7. type: $<n,<t,v>>$

   $Yr(\alpha)$ (primitive)

8. type: $<<t,v>,<<t,v>,n>>$

   $Dur(\alpha,\beta)$ (primitive)

9. type: $<t,v>$

   Day (primitive)

10. type: $<<t,v>,<v,v>>$

    $RT(\alpha)\Phi ::= \forall t\; \alpha(t) \rightarrow R(t)\; \Phi$

    $RD(\alpha)\; \Phi ::= \exists \mu\; PT(\mu,\alpha) \;\&\; RT(\mu)$

11. type: <<t,<v,v>>

$$RB(\gamma)\ \Phi ::= \exists\mu\ [\mu \leq \gamma\ \&\ RD(Span(\gamma,\mu))\Phi]$$

I-L. ACTION

Earlier, the connective T, a construct due to von Wright (1965), was introduced in order to describe generic changes. We now follow von Wright's development further to obtain a description of *actions*.

Von Wright introduces another connective, I, with a syntax like that of T, i.e.,

Syn. If Φ and Ψ are in ME_v, then $\Phi\ I\ \Psi \in ME_v$

This connective has the reading "instead of." Its effect is that, due to the intercession of some agent, Φ is true instead of Ψ being true.

As von Wright points out, I serves as to coordinate two possible worlds.

Interpreting von Wright's sense for I in the Montague framework as we have developed it so far we have:

Sem. For Φ and Ψ in ME_v, then $Ext_{M,i,j,g}(\Phi$ = True iff $Ext_{M,i,j,g}(\Phi)$ = True and $Ext_{M,i',j,g}(\Psi)$ = True for some i' just like i except that i' lacks the interference of some agent.

We extend von Wright's notation slightly by adding a place in the connection specifying the agent. Thus,

Syn'. If Φ and Ψ are in ME_v and α is a term of type e, then $[\Phi\ I\alpha\ \Psi] \in ME_v$.

The corresponding semantic rule is as follows:

Sem'. For Φ and Ψ in ME_v, and α a term of type e, then $Ext_{M,i,j,g}[\Phi \, I\alpha \, \Psi]$ = True iff $Ext_{M,i,j,g}(\Phi)$ = True and $Ext_{M,i',j,g}(\Psi)$ = True for some i' just like i except that i' lacks the interference of agent α.

This concept of "interference" is admittedly, rather uncomfortable. If we compare the models <D, F> and <D', F'> of i and i' respectively, what is different about them? Precisely that Φ is true in the first, and Ψ is true in the second. This *is* the interference.

The I connective combines with T to form what von Wright calls "TI expressions." It is these expressions which are used to express actions, i.e.,

$$\alpha \, T \, (\beta \, I\mu \, \gamma)$$

is read: "α is true and then β is true instead of γ due to the interference of μ."

For instance, if the action is for John to open a window, we would have

CLOSED(Window) T (OPEN(WINDOW) I(John) CLOSED(Window))

i.e., the window was closed and then it was open instead of remaining closed due to the interference of John.

I-M. MODALS, DEONTIC OPERATORS

Montague's Intentional Logic includes the modal operators $\diamond$ and $\Box$, for possibility and necessity, respectively, by means of the following syntactic and semantic rules.

Syn.1. If $\Phi \in ME_v$, then $\diamond\Phi \in ME_v$.

Syn.2. If $\Phi \in ME_v$, then $\Box\Phi \in ME_v$.

Sem.1. For $\Phi\& \in ME_v$

$Ext_{M,i,j,g}(\diamond\Phi)$ = True iff $Ext_{M,i',j',g}(\Phi)$ = True for some i' in W and some j' in T.

Sem.2. For $\Phi \in ME_v$

$Ext_{M,i,j,g}\Box\Phi$ = True iff $Ext_{M,i',j',g}\Phi$ = True for all i' in W and all j' in T.

Thus $\diamond\Phi$ indicates that Φ is possibly true, i.e., it is true in some possible world at some time. Correspondingly $\Box\Phi$ indicates Φ to be necessarily true, i.e., true in all possible worlds at all times.

Either one of these rules could have been omitted, recognizing that the two concepts are inter-definable, i.e.,

$$\Box\Phi ::= {}^{\sim}\diamond{}^{\sim}\Phi$$

or

$$\diamond\Phi ::= {}^{\sim}\Box{}^{\sim}\Phi.$$

(This follows from the inter definability of the quantifiers $\forall$and $\exists$ implicit in the semantic interpretation of these operators.)

Von Wright points out that this is only one version of necessity (possibility), what he calls logical necessity (possibility). That is, if $\Box\Phi$, then Φ is true by virtue of the interaction of the truth assignments of its composite formulae, independent of what these formulae denote—i.e., Φ is true in all possible models. Alternative terminology is that Φ is a tautology or that Φ is analytically true.

Another version of necessity (possibility) is what von Wright calls *natural* necessity (possibility). We write this as

$$\Box_N\Phi$$

and

$$\Diamond_N\Phi.$$

If $\Box_N\Phi$ is true, then Φ is true in all worlds and at all times "because of the way the world operates." Natural necessity is stronger than logical necessity. For instance, "if x is a human, then x is warm blooded" is a natural necessity, though not a logical one.

In order to portray natural necessity or possibility in our semantic framework, we would need to qualify certain possible worlds as being "natural," i.e., conforming to the laws of nature. Call this the set W_N such that $W_N \subseteq W$. The syntactic and semantic rules would therefore be as follows:

Syn'1. If $\Phi \in ME_v$ then $\Diamond_N\Phi \in ME_v$.

Syn'2. If $\Phi \in ME_v$ then $\Box_N\Phi \in ME_v$.

Sem'1. For $\Phi \in ME_{\upsilon}$, $Ext_{M,i,j,g}(\Diamond_N\Phi)$ = True iff $Ext_{M,i',j',g}\Phi$ = True for some $i' \in W_N$ and some $j' \in T$.

Sem'2. For $\Phi \in ME_{\upsilon}$, $Ext_{M,i,j,g}\Box_N\Phi$ = True iff $Ext_{M,i',j',g}(\Phi)$ = True for all $i' \in W_N$ and all $j' \in T$.

The logical duality of these concepts again holds, i.e.,

$$\Box_N\Phi ::= \sim\Diamond_N\sim\Phi$$

or

$$\Diamond_N\Phi ::= \sim\Box_N\sim\Phi.$$

Von Wright extends this one step further to address the concepts of permission and obligation, which he calls the *deontic* modalities. We will abbreviate these as

$\Diamond_D\Phi$ for Φ is permitted

and

$\Box_D\Phi$ for Φ is obligatory.

(Von Wright uses the notation P and O here, but we reserve that for later uses.)

To describe this in our semantic framework, we need to further qualify certain possible worlds as being legitimate within a general ethical or legal code. (Given that there are numerous such codes, e.g., for different countries, there are correspondingly different definitions of permission and obligation. We ignore this aspect for present purposes.)

Let me denote the set of deontically permissible worlds as W_D, where

$$W_D \subseteq W.$$

The syntactic and semantic rules are of similar form:

Syn''1. If $\Phi \in ME_v$ then $\Diamond_D\Phi \in ME_v$.

Syn''2. If $\Phi \in ME_v$ then $\Box_D\Phi \in ME_v$.

Sem''1. For $\Phi \in ME_v$ then $Ext_{M,i,j,g}(\Diamond_D\Phi)$ = True iff $Ext_{M,i',j',g}(\Phi)$ = True for some $i' \in W_D$ and some $j' \in T$.

Sem''2. For $\Phi \in ME_v$ then $Ext_{M,i,j,g}(\Box_D\Phi$ = True iff $Ext_{M,i',j',g}(\Phi)$ = True for all $i' \in W_D$ and all $j' \in T$.

Once again, these are logical duals:

$$\Box_D\Phi ::= {\sim}\Diamond_D{\sim}\Phi$$

or

$$\Diamond_D\Phi ::= {\sim}\Box_D{\sim}\Phi.$$

That is, if something is obligatory, it is not permissible not to do it. Contrariwise, if something is permitted, it is not obligatory not to do it. Prohibition, i.e., that something is forbidden, is a deontic impossibility, i.e., the negation of permissibility:

$${\sim}\Diamond_D\Phi$$

says it is not permitted (forbidden) to do Φ.

It is often argued that "ought" implies "can"—i.e., that if something is obligatory then it should be naturally possible. This would be reflected in the assumption:

$$W_D \subseteq W_N \subseteq W.$$

The deontic modalities differ from the other in that they generally apply only to actions. This entails that the formula Φ be a TI expression. We would therefore write

$$\Diamond_D \alpha T [\beta I \mu \gamma]$$

to indicate that μ is permitted to bring about β (from the previous state α, instead of allowing γ to occur), and

$$\Box_D \alpha T [\beta I \mu \gamma]$$

to indicate that μ is obliged to bring about β.

Note on Contingent Obligations, Permissions

A contingent obligation (permission) is where an action Φ is obligatory (permitted) if Ψ occurs. Considering first the case of contingent obligation, there seems to be two possible representations:

a) $\Box_D [\Psi \rightarrow \Phi]$ (it is obligatory that if Ψ then Φ)

b) $\Psi \rightarrow \Box_D \Phi$ (if Ψ then it is obligatory to Φ).

The English reading in these two cases does little to help choose between them—both readings seem adequate.

However, if we examine the semantic interpretations in both cases there is an important difference. We have

Sem_a $\quad Ext_{M,i,j,g}\Box_D[\Psi \rightarrow \Phi] = True$ iff $Ext_{M,i',j',g}(\Psi \rightarrow \Phi) = True$ for all $i' \in W_D$ and $j' \in T$.

Sem$_b$ $\quad Ext_{M,i,j,g}\ \Psi \rightarrow \Box_D\Phi$ = True iff action $Ext_{M,i,j,g}(\Psi)$ = False or $Ext_{M,i',j',g}(\Phi)$ = True for all $i' \in W_D$ and $j' \in T$.

In case a, $\Psi \rightarrow \Phi$ must be true at all indices involving permissible worlds (i.e., elements of W_D). In case b, if Ψ is true at the current index, then Φ must be true at all indices involving permissible worlds. The point is that in the second case, Ψ and Φ do not necessarily apply to the same possible world. Thus if Ψ were not true at the current index but were true at some other index involving a permissible world, Φ would not necessarily hold at this other index.

This problem is however avoided in case a, and is thus the preferred method of representing contingent obligation. Analogous arguments hold in the case of contingent permission.

Contractual Obligation and Permission

The concepts of obligation and permission discussed thus far pertain to the structure of a general ethical or legal code. In the case of contracts we are concerned with obligation and permission at a more specific level—e.g., x is obliged *to y* to do Φ or x is permitted *by y* to do Φ.

Our view here is that these specific obligations and permissions depend on protection under the general legal system in force. We regard this protection to be in the form of the possibility of taking legal action if the terms of the contract are violated. We abbreviate party x taking legal action against party y as LA(x,y).

Thus, our interpretation of x's obligation to y to do Φ is that y is permitted to take legal action against x if Φ does not occur.

We abbreviate this as follows:

$$O(x,y)\Phi ::= \Diamond_D {\sim}\Phi \rightarrow LA(y,x)$$

The symbol "$O(x,y)\Phi$" will be read "x has the obligation to y to see to it that Φ." Usually, x will be an agent of an action contained in Φ, though this is not required. For instance, Φ might be performed by someone else sub-contracted by x.

The concept of contractual permission is slightly less direct than for contractual obligation. We will use the notation

$$P(x,y)\Phi$$

to indicate that "x permits y to bring about Φ." We begin with the observation that this generally presupposes that y would otherwise be prohibited from doing (bringing about) Φ which is to say that x would be permitted legal action against y if y did Φ.

Thus, by granting permission to y to do Φ, x foregoes this right to take legal action. In symbolic form this is summarized as follows:

$$P(x,y)\Phi ::= {\sim}[\Diamond_D\Phi \rightarrow LA(x,y)]$$

Reading: that x permits y to do Φ is defined as that it is not permitted for x to take legal action against y if Φ.

In the preceding section we saw that the various forms of modal operators, including the deontic modals, were logical duals of one another. This is also the case with contractual obligation and permission as we have defined it—however, with one interesting difference: the order of the agent and recipient places is reversed in the dual form. Thus,

$$
\begin{aligned}
P(x,y)\Phi \quad & ::= {\sim}O(y,x)\ {\sim}\Phi \\
& ::= {\sim}[\Diamond_D {\sim}({\sim}\Phi \rightarrow LA(x,y)] \\
& ::= {\sim}[\Diamond_D \Phi \rightarrow LA(x,y)]
\end{aligned}
$$

Reading: x permits y to Φ is defined that, y is not obligated to x not to Φ. The subsequent substitutions lead to the definition of contractual permission given previously.

One additional aspect needs to be considered. In the contracts discussed so far, the enforcement of the contract was an implicit recourse to legal action. However, in certain contracts this enforcement is made explicit in the form of a penalty clause indicating some other action to be taken. We will indicate such penalty clauses by adding an additional place in the contractual obligation and permission operators, separated by a "/".* Thus

$$O(x,y)\ \Phi/\Psi ::= \Diamond_D\, [{\sim}\Phi \rightarrow \Psi].$$

The previous syntax is thus a special case of this, where $\Psi = LA(y,x)$:

$$
\begin{aligned}
O(x,y)\ \Phi \quad & ::= O(x,y)\ \Phi/LA(y,x) \\
& ::= \Diamond_D {\sim}\Phi \rightarrow LA(y,x).
\end{aligned}
$$

While explicit penalty clauses are fairly common in the case of contractual obligation, they are rare for contractual permission. Nevertheless, for the sake of completeness and symmetry, we offer the following defini-

* Note: von Wright also uses a slash notation resembling this, but with a different interpretation.

tion:

$$P(x,y)\ \Phi/\Psi ::= {\sim}\Diamond_D\,[\Phi \rightarrow \Psi]$$

Letting Ψ be LA(x,y), the earlier definition follows as a special case:

$$P(x,y)\ \Phi) \quad ::= P(x,y)\ \Phi/LA(x,y)$$

$$::= {\sim}\ \Diamond_D\,[\Phi \rightarrow LA(x,y)]$$

I-N. SUMMARY OF THE LANGUAGE CANDID

Syntax of CANDID

Corresponding to each type a, the intension of that type will be written <s,a>. The set of types (i.e., Type) is defined recursively as follows:

e is a type

c is a type

n is a type

t is a type

if a and b are any types, then <a,b> is a type

if a is any type, then <s,a> is a type.

The basic expressions of CANDID consist of:

1) for each type a, the set of *constants of type a*, denoted Con_a. (Names for particular constants follow the conventions described earlier.)

2) for each type a, the set of *variables of type a*, denoted Var_a. (Names for variables are as before.)

3) ***terms of type a***, denoted $Term_a$, defined recursively as follows:

— if $\alpha \in Con_a$ then $\alpha \in Term_a$.

— if $\alpha \in Var_a$ then $\alpha \in Term_a$

— if $\beta_2, \ldots, \beta_n$ are terms of types $x_1,\ldots,x_n$ respectively and Φ is an operation of type $<x_1,\ldots,<x_n,a>>$ then $\Phi(\beta_1, \ldots, \beta_n \in Term_a$.

— if $\mu \in Var_a$ and $\Phi \in ME_{<a,t>}$, whose only unbound variable is μ, then $\iota u\Phi \in Term_a$.

Formation Rules of CANDID

The set of meaningful expressions of type a, denoted ME_a, is defined recursively as follows:

1. Every term of type a is in ME_a.
2. If $\alpha \in ME_{<a,b>}$ and $\beta \in ME_a$, then $\alpha(\beta) \in ME_b$.
3. If $\alpha \in ME_a$ and μ is a variable of type b, then $\lambda u\alpha \in ME_{<b,a>}$.
4. If α and β are both in ME_a, then $[\alpha=\beta] \in ME_v$.
5. If $\Phi \in ME_{<a,t>}$ and $\mu \in Var_a$ then $\iota\, \mu\, \Phi \in ME_a$.

6-10. If Φ and Ψ are in ME_v, then so are

6. $[\sim\Phi]$
7. $[\Phi \,\&\, \Psi]$
8. $[\Phi \vee \Psi]$
9. $[\Phi \rightarrow \Psi]$

10. $[\Phi \leftrightarrow \Psi]$

11-12. If $\Phi \in ME_v$ and μ is a variable (of any type) then

11. $\forall\mu\, \Phi \in ME_v$

12. $\exists\mu\, \Phi \in ME_v$

13. If $\mu \in Var_t$ and $\Phi \in ME_v$ then $R\, \mu\, \Phi \in ME_v$

14. If Φ and Ψ are in ME_v, then $\Phi\ T\ \Psi \in ME_v$

15. If $\Phi \in ME_a$ then $^{\wedge}\Phi \in ME_{<s,a>}$

16. If $\Phi \in ME_{<s,a>}$ then $^{\vee}\Phi \in ME_a$.

17. If Φ and Ψ are in ME_v and α is a term of type e, then $[\Phi\ I\alpha\ \Psi] \in ME_v$.

18,19. If $\Phi \in ME_v$ then

18. $\Diamond_D\Phi \in ME_v$

19. $\Box_D\Phi \in ME_v$

Semantics of CANDID

A model for CANDID is the ordered tuple M = <D, C, N. T, W, F> where D, C, N, T and W are non-empty sets, that assigns an appropriate denotation to each (non-logical) constant of CANDID relative to each pair <i,j>, for $i \in W$ and $j \in T$. (Thus "$F(<i,j>,\alpha) = \gamma$" asserts that the denotation of the constant α in the possible world i at time j is the object γ.)

The set of ***possible denotations*** of type a is defined as follows (where a and b are any types):

$D_e = D$

$D_e = C$

$D_n = N$

$D_t = T$

D_v = {True, False}

$D_{<a,b>} = D_b^{D_a}$

$D_{<s,a>} = D_a^{WXT}$ (where WXT is the set of all world, time point pairs, i.e., the set of all indices <i,j>.

Semantic Rules

The semantic rules of CANDID define recursively for any expression α, the extension of α with respect to model N, $i \in W$, $j \in T$ and value assignment g, written $Ext_{M,i,j,g}(\alpha)$ as follows:

1. If α is a non-logical constant then $Ext_{M,i,j,g}(\alpha) = F(\alpha)$ (<i,j>), (i.e., the extension of α at <i,j> is simply the result of applying the intension of α, which is supplied by F, to <i,j>).

2. If α is a variable, then $Ext_{M,i,j,g}(\alpha) = g(\alpha)$

3. If $\alpha \in ME_a$ and μ is a variable of type b, then $Ext_{M,i,j,g}$ $(\lambda u \alpha)$ is that function h with domain D_b such that for any object x in that domain, $h(x) = Ext_{M,i,j,g'}(\alpha)$, where g' is that value assignment exactly like g with the possible difference that $g'(\mu)$ is the object x.

4. If $\alpha \in ME_{<a,b>}$ and $\beta \in ME_a$, then $Ext_{M,i,j,g}(\alpha)(Ext_{M,i,j,g}(\beta))$ (i.e., the result of applying the function $Ext_{M,i,j,g}(\alpha)$ to the argument $Ext_{M,i,j,g}(\beta)$).

5. If α and β are in ME_a, then $Ext_{M,i,j,g}(\alpha=\beta)$ is True if and only if $Ext_{M,i,j,g}(\alpha)$ is the same as $Ext_{M,i,j,g}(\beta)$.

6. If $\Phi \in ME_t$, then $Ext_{M,i,j,g}(\sim\Phi)$ is True if and only if $Ext_{M,i,j,g}(\Phi$ is False, and $Ext_{M,i,j,g}(\sim\Phi)$ is False otherwise.

7. If Φ and Ψ are in ME_v, then $Ext_{M,i,j,g}[\Phi \ \& \ \Psi]$ is True if and only if both $Ext_{M,i,j,g}(\Phi)$ and $Ext_{M,i,j,g}(\Psi)$ are True.

8. $Ext_{M,i,j,g}[\Phi \vee \Psi]$ is True if and only if either $Ext_{M,i,j,g}[\Phi]$ is True or $Ext_{M,i,j,g}[\Psi]$ is True. $Ext_{M,i,j,g}[\Phi \rightarrow \Psi]$ is True if and only if either $Ext_{M,i,j,g}[\Phi]$ is False or $Ext_{M,i,j,g}[\Psi]$ it True.

10. $Ext_{M,i,j,g}[\Phi \leftrightarrow \Psi]$ is True if and only if either both $Ext_{M,i,j,g}[\Phi]$ and $Ext_{M,i,j,g}[\Psi]$ are both True or are both False.

11. If $\Phi \in ME_v$ and μ is a variable of type e, then $Ext_{M,i,j,g}$ $(\forall\mu\Phi)$ is True if and only if $Ext_{M,i,j,g}(\Phi)$ is True for all g' exactly like g except possibly for the value assigned to μ.

12. If $\Phi \in ME_v$ and μ is a variable, then $Ext_{M,i,j,g}(\exists\mu\Phi)$ is True if and only if $Ext_{M,i,j,g'}(\Phi)$ is True for some value assignment g' exactly like g except for the value assigned to μ.

13. For α a term of type t, and $\Phi \in ME_v$, $Ext_{M,i,j,g}(R\ \alpha\ \Phi)$ = True iff for some j', j'+α and j'<j, $Ext_{M,i,j',g}(\Phi)$ = True.

14. For Φ and Ψ in ME_v, $Ext_{M,i,j,g}(\Phi\ T\ \Psi)$ = True iff $Ext_{M,i,j',g}(\Phi)$ = True and $Ext_{M,i,j'',g}(\Psi)$ = True for some j' and j'' such that j' immediately precedes j''.

15. If $\alpha \in ME_a$, then $Ext_{M,i,j,g}(^\wedge\alpha)$ is that function h with domain WXT such that for all <i',j'> in WXT, h(<i',j'>) is $Ext_{M,i',j',g}(\alpha)$.

16. If $\alpha \in ME_{<s,a>}$, then $Ext_{M,i,j,g}({}^{V}\alpha)$ is $Ext_{M,i,j,g}(\alpha)$ (<,i,j>) (i.e., the result of applying the function $Ext_{M,i,j,g}$ (α) to the argument <i,j>).

17. For Φ and Ψ in ME_v and α a term of type e, then $Ext_{M,i,j,g}[\Phi \; I\alpha \; \Psi]$ = True iff $Ext_{M,i,j,g}(\Phi)$ = True and $Ext_{M,i',j,g}(\Psi)$ = True for some i' just like i except that i' lacks the interference of agent α.

18. For $\Phi \in ME_v$ then $Ext_{M,i,j,g}(\Diamond_D \Phi)$ = True iff $Ext_{M,i',j',g}(\Phi)$ = True for some $i' \in W_D$ and some $j' \in T$.

19. For $\Phi \in ME_v$ then $Ext_{M,i,j,g}(\Box_D \Phi$ = True iff $Ext_{M,i',j',g}(\Phi)$ = True for all $i' \in W_D$ and all $j' \in T$.

Additional Primitive and Defined Predicates and Operations

For Domain C (character strings)

None

For Domain N (numbers)

1. type: <n,<n,v>>

$[\alpha < \beta]$	(primitive)
$[\alpha \leq \beta]$ ::=	$[\alpha < \beta] \vee [\alpha = \beta]$
$[\alpha > \beta]$ ::=	$\sim[\alpha \leq \beta]$
$[\alpha \geq \beta]$ ::=	$[\alpha > \beta] \vee [\alpha = \beta]$
$[\alpha \neq \beta]$ ::=	$\sim[\alpha = \beta]$

2. type: <n,<n,n>>

   $[\alpha + \beta]$ (primitive)

   $[\alpha - \beta]$ (primitive)

   $[\alpha * \beta]$ (primitive)

   $[\alpha / \beta]$ (primitive)

   $[\alpha ** \beta]$ (primitive)

For Domain T (times)

1. type: <t,<t,v>>

   $<, \leq, >, \geq, \neq$ (defined as for numbers,)

   $\mathrm{NEXT}(\alpha,\beta) ::= \alpha<\beta \;\&\; \forall t[[T\neq\alpha \;\&\; [t \neq \beta] \rightarrow {}^{\sim}[\alpha < t < \beta]]$

2. type: <t,<t<t,v>>

   $\mathrm{Span} ::= \lambda x\, \lambda y\, \lambda z\, [z >= x \;\&\; z \leq y]$

3. type: <<t,v>,t>

   $\mathrm{Beg} ::= \lambda x\, \iota y\, \exists z\, [x = \mathrm{Span}(y,t)]$

   $\mathrm{End} ::= \lambda x\, \iota z\, \exists y\, [x = \mathrm{Span}(y,z)]$

4. type: <<t,v><<t,v>,v>

   $\mathrm{PT} ::= \lambda x\, \lambda y\, [\mathrm{Beg}(x) \geq \mathrm{Beg}(y) \;\&\; \mathrm{End}(x) \leq \mathrm{End}(y)]$

5. type: <n,<n,<n,<t,v>>>>

   $\mathrm{Date}(\alpha,\beta,\gamma)$ (primitive)

6. type: <n,<n,<t,v>>

7. type: <n,<t,v>>

   $\mathrm{Yr}(\alpha)$ (primitive)

8. type: $<<t,v>,<<t,v>,n>>$

$\mathrm{Dur}(\alpha,\beta)$ (primitive)

9. type: $<t,v>$

Day (primitive)

10. type: $<<t,v>,<v,v>>$

$\mathrm{RT}(\alpha)\Phi ::= \forall t\ \alpha(t) \rightarrow \mathrm{R}(t)\ \Phi$

$\mathrm{RD}(\alpha)\ \Phi ::= \exists \mu\ \mathrm{PT}(\mu,\alpha)\ \&\ \mathrm{RT}(\mu)$

11. type: $<<t,<v,v>>$

$\mathrm{RB}(\gamma)\ \Phi ::= \exists \mu\ [\mu \leq \gamma\ \&\ \mathrm{RD}(\mathrm{Span}(\gamma,\mu))\Phi]$

12. type: $<e,<e,v>>$

LA (primitive)

13. type: $<e,<e,<v,v>>>$

$\mathrm{O}(\mu,\nu)\Phi \quad ::= \Diamond_D \sim\Phi \rightarrow \mathrm{LA}(\nu,\mu)$

$\mathrm{P}(\mu,\nu)\Phi \quad ::= \sim\mathrm{O}(\nu,\mu) \sim\Phi$

$\quad ::= \sim[\Diamond_D\Phi \rightarrow \mathrm{LA}(\mu,\nu)]$

14. type: $<e,<e,<v,<v,v>>>>$

$\mathrm{O}(\mu,\nu)\Phi/\Psi \quad ::= \Diamond_D \sim\Phi \rightarrow \Psi$

$\mathrm{P}(\mu,\nu)\Phi/\Psi \quad ::= \sim[\mathrm{O}(\mu,\nu)\ (\Phi)/\Psi]$

$\quad ::= \sim[\Diamond_D\Phi \rightarrow \Psi]$

PART II: FORMAL DESCRIPTION OF ECONOMIC ACTORS AND OBJECTS

CONTENTS

A.	INTRODUCTION	76
	Legal Framework	79
	Ownership and Possession	82
	Some Further Definitions and Notation	83
B.	ECONOMIC ACTORS	84
	Persons, Proprietorships	84
	Joint Ownership, Partnerships	84
	Private Corporations	85
C.	ECONOMIC OBJECTS	90
	Physical Ojbects	90
	Promissory Objects	91
	Monetary Objects	93
	Information Objects	95
D.	SUMMARY	98

II-A. INTRODUCTION

The purpose in this part is to illustrate the use of CANDID to the formal description of principal actors and objects of economic activity. This step contributes to the larger goal of formalizing the legal/accounting aspects of commerce that they may be subjected to a system of mechanical inference. Applications of such a system include aiding management to interpret internal cost systems, assistance in the management of the firms commercial, financial and legal obligations, and the analysis of commercial and financial regulatory systems.

The concepts that appear in such applications range from the mundane and commonplace, e.g., nuts, bolts, to the complex and esoteric, e.g., partially allocated costs, sale-leaseback agreements, the U.S. Securities and Exchange Commission Regulations.

The job of a formal language for describing such concepts is to render them unambiguous down to a limited set of primitive concepts which are consensually understood by all parties using the language.

A computer system using this language could therefore aid in rectifying definitional misunderstandings between disagreeing parties. Likewise, as an aid to individual decision making, it can explain any of its inferences in step by step elementary terms.

A critical factor, however, is that the language be based on primitive concepts that are clearly and unambiguously understood by all its users. Subsequent definitions based on these elementary terms can then be as intricate as necessary without the danger of magnifying an elementary ambiguity.

A fundamental issue here is the so-called "identification of particulars," of having consensual recognition and labeling of the individual entities described by the language.

Strawson (1959) argues that the proper basis for such identification is the locatability of these entities in a spatial/temporal framework. Thus for instance individual people in that they are borne at a particular place and time, and have continuity in space and time, and have continuity in space and time until their deaths. Given sufficient factual data about a person's whereabouts throughout time, an arbitrary group of observers could presumably agree as to the identification of this individual (e.g., whether it were really an actual person, or multiple persons, etc.)

Phenomena that do not have continuity in space and time are prone to much more disagreement of identification. Consider for example Beethoven's 9th Symphony. Is there one unique referent to this name or many? We may individuate versions of this symphony by its reproductions on paper or specific performances by orchestras but in both cases we re-cast it into a representation locatable in a space time framework. Textual works offer a similar difficulty. A more modern example is a computer program, for instance SPSS (statistical package for the social sciences), as an arbitrary example. There have been numerous versions of this program and hundreds of computer installations have one of these versions. Further, at any given installation, more than one copy of the program may be executed in the machine's memory at a given time.

This problem of individuation becomes especially important when we consider not just information objects, like symphonies and computer programs, but contractual objects like notes, bonds, stocks, options, licenses, insurance policies, etc. Clearly it is of critical importance for a company to know if it has a certain right or obligation. Indeed it is precisely because of this problem of identification that signed documents play such an important role in contractual transactions: the signed document represents the agreement in a form locatable in space and time.

As mentioned in the opening sentence, our goal here is to formally describe the principle actors and objects of economic activity. Our criterion for formalization will be the unique identification of such entities in this spatial/temporal framework.

If we consider only persons as economic actors and physical objects as economic objects, the problem is trivial: both types of entities are locatable in space and time.

However, another common type of economic entity (at least in western societies) is a corporation. A corporation is more problematic from this perspective since it has no essential physical reality: no one of its assets, including its buildings, nor any one of its employees nor any of its executives or board members nor any one of its stockholders is essential to the identification of the corporation. Any one of these may change or be removed from the corporation, and the identity of the corporation can still continue.

The objects of economic activity, i.e., the things that are traded, present analogous problems for formal description. Money for instance is a key object of exchange. Yet money is no longer uniquely represented by physical objects such as coins and bills, but often appears merely as magnetic records in bank accounts. These, like computer programs, lose the easy location in a unique place at a given time.

Information objects, such as recorded music, printed texts and computer programs were already mentioned as presenting a problem for identification. Such objects present an interesting legal problem in that they can be "stolen" (copied) without removal of the original. (Our notion of theft is basically a physical one.) Computer, communications and photocopy technology are bring the characteristics of this type of object to prime economic importance.

One other type of non-physical economic objects was also already cited: contractual objects. Signed documents have historically provided these types of objects with an easy physical identifiability. However, in most concentrated centers of trading in contractual objects, namely commodity, bond and stock market exchanges, there is a definite move towards automation of records and transactions, so that here to the identifiability of such objects becomes problematic.

Legal Framework

Concepts of economic actors and objects are defined within a general legal framework, which to a certain extent varies from one country to the next. The perspective taken here is an essentially capitalistic one, where corporations, independent though perhaps regulated by government, play a major economic role. Legal definitions and rules are all taken from United States law, the only code where the author has sufficient familiarity.

As a reference for the definitions used here, we have made use of *College Business Law*. (Rosenberg, R.R. and Ott, W.G., Sahum's Outline Series, McGraw-Hill Co., 1977). This is also suggested as a useful elementary-level reference text.

However, we hope that this starting point not be taken as a boundary. The foundation concepts of contractual obligation, permission, etc. have their analogues in any society that has moved beyond a simple barter system, and it is our belief that the concepts presented here are extendible to other economic systems, whether free market, centrally controlled, or some intermediate combination.

The general legal system we mention is of course established by the ruling government which is itself an important economic actor. However, insofar as the legal system generally reflects a long evolution in comparison to the shorter time frame of a government's transactions (i.e., the government generally cannot change the law from one transaction to the next), we prefer to separate the legal code from the government as economic actor, and consider the government and its agencies as regulated by the law as are other economic actors.

The assumption of a single legal code confines our attention here to transactions covered entirely by that code, i.e., to to domestic transactions. In Part I general permission and obligation (relative to an arbitrary set of laws and norms) was denoted as:

$\Diamond_D$ for permission (deontic possibility)

$\Box_D$ for obligation (deontic necessity)

Here to indicate our somewhat more restricted assumption to U.S. law, we designate this as

$\Diamond_L$ for permission under U.S. law

$\Box_L$ for obligation under U.S. law

Actually, in the U.S. there are two levels of commercial laws, one at the state and one at the federal level. The scope of the federal laws pertains primarily to inter-state commerce. When we want to indicate the operation of state law, as distinguished from U.S. federal law, we will use the notation:

$\Diamond_{L,x}$ for permission under the law of state x

$\Box_{L,x}$ for obligation under the law of state x

For instance,

$$\Diamond_{L,NY}\Phi$$

would indicate that Φ is permitted in the state law of New York.

Extension of this work to international commerce would employ still another legal level: international law. An essential difference at this level—which we avoid for present purposes—is the ultimate source of legal enforcement. In domestic transactions, the physical power of the ruling government is the ultimate enforcement of the law.

At the international level, lacking a single dominating world government, such transactions are subject to the treaties and agreements established between the nations involved, and the appeal for enforcement is correspondingly complicated.

Ownership and Possession

The most fundamental concept of economics, perhaps, is that of (legal) ownership, which is designated by the predicate:

$$\mathrm{OWN}(x,y)$$

meaning that x, an economic actor, *owns* y, an economic object. The essence of this paper is to elaborate the predicates that qualify x and y.

Here we adopt OWN as a primitive predicate. That is not to say it could not be analyzed further. For instance, there are certain differences in the concept of ownership between capitalist and communist countries, and to explicate international commerce one may want to describe these differences in terms of more elementary concepts.

Another relationship between economic actors and objects is that of possession, written

$$\mathrm{POSS}(x,y)$$

indicating that actor x possesses object y. Again, this is a fundamental concept which we take as primitive, though its meaning might vary somewhat in other economic systems.

Intuitively speaking, ownership constitutes a set of right granted by the legal system of an actor towards an object. Possession on the other hand refers to physical custody. Usually, an actor possesses what it owns, but not always, as in the case of loans and rentals.

Actually, in this paper, possession has only a minor role. It however figures more prominently in Part III, which discusses representation of financial contracts.

Some Further Definitions and Notation

In CANDID, predicates indicating a change in state may be defined using the connective T. Here, we will suffix the names of predicates so defined with the character "!" as a visual aid to reading the expressions. Using OWN and POSS, two such change predicates are defined as follows:

$$\mathrm{OCHANGE!}(x,y,z) ::= \mathrm{OWN}(x,y)\ \mathrm{T}\ \mathrm{OWN}(y,z)$$

$$\mathrm{PCHANGE!}(x,y,z) ::= \mathrm{POSS}(x,z)\ \mathrm{T}\ \mathrm{POSS}(y,z)$$

OCHANGE! indicates a change in ownership of the object z from x to y. PCHANGE! indicates a change in possession of the object z from x to y.

Also, in CANDID a concept of action is defined by using so-called TI expressions containing the connectives T and I. Here, again only as a visual aid, we use the suffix "!!" on the names of such predicates. Using OWN and POSS, four such action predicates may be defined:

$$\mathrm{OGIVE!!}(x,y,z) ::= \mathrm{OWN}(x,z)\ \mathrm{T}\ [\mathrm{OWN}(y,z)\ \mathrm{I}(x)\ \mathrm{OWN}(x,z)]$$

$$\mathrm{OTAKE!!}(x,y,z) ::= \mathrm{OWN}(x,z)\ \mathrm{T}\ [\mathrm{OWN}(y,z)\ \mathrm{I}(y)\ \mathrm{OWN}(x,z)]$$

$$\mathrm{PGIVE!!}(x,y,z) ::= \mathrm{POSS}(x,z)\ \mathrm{T}\ [\mathrm{POSS}(y,z)\ \mathrm{I}(x)\ \mathrm{POSS}(x,z)]$$

$$\mathrm{PTAKE!!}(x,y,z) ::= \mathrm{POSS}(x,z)\ \mathrm{T}\ [\mathrm{POSS}(y,z)\ \mathrm{I}(y)\ \mathrm{POSS}(x,z)]$$

In OGIVE!!, x causes a change of ownership of z from x to y. In OTAKE!!, y causes this same change of ownership to occur. In PGIVE!!, x causes a change of possession of z from x to y. In PTAKE!!, y causes this same change to occur.

II-B. ECONOMIC ACTORS

Persons, Proprietorships

The most obvious type of economic actor is individual persons, designated as:

$$PERSON(x).$$

However, in U.S. law, not all persons qualify as legitimate economic actors—minors and the insane are excluded. This more restricted set is designated LPERSON (legal person), defined as:

$$LPERSON(x) ::= PERSON(x) \,\&\, AGE(x,YR) \geq 18 \,\&\, SANE(x).$$

Personal businesses, owned by a single individual are called *proprietorships*. In U.S. law they are not distinguished from their owner, hence

$$PROPRIETORSHIP(x) ::= LPERSON(x)$$

Joint Ownership, Partnerships

Joint ownership is where one or more parties share equally in the ownership of an object. Essentially, the group of owners form a set which as a unit owns the object. For instance, for joint owners $x_1,\ldots,x_n$

$$OWN(z,y) \,\&\, z = \{x_1,\ldots,x_n\}$$

In U.S. law, a partnership is an economic actor consisting of such a set of equally participating persons. Hence,

$$PARTNERSHIP(z) ::= \exists x_1,\ldots,x_n\ LPERSON(x_1)\ \&\ \ldots\ \&\ LPERSON(x_n)\ \&\ z = \{x_1,\ldots,x_n\}$$

Private Corporations

It is at this level that the concept of an economic actor becomes philosophically challenging. A corporation is an artifice of the legal system. In the U.S., it is a "legal entity," entirely separate from and independent of its owners. Unlike proprietorships and partnerships, which are formed simply by the volition of the parties involved and have no separate legal status, a corporation is formed by a specially granted permission from the state.

Informally, this process is as follows. The group of people who want to start the corporation, called its *promoters*, submit registration information, called *incorporation papers*, and a *prospectus*, which describes the capital structure and intended function of the corporation to the governing state. If the corporation is to engage in interstate commerce, the prospectus must also be approved by the Securities and Exchange Commission (SEC).

In addition, a *certificate of incorporation* is filed by the promotors, which, if approved, is maintained by the office of the secretary of the state of incorporation. This certificate lists the corporation's principal offices, names of directors and incorporators, the total number of stock shares (each at a common value called the *par value)* and the name and number of shares held by each stockholder. The corporation cannot sell more than this initial number of shares without obtaining additional permission from the state. On acceptance by the state, this certificate becomes the corporation's *charter*.

This charter is a contractual permission by the state which, in gross terms, says the following: Stockholders have a right to vote members of the ***board of directors*** (at least three people) of the firm and to participate in the division of residual assets on the dissolution of the firm.

The board of director's main responsibility is to appoint *officers* of the corporation, which serve as the agents of the corporation in legal transactions (e.g., engaging the corporation in contracts, hiring and management of employees).

Only the officers, and the people they employ, can engage in the direct operation of the firm. Note that being a stockholder does not carry the right to participate in the management of the corporation nor to act as its agent in contracts.

To summarize, a corporation is essentially a locus of ownership, on one hand, and a locus of contractual commitment on the other. (These will define the two sides of the corporate balance sheet: its assets and its liabilities, including stockholder equity.) Changes in the things owned by the corporation and its commitments to other parties are made by the corporate officers and their employees, acting as agents. Corporate officers are appointed by the Board of Directors which in turn are voted by the stockholders.

A crucial issue from a formal standpoint, however, is the identification of this locus of ownership and commitment. If we simply dismiss it as an 'abstract object' having no spatial/temporal location, we are left with the theoretical as well as very pragmatic problem of determining when the corporation exists and the boundaries of its rights and obligations.

However, as noted above, the critical event in the formation of a corporation is the granting, by the secretary of the state of jurisdiction, of the corporate charter. This provides the creation of the corporation with a unique location in space and time. Furthermore, the corporate charter provides the corporation with a unique *corporate name* (within that state). This provides any subsequent contracts and titles of ownership with a reference to the corporate charter, and hence to a unique spatial/temporal location.

Though this provides the means to identify a corporation, we have still not explained what a corporation *is*. Clearly, it is not in itself something physical. Rather it is a complex of contingent rights and privileges as established by the corporate laws of the state.

Let us refer to this complex as CORP-RIGHTS. These are granted by a particular state, and associated to a unique (within the state) corporate name. Using the notation described earlier for permission under the law of state μ, this would be

$$\Diamond_{L,\mu}\text{CORP-RIGHTS}(\nu)$$

where ν is a variable of type c, a character string indicating the name of the corporation. This describes the situation where state μ permits corporate rights associated with name ν.

The types of this expression are:

$$\underbrace{\overset{\diamond_{L,\mu}}{e} \qquad \underbrace{\overset{\text{CORP-RIGHTS }(\nu)}{c}}_{<c,v>}}_{<<e,<c,v>>,v>}$$

That is, the characteristic function of CORP-RIGHTS maps from character strings to truth values. The state's legal permission is a mapping from an entity (the state) and the previous expression to a truth value.

We would like to say that the corporation is simply this permission. However, if we are speaking of a certain time, t, the corporation is not simply this permission at time t but to account for the corporation's ownership of assets, it must also include permission at previous times when the assets were acquired. Further, if the corporation is in operation it will presumably have contractual obligations to other parties. These involve evaluation of these corporate rights not only in future times but under alternative circumstances, i.e., in other possible worlds.

What we need then is to evaluate the corporate rights predicate not just currently in the 'actual' world, but across all times and in all possible worlds. This, as explained in Part I is provided by the intension operator, "$^\wedge$". Thus

$$^\wedge[\diamond_{L,\mu} \text{CORP-RIGHTS}(\nu)]$$

The earlier expression was of type $<<e,<c,v>>,v>$. The present expression will therefore be of type $<s,<<e,<c,v>>,v>>$, i.e., adding the additional argument of type s, which is an index to a possible world, time pair. Thus the characteristic function of this expression evaluates whether the corporate rights associated with name ν are permitted by state μ at each possible index. This, in our view, is what a corporation *is*. Hence

$$\text{PRIVATE-CORPORATION}(x) ::= \exists y \, \exists z \, \text{STATE}(y) \,\&\, \text{CHAR-STRING}(z) \,\&$$
$$x = {}^{\wedge}[\Diamond_{L,y} \, \text{CORP-RIGHTS}(z)]$$

The discussion here has been directed towards the formal description of *private corporations*, i.e., those which are profit oriented and have stockholders who ultimately receive these profits either through dividend distribution or dissolution of the corporation and sale of its assets.

Other types of corporations might also be described with a similar form of analysis. For instance, *non-profit corporations* do not have stockholders nor do they pay income tax. *Quasi public* corporations are private corporations which provide certain public services (e.g., certain utilities, toll roads) and which are supervised by public authorities. *Public corporations*, such as cities and certain department of local and state governments, also provide public services but are financed by the state. Each of these present certain variants on the concept of corporation we have just described.

Additionally, the concepts of state and federal governments themselves present a challenge to formal description. Indeed, they appear to be corporate-like entities, having no essential physical existence. However, in these cases one cannot appeal to a larger deontic framework as the basis for their definition, for they *are* this framework. Instead, at least in democratic societies, one would appeal to the consensus of the voting population (present and past) as a deontic basis. However, since our objectives here are to primarily concerned with commercial and financial activities, we confine our discussion only to the three classes of economic actors described above: proprietorships, partnerships, and private corporations. Hence,

$$\text{ECON-ACTOR}(x) ::= \text{PRORIETORSHIP}(x) \vee \text{PARTNERSHIP}(x) \vee \text{PRIVATE-CORPORATION}(x).$$

II-C. ECONOMIC OBJECTS

Physical Objects

The most obvious type of economic object are physical ones (i.e., having mass). As before, to admit these types of entities into the descriptive formalism we must be able to locate them in a spatial/temporal framework. For most types of physical objects we think of—e.g., tables, chairs, automobiles, real estate, this is unproblematic. However, when granular substances such as corn and wheat, or liquids or gases are involved, problems of identification arise because of the fluid movement of these substances. For instance consider a contract to buy a certain volume of ocean water located at a certain latitude and longtitude at a given depth, etc. Though the geographical coordinates may be certain, the particular volume of ocean water at this location is not.

The practical device that resolves this logical problem in nearly any reasonable commercial context is that of a *container*. Liquids, gases and grains are always handled in a container of some sort, and the container provides the fluid substance with a unique and stable spatial/temporal location and with that discrete identifiability.

Thus, our attention here is confined to what we call *discrete-physical- objects*, which have distinct spatial/temporal coordinates (for instance at their center of gravity) and can be uniquely identified and named. Liquids, gases and grains are assumed always to appear within discrete containers so that the filled container is itself a discrete physical object.

We are concerned here with those types of objects that can be owned. Normally, any discrete physical object can be owned; however U.S. law specifically excludes one type, persons (slavery having been abolished). Hence, we introduce a concept of LPHYS-OBJ (legal physical object) which are those that can be owned:

$$\text{LPHYS-OBJ}(x) ::= \text{DISCRETE-PHYS-OBJ}(x) \;\&\; {\sim}\text{PERSON}(x).$$

Promissory Objects

If one examines asset side of the balance sheet of a company (which lists categories of what the company owns) one of course finds a number of categories which are types of physical objects, e.g., land, plant and equipment, inventory. However , beyond these there are typically other categories that do not comprise physical objects—e.g., accounts receivable, negotiable securities, patents, licenses.

These are what we call *deontic objects*. They arise as the result of a contractual permission of which the company is the beneficiary, i.e., they are 'rights' permitting the company to do something (as with licenses) or obligations of other parties to the company (as with accounts receivables, and negotiable securities).

We consider the case of contractual permissions first. This is a permission by some other party, say x, to the economic actor, call it y, to do some action, say Φ. Hence

$$P(x,y)\Phi.$$

We would like to say that y owns this permisison. However, it is not the assertion itself that y owns, but its sense or 'intension,' that is, its interpretation across all possible worlds and times.* This is given by once again using the intension operator:

$$^{\wedge}[P(x,y)\Phi]$$

Thus, to intent a term for the 'object' form of a permission, we call it LPRIVILEGE (legal privilege). Hence for economic actors "y" and "z" and some action, "act"

$$\text{LPRIVILEGE}(x) ::= \exists y \exists z \exists \text{act} \quad x = {}^{\wedge}[P(y,z)\ \text{act}]$$

The treatment for the case of contractual obligations is similar. Here we will call the object form an LPROMISE (legal promise). Again for

*Note: Contractual permission was defined in terms of general permission (deontic possibility) which in turn had a semantic interpretation across possible worlds and times. Thus contractual permission is not just permission in the present but in certain future times and circumstances as qualified by Φ. Use of the intension operator here thus appears as a second lambda abstraction across indices. The purpose of this second abstraction is essentially to 'objectify' the permission, equating it with its characteristic function across possible worlds and times.

economic actors "y" and "z" and an action, "act,"

$$LPROMISE(x) ::= \exists y \exists z \exists act \quad x = {}^{\wedge}[O(y,z)\ act].$$

A deontic object is one of either of these types:

$$DEONTIC\text{-}OBJ(x) ::= LPRIVILEGE(x) \vee LPROMISE(X).$$

Monetary Objects

Money is obviously an important type of object in the description of commercial and financial phenomena. If we consider money only in the form of 'hard cash,' i.e., coins and bills, money is simply a type of physical object:

$$CASH\text{-}MONEY(x) \rightarrow LPHYS\text{-}OJB(x).$$

Coins and bills are obviously of a particular national currency and have a face value. Thus for instance in the U.S., predicates indicating common types of bills and coins are

ONE-CENT-COIN(x)
FIVE-CENT-COIN(x)
TEN-CENT-COIN(x)
ONE-DOLLAR-BILL(x)
TEN-DOLLAR-BILL(x)

etc.

However, in commercial transactions, money is seldom handled at this detail level, but rather as sums of money. In this case we add up the face values of the various coins and bills, and convert them to a common currency unit—e.g., cents or dollars.

Thus, suppose that y is a set of coins and bills, $x_1,\ldots,x_n$. Then the monetary value of y, say n, would be given by a measurement function:

$$y = \{x_1,\ldots,x_n\} \;\&\; \text{MONEY-VALUE}(y, \text{Dollar}, \text{US}) = n$$

Note here that the measurement function has a third place indicating the nationality of the currency, for instance to distinguish measurement in U.S. dollars versus Canadian dollars. (Exchange rates between currencies are described as the tabulated face value of one currency exchangeable for a unit tabulated face value in another currency.)

Since our examples here have assumed a U.S. environment, we introduce as a notational convenience an abbreviation for money in U.S. dollars:

$$\$(y)=n \quad ::= \quad \text{MONEY-VALUE}(y, \text{Dollar}, \text{US})=n$$

This measurement function is for tabulating face values of a sum of currency in a given nationality. Measuring one nation's currency in terms of another with this function would thus evaluate zero.

So far we have regarded money as a special type of physical object. However, the services provided by lending institutions in most countries have extended this concept of money.

In the U.S., it is quite common that a bank check is given and accepted in lieu of cash money. These checks are made against 'demand deposit' accounts in a bank, which promises to pay the payee named on the check a sum of money whose tabulated value equals the amount specified on the check.

Demand deposits are thus a deontic object, indicating the obligation of the bank, say b, to the party named on the check, say x, an amount of money, assuming U.S. dollars, n:

$$\text{DEMAND-DEPOSIT}(b,x,n) ::= {}^{\wedge}[O(b,x){:}\ \exists m\ \&\ \$(m)=n\ \&\ \text{OGIVE!!}(b,x,m)]$$

Reading: a demand deposit from bank b to party x in amount n for some amount of money m, whose tabulated face value in U.S. dollar is n, b given ownership of m to x.

Because checking accounts are used so often, we introduce another notational abbreviation to indicate money either in the form of cash or check:

$$\$\$(x) = n\ ::=\ \$(z) = n \vee (\exists b)(\exists x)\ x = \text{Demand-Deposit}(b,x,n)$$

The two abbreviations for U.S. dollars correspond to the two concepts of money used by the U.S. Federal Reserve Board to calculate the money supply. Our notation **$** corresponds to the money supply measure, M1, our **$$**, corresponds to M2.

Information Objects

Physical objects, deontic objects and money account for most of the types of objects that are owned by economic actors and traded in commercial transactions.

However, there appears to be one additional class of ownable and tradable objects not yet included: what we call *information* objects.

Informally, an information object is some meaningful arrangement of symbolic patterns on a representational medium, e.g., ink on paper or electronic codes on a magnetic tape or disk.

Our concept of information object corresponds to what Thompson (1981) calls "ethereal goods." He makes the excellent observation that what is distinct about this type of object is the technology of its reproduction. Thus, to him, an ehtereal good is one that can be reproduced more cheaply than it can be purchased.

Thus, up until the time of the photocopy machine, a book was not an ethereal good. Now there are many books that are cheaper to photo-copy than purchase from the publisher (especially low volume technical books).

Similarly, home stereo tape recorders made it cheaper to copy musical recordings than buy them.

However, the innovation that really expanded the class of ethereal goods was the electronic computer. A fundamental concept in this technology is that data is easily and instantly copyable. Hence any information converted for computer storage (or indeed programs directing the processing of data) can be instantaneously reproduced (copied to another magnetic medium or sent over communication lines) at practically no cost.

Since considerable labor is often expended in the original creation of such information objects, the legal problem this presents is how to protect the developer from having his/her work "stolen," i.e., reproduced, without compensation.

Our concern here, however, is only with the description of these types of objects. As we have seen, their essential characteristics are not the physical medium on which they are represented, but their reproductibility.

In owning such an information object, therefore, one of course owns the physical representation medium, but more importantly, one owns rights controlling the reproduction of the object. (Thus, the copyright laws for textual material prescribe the "copy rights" of the author and publisher.)

Thus, in the perspective here, the essential features of an information object are very similar to that of a license, i.e., a contractual permission from one party to another. In the case of information objects, the permitted action is a certain limited range of reproduction. Let us refer to instances of these actions is LTD-REPRODUCTION!!. In acquiring an information object, one therefore acquires a physical representation of the information object plus certain rights of limited reproduction.

Let k be this physical representation, x be the party acquiring the information object, and y be the author or holder of the copyright of the object. Then the rights transferred, which for us *is* the information object is defined as follows:

$$\text{INFO-OBJ}(z) ::= \exists x \exists y \exists k \; z + {}^{\wedge}[P(y,x) \; \text{LTD-REPRODUCTION!!}(k)]$$

Reading: An information object, z, is defined as for some parties x and y and a physical representation k, the permission of y to x to certain actions of limited reproduction of k.

II-D. SUMMARY

An economic actor is defined:

$$\text{ECON-ACTOR}(x) ::= \text{LPERSON}(x) \vee \text{PROPRIETORSHIP}(x) \vee \text{PARTNERSHIP}(x) \vee \text{CORPORATION}(x)$$

An economic object is defined:

$$\text{ECON-OBJ}(z) ::= \text{LPHYS-OBJ}(z) \vee \text{DEONTIC-OBJ}(z) \vee \text{MONETARY-OBJ}(z) \vee \text{INFO-OBJ}(z).$$

As noted earlier, the class of monetary objects comprises certain physical objects (coins and bills) and certain deontic objects (demand deposits). Also, an information object has both physical and deontic aspects to it—the physical representation of the original and the limited rights of reproduction. Thus the above definition of economic object is redundant to this extent.

The two place predicates OWN and POSS were taken as primitive. To indicate that each is a relation between economic actors and economic objects, we have the following controlling axioms:

$$\text{OWN}(x,z) \rightarrow \text{ECON-ACTOR}(x) \ \& \ \text{ECON-OBJ}(z)$$
$$\text{POSS}(X,z) \rightarrow \text{ECON-ACTOR}(x) \ \& \ \text{ECON-OBJ}(z).$$

PART III: CANDID DESCRIPTION OF FINANCIAL CONCEPTS

CONTENTS

A.	INTRODUCTION	100
B.	ADDITIONAL DEFINITIONS, NOTATIONAL CONVENTIONS	101
C.	ELEMENTARY FINANCIAL CONCEPTS	103
	Loans	103
	Loans of Money	104
	Simple vs. Compound Interest	107
	Present Value of a Debt	108
	Partial Payments	110
D.	FINANCIAL INSTRUMENTS	112
	Leases	113
	Options	114
	Insurance	117
	Easements, Licenses	118
	Dept Instruments	119
	Equity Instruments	125
	Convertibles	128
E.	CONCLUDING REMARKS	129

III-A. INTRODUCTION

In Part I, the formal descriptive language CANDID was developed. In Part II this was applied to the description of the principal entities of economic activity, what we called economic actors and economic objects. In this part, we extend the application of CANDID to consider the processes of economic activity itself, in describing the concepts of elementary finance, i.e., common types of transactions and financial instruments. We find this domain to be not only a fairly central and important one to understanding commercial activity more broadly, but also reasonably representative of the classes of conceptual problems likely to arise in efforts to formalize other aspects of business. We thus believe that analogous analyses could be applied for instance to financial accounting, cost accounting, tax law, contract law, regulatory law, etc. Again, we want to emphasize that CANDID is proposed as a *framework* for formalizing business theory, but is not intended as a theory itself. The discussion here is thus meant to be only illustrative, attempting to capture what we see as the ordinary usage and understanding of basic financial terminology and concepts. Various contemporary theories of accounting, finance and economics might therefore disagree with aspects of the analysis given here. (The only responsibility we would claim for CANDID is to explicate this disagreement.)

As a general guide to what concepts should be included here, we made use of *Mathematics of Finance* (Ayres, F., Jr., Shaum's Outline Series, McGraw Hill, 1963), beginning level college primer. This is likewise suggested as an elementary background reference.

III-B. ADDITIONAL DEFINITIONS, NOTATIONAL CONVENTIONS

In Part II the concepts of an *economic actor* and an *economic object* were developed. Informally, an economic actor is a legally able person or organization (proprietorship, partnership or corporation) while an economic object is a physical object (excluding persons), a contractual object (e.g., stock, bonds, licenses), a monetary object (cash or demand deposit checks) or an information object (e.g., textual materials, computer data and programs).

In addition, two two-place relations between economic actors and economic objects were assumed, OWN (for ownership) and POSS (for possession). These have the following associated axioms:

$$\text{OWN}(x,y) \rightarrow \text{ECON-ACTOR}(x)\ \&\ \text{ECON-OBJ}(y)$$

$$\text{POSS}(x,y) \rightarrow \text{ECON-ACTOR}(x)\ \&\ \text{ECON-OBJ}(y).$$

Also, the notation **$$** is used to indicate U.S. currency in cash or check form. E.g.,

$$\mathbf{\$\$}(m) = 158.32$$

indicates that the object m is a sum of money totaling \$158.32.

As in the earlier parts, parentheses are used for functional application arguments for predicates and functions), while square brackets are used for syntactic disambiguation. Also, to reduce the notation, a colon is used to abbreviate a left bracket matched by an implicit right bracket before the next open right bracket or at the end of the sentence, whichever comes first.

Also as previously, predicates may indicate states, changes or actions. As a visual aid, we append "!" to predicate names for changes and "!!" to names of actions. Thus, as in Part II we have the following definitions of changes and action relating to ownership and possession.

OCHANGE!(x,y,z) ::= OWN(x,z) T OWN(y,z)

PCHANGE!(x,y,z) ::= POSS(x,y) T POSS(y,z)

OCHANGE! indicates a change in ownership of z from x to y. PCHANGE! indicates an analogous change of possession.

OGIVE!!(x,y,z) ::= OWN(x,y,z) T [OWN(y,z) I(x) OWN(x,z)]

OTAKE!!(x,y,z) ::= OWN(x,y,z) T [OWN(y,z) I(y) OWN(x,z)]

PGIVE!!(x,y,z) ::= POSS(x,z) T [POSS(y,z) I(x) POSS(x,z)]

PTAKE!!(x,y,z) ::= POSS(x,z) T [POSS(y,z) I(y) POSS(x,z)]

OGIVE!! indicates a change of ownership from x to y of z initiated by x, whereas OTAKE!! indicates the same change of ownership, but initiated by y. PGIVE!! and PTAKE!! are defined similarly but for possession.

One additional definition is added for the purposes of this part, an action, PROMISE!!, indicating the creation and giving of a "deontic object," i.e., (the intension of) a contractual obligation or permission:

PROMISE!!(x,y,z) ::= ~EXISTS(p) T [EXISTS(p) & OGIVE!!(x,y,z)]

Also, we will here need a shorthand device for describing series of conjuncts that vary only in the definition of variable names and certain numeric parameters. We call this device *iteration* and define it as follows.

The notation

$$[1 \leq i \leq n]$$

is read "for i from 1 to n" and is meant to assign integer values to i = 1,2,...,n. Further, for a variable μ,

$$\mu[i]$$

is replaced with respective subscripts i = 1,...,n.

Thus for a formula Φ, containing variables $\mu,\ldots,\nu$

$$[1 \leq i \leq n]:\ \Phi(\mu[i],\ldots,\nu[i]) ::= \Phi(\mu_1,\ldots,\nu_1)\ \&\ \Phi(\mu_2,\ldots,\nu_2)\ \&\ \ldots\ \Phi(\mu_n,\ldots,\nu_n)$$

III-C. ELEMENTARY FINANCIAL CONCEPTS

Loans

Loans are a familiar and everyday concept. We think usually of a loan as letting someone use something of ours with the understanding that they will return it to us at a later time. Implicit in this notion of lending is the expectation that the borrower return the *same* object lent. We call this a *loan in substance*. For instance, renting a car or house involve loans in substance.

Another type of loan, one which is especially common in business, might be called a *loan in kind*. Here the expectation is that the object returned need not be the same object, but only of the same type. For instance, loans of money, grain or oil are typically loans in kind.

These two types of loans are discriminated in CANDID as follows:

$$\text{LOAN-IN-SUBSTANCE!!}(x,y,z,t) ::=$$
$$\text{PGIVE!!}(x,y,z)\ \&\ (\exists p)\ \text{PROMISE!!}(y,x,p)\ \&$$
$$p = {}^{\wedge}[O(x,y)\text{: } RD(t)\text{: } \text{PGIVE!!}(y,x,z)]$$

Reading: x, the lender, gives y, the borrower, the object z, and y promises x that it be obligatory for y to realize sometime during time t the giving back of the same object, z.

$$\text{LOAN-IN-KIND!!}(x,y,\Phi,t) ::=$$
$$(\exists z_1)\ \Phi(z_1)$$
$$\&\ \text{PGIVE!!}(x,y,z_1)\ \&\ (\exists p)\ \text{PROMISE!!}(y,z,p)\ \&$$
$$p = {}^{\wedge}[O(x,y)\text{: } (\exists z_2)\ \Phi(z_2)\ \&\ RD(t)\ \text{PGIVE!}(y,x,z_2)]$$

The reading here is similar to before except that now the object returned is not necessarily that same one, but only one that satisfies the same predicate, Φ. Note that this second object does not necessarily exist when the LOAN-IN-KIND!! is realized.

Loans of Money

Loans of money are loans in kind where Φ is a money predicate. Most commonly, however, the borrower is obligated to repay a larger amount than what was borrowed, the difference being the *interest* of the loan. A loan of money with interest is thus a loan involving two kinds:

$$\text{LOAN-OF-TWO-KINDS!!}(x,y,\Phi\ \Psi,t) ::=$$
$$(\exists z_1)\ \Phi(z_1)\ \&\ \text{OGIVE!!}(x,y,z_1)\ \&\ (\exists p)\ \text{PROMISE!!}(x,y,p)\ \&$$
$$p = {}^{\wedge}[O(x,y)\text{: } (\exists z_2)\ \Psi(z_2)\ \&\ (RD(t)\text{: } \text{OGIVE!!}(x,y,z_2)$$

Here, x gives z_1 (which satisfies Φ) to y, in exchange for y's promise to later return to x some object z_2, which satisfies Ψ. Thus the thing given and the thing returned neither are the same thing, nor do they even satisfy the same predicate. This hardly seems like a loan any more. However, in loans of money, Φ and Ψ are both money predicates which differ only in amount. For simplicity, let us assume that the currency is U.S. dollars. Then, a loan of money with interest can be defined more specifically as follows:

$$\begin{gathered} \text{LOAN-OF-MONEY1!!}(x,y,n_2,t) ::= \\ (\exists m_1)\ \$\$(m_1)=n_1\ \&\ \text{OGIVE!!}(x,y,m_1)\ \& \\ (\exists p)\ \text{PROMISE!!}(y,x,p)\ \& \\ p = {}^{\wedge}[O(x,y){:}\ (\exists m_2)\ \$\$(m_2)=n_2\ \&\ (RD(t){:}\ \ \text{OGIVE!!}(y,x,m_2))] \end{gathered}$$

It is more usual to specify the second amount of money as a multiple of the first. The common method is to designate a fraction, r_1, (where 100 * r_1 = percentage) which is the incremental portion of the first amount to be added in repayment. In this form we have:

$$\begin{gathered} \text{LOAN-OF-MONEY2!!}(x,y,n_1,r_1,t) ::= \\ (\exists n_2)\ n_2 = n_1 * (1 + r_1)\ \& \\ \text{LOAN-OF-MONEY1!!}(x,y,n_1,n_2,t) \end{gathered}$$

It is also common, at least in the U.S. to specify r_1 as an *annual* rate; i.e., the actual multiplier to be applied to n_1, call it r_2, is determined by multiplying r_1 by the duration of t in years.

Thus the loan of money pr$_1$edicate which takes r to be an annual rate would be as follows:

$$\text{LOAN-OF-MONEY3!!}(x,y,n_1,r_1,t) ::=$$
$$(\exists r_2)\ r_2 = r_1 * \text{Dur}(t,\text{Yr})\ \&$$
$$\text{LOAN-OF-MONEY2!!}(x,y,n_1,r_2,t)$$

where "Dur" measures the duration in years of the time span t.

The interpretation so far has been that the borrower is obliged to repay the principal and interest some time within the period t. As described in this last predicate, the borrower must pay the full amount of interest irregardless of how early in this period re-payment is made. While this is in fact the condition of some loans, others limit the amount of interest to apply only to the period up to the point of repayment. This form of loan would be defined as follows:

$$\text{LOAN-OF-MONEY4!!}(x,y,n_1,r_1,t_1) ::=$$
$$(\exists m_1)\ \$\$(m_1)=n_1\ \&\ \text{OGIVE!!}(x,y,m_1)\ \&$$
$$(\exists p)\ \text{PROMISE!!}(y,x,p)\ \&$$
$$p = {}^{\wedge}[O(x,y){:}\ (\exists m_2)\ (\iota\, t_2)\ (\exists t_3){:}$$
$$\text{End}(t_2) < \text{End}(t_1)\ \&$$
$$\text{RT}(t_2)\ [\text{OGIVE!!}(y,x,m_2)]\ \&$$
$$\text{Beg}(t_3)=\text{Beg}(t_1)\ \&\ \text{End}(t_3)=\text{End}(t_2)\ \&$$
$$\$\$(m_2)= m * (1 + r_1 * \text{Dur}(t_3,\text{Yr})))$$

Reading: For some money, m_1, in the amount n_1, x gives this money to y; y promises that for some other money, m_2, a unique time span t_2 and some other time span, t_3, where t_2 ends before t_1 ends and ***throughout*** t_2, y gives x the money, m_2, and for the time span t_3 which began with t_1 and ended with t_2, m_2 is an amount of money equal to n_1 plus the interest on n_1 over time t_3.

Note that the promise in this case involved the introduction of two time periods, t_2 and t_3, where t_2 was the (relatively short) time in which repayment is realized *throughout*, while t_3 was the time from the start of the loan to this repayment.

Simple vs Compound Interest

The interest computation in the last case is called simple interest. Often a more complex computation is used called *compound interest*. The basic effect of this is that for some time interval, called the compounding period, the interest for the period is computed and added *to the principal* for the subsequent computation.

Suppose the loan is for $1,000 at an annual rate of .05 for three years. Assuming a compounding period of a year, a comparison of the two methods is as follows:

End of year	Simple Interest		Compound Interest	
	princ	int	princ	int
1	1000	50	1000	50
2	1000	50	1050	52.50
3	1000	50	1102.50	55.12
Total (princ + int)	3150		$3157.62	

Compounding is obviously advantageous to the lender. The computations for simple and compound interest, assuming principal = m, annual rate = r, total loan duration t_1, and compounding period t_2, are as follows:

$$nsimp = m * (1 + * Dur(t_1,Yr)))$$

$$ncomp = m * (1 + r) ** Dur(t_1,t_2)$$

While adding arithmetic complexity, compounding does not seriously complicate the descriptive complexity of our calculus. To modify the previous example to reflect compounding, one would simply change the formula for the amount of m_2 in the last line.

Present Value of a Debt

A loan or debt has value to the lender. Insofar as the promised future repayments are reasonably assured, the lender typically regards this as a component of his/her *present* wealth, even though it is only the promissory object which is actually owned. (Wealth here is taken to be the collection of things owned, according to the CANDID definition of OWN.) In business it is very important to measure these and other forms of wealth. Since it is by the proxy of such measurements that economic objects are made numerically comparable, decision making is simplified by reducing it to arithmetic calculations and comparisons. Usually wealth is measured in monetary terms. For cash, wealth obviously is the total face amount of the currency. For physical and informational objects, wealth is typically measured as the original amount of cash paid for the object (sometimes with an adjustment for deterioration and/or obsolescence). With respect to promissory objects for *future* cash, one might initially value them as the amount of the cash expected. However, most business and economic theorists would regard this as incorrect for two reasons:

a) there is always some chance that the borrower may renege on the promise and the future cash may not be collectable.

b) if the total amount to be paid were immediately available, one could invest it elsewhere (e.g., in a bank, securities, other loans) and make additional interest.

Thus a promise for future cash is usually regarded as having *less* monetary value than an equal amount in the present. This more conservative valuation is termed the *present value* of the promise.

While our concern in CANDID is with the formal *description* of phenomena only, and not with *valuation* (which we see as a problem for accounting and economics), there is a commonly accepted and used method for computing the present value of future cash receipts that we feel should be mentioned here.

This method involves the assumption of a rate, d, called the *discount rate* which might be considered as a sort of counter-factual interest rate. It is the hypothetical average rate of return at which cash presently available could be invested.

Considering some future cash amount, n_1, expected after a period t_1, the present value is the amount, n_2, which if invested now at the discount rate would yield money in the amount n_1. That is,

$$n_1 = n_2 * (1 + (d * Dur(t_1, Yr)))$$

hence,

$$n_2 = n_1 / (1 + (d * Dur(t_1, Yr)))$$

Partial Payments

Loans are often re-paid in a series of partial payments rather than as a lump sum. Sometimes these are of equal size and in regular intervals, though not necessarily. With respect to partial payments, it is important to distinguish the requirements of the loan from the options available to the borrower. For instance, a loan may specify payment of 36 monthly installments of a certain amount. Sometimes, however, the terms of the loan may disallow early payment. This, as we will understand it here, is not to be taken literally. Early payment is always advantageous to the lender. By such a stipulation, it is generally intended that the borrower will receive no reduction in interest due by such pre-payment. This is basically the distinction made in the predicates LOAN-OF-MONEY3!! and LOAN-OF-MONEY4!! above. As observed there, the difference in the loan specification is that in the latter case, the amount of interest depends on the time of pre-payment. To describe loans involving partial repayments with no adjustments of interest for early payment, we can ignore the interest computation and regard the borrower's promise as a series of payments of certain pre-specified amounts n_1, n_2, ..., n_k required on or before certain dates, t_1, t_2, ..., t_k. The borrower''s obligation in this case simply covers a series of realization formulas in conjunction. For instance, suppose that on Jan. 1, 1980 John Doe (j) borrows $1000 from his local bank (b), with repayment specified in three amounts as follows:

$250 on Dec. 31, 1982

$500 on Dec. 31, 1983

$300 on Dec. 31, 1984.

The CANDID description of this loan event and John's obligation are as follows:

$$
\begin{gathered}
RD((Date(1,1,1980)):\ (\exists m_1)\ \$\$(m_1) = 1000 \\
\&\ OGIVE!!(b,j,m_1)\ \&\ (\exists p)\ PROMISE!!(j,b,p)\ \& \\
p = {}^{\wedge}[O(x,y):\ (\exists m_1)\ (\exists m_2)\ (\exists m_3): \\
\$\$(m_1)=250\ \& \\
\$\$(m_2)=500\ \& \\
\$\$(m_3)=300\ \& \\
(\exists t_1)\ (\exists t_2)\ (\exists t_3): \\
Beg(t_1)=Beg(t_2)=Beg(t_3)=Beg(Date(1,1,1980))\ \& \\
End(t_1)=End(Date(31,12,1982))\ \& \\
End(t_2)=End(Date(31,12,1983))\ \& \\
End(t_3)=End(Date(31,12,1984))\ \& \\
RD(t_1)\ [OGIVE!!(j,b,m_1)]\ \& \\
RD(t_2)\ [OGIVE!!(j,b,m_2)]\ \& \\
RD(t_3)\ [OGIVE!!(j,b,m_3)]
\end{gathered}
$$

A more common formulation of a loan involves a series of equal size payments over regular intervals. The intervals most commonly used are that of a month or year which, as was noted earlier, are of varying length but nonetheless unambiguous. A loan of amount n_1 to be repaid as a series of k installments each of size n_2 in intervals of length t_1 beginning at time t_0 is described as follows:

$$\text{LOAN-OF-MONEY5!!}(x,y,n_1,n_2,k,t_1,t_0) ::=$$
$$(\exists m_1)\colon\ \$\$(m_1)=n_1\ \&\ \text{OGIVE!!}(x,y,m_1)\ \&$$
$$(\exists p)\colon\ \text{PROMISE!!}(x,y,m_1)\ \&$$
$$p = {}^{\wedge}[O(x,y)\colon\ [i = 1,\ k]\colon (\exists t[i])\ (\exists m[i])\colon$$
$$\text{Beg}(t[i])=\text{Beg}(t_0)\ \&\ \text{Dur}(t[i])=t_1\ \&$$
$$\$\$(m[i])=n_2\ \&$$
$$\text{RD}(t[i])\colon\ \text{OGIVE!!}(y,x,m[i])$$

These descriptions provide for no reduction in interest for early payment. When that is the case, a modification analogous to that in LOAN-OF-MONEY4!! is required.

III-D. FINANCIAL INSTRUMENTS

In the last section we looked mainly at the process of loaning money. That is, the lender gave some sum of money in exchange for the borrower's promise to pay it back in various ways. We now broaden our scope to include other financial mechanisms. As shall be seen, the notion of promise, hence promissory objects, will continue to play a central role.

In approximate accordance with general usage, we refer to the promises themselves as *financial instruments*. Also in deference to general usage, the terminology of "lender" and "borrower" needs to be generalized. Broadly, we will call these the "promises" and "promissor," respectively. In more narrow contexts, these parties will be assigned more specific role names.

Leases

Leases are agreements involving monetary payments in exchange for rental or temporary possession of a physical economic object, e.g., an apartment, house, car, truck, machine, building, land. Accountants are quick to focus on the temporariness of this possession, and when it approximates the useful life of the object, they argue that the lease effectively amounts to a sale of the object plus a corresponding financing arrangement (loan). The technicality of casting such would-be sales as leases often has certain tax advantages.

Leases where the duration of possession is short relative to the object's life are termed *operating leases*. Those where the possession approximates the useful life of the object are ***financial leases***.

Let p be a promise (promissory object) to pay certain amounts of cash over a specified period. Then a rental for an object, z, over a period t_1, is described as follows:

$$\text{LEASE1!!}(x,y,z,t_1,p_1) ::=$$
$$\text{PGIVE!!}(x,y,z)\ \&\ \text{PROMISE!!}(y,x,p_1)\ \&$$
$$(\exists p_2)\ \text{PROMISE!!}(y,x,p_2)\ \&$$
$$p_2 = {}^{\wedge}[O(x,y){:}\ \ RD(t_1){:}\ \ \text{PGIVE!!}(y,x,z)]$$

Reading: x gives possession of z to y and y makes the promise p_1 (left unspecified, but presumably to pay money), and in addition y agrees to the promise p_2 which is the obligation to realize during t_1 the giving back of possession of z to x. Here the roles indicated as x and y are usually termed "lessor" and "lessee." Note: as described here, the lease involves two $p_1$romises: p, to pay money, and p_2, to return the rented object. Had we wished to specify p, these could have been combined as a single promise.

Financial leases often provide an option for the lessee to purchase the object at the end of the lease period for a usually insignificant amount, call it n_1. Such a provision is incorporated as follows:

$$\begin{gathered} \text{LEASE1!!}(x,y,z,t_1,p_1,n_1) ::= \\ \text{PGIVE!!}(x,y,z) \ \& \ \text{PROMISE!!}(y,x,p_1) \ \& \\ (\exists p_2) \ \text{PROMISE!!}(y,x,p_2) \ \& \\ p_2 = {}^\wedge[O(x,y){:}\ \exists m_1) \ \$\$(m_1)=n_1 \ \& \\ ((\text{RD}(t_1) \ \text{PGIVE!!}(y,x,z)) \ \text{W} \\ (\text{RD}(t_1) \ \text{OGIVE!!}(y,x,m_1) \ \text{T} \ \text{OCHANGE!}(x,y,z))] \end{gathered}$$

Reading: x gives to y $p_1$ossession of z; y promises p (unspecified cash payments) to x; y also promises p_2 to x; the effect of p_2 is the obligation that: for some money m_1 in the amount n_1, either y gives to x the object z, or y gives to x the money m_1, in which case there is an (automatic) ownership change from x to y of the object z.

Options

Options as a general concept are a sort of conditional promise subject to the promisee's control. The two parties involved are sometimes distinguished as the ***issuer*** of the option (the promissor) and the option ***holder*** (the promisee).

Let Q1 and Q2 be temporally unbound states of affairs, and t_1 be the span of time in which the option holds. Then the CANDID description of this is as follows. The general form of an option is the issuer's promise that if the holder acts to bring about the state of affairs Q1, then the issuer is obligated to act to bring about state Q2:

$$\begin{gathered}\text{OPTION!!}(x,y,Q1,Q2,t_1) ::= \\ (\exists p)\ \text{PROMISE!!}(x,y,p)\ \& \\ p = {}^{\wedge}[O(x,y):(\forall t_2)\ [PT(t_2,t_1)\ \&\ RT(t_2) * T\ (Q1\ (I\ x)\ *))] \rightarrow \\ [(\exists t_3):\ Beg(t_3)=End(t_2)\ \&\ RD(t_3) * T\ (Q2\ (I\ y)\ *))]]\end{gathered}$$

Reading: x makes some promise to y that for any time t_2 in t_1, if x brings about Q1 (from any state instead of any state) then it is obligatory that for some t_3 which begins as y ends, it is realized during t_3 that y brings about Q2 (from any state instead of any state).

Commonly occurring types of options are made for the purchase or sale of publicly traded stock, usually in units of 100 shares.

A "call" is an option to buy 100 shares of stock at a predetermined price. Obviously, if the market price of the stock goes above this pre-set price, one can exercise the option and sell the stock in the open market at a profit. Thus, for stock in company z, at a call price of m, a call can be defined in terms of the preceding definition for an option as follows:

$$\begin{gathered}\text{CALL!!}(x,y,z,n_1,t_1) ::= \\ \text{OPTION!!}(x,y,Q1,Q2,t_1)\end{gathered}$$

where

$$\begin{gathered}Q1 \leftrightarrow [(\exists m_1)\ \$\$(m_1)=n_1\ \&\ \text{OGIVE!!}(x,y,m_1)] \\ Q2 \leftrightarrow [(\exists w):\ w = \{u|\ Stock(u,z)\}\ \& \\ Count(w,\ Stock)=100\ \&\ \text{OGIVE!!}(y,x,w)]\end{gathered}$$

Here, Q1, the condition of the option, is that x gives y money in the amount n_1. Q2, the obligation initiated by Q1, is that y gives a collection consisting of 100 shares of stock in company z to x.

A "put" is the converse of a call. It is an option to sell 100 shares of stock at a pre-established price. The holder's strategy in a put is usually that if the market price declines to below the pre-set price, the holder can buy the lower cost stock in the market and then exercise the option in order to sell it at the higher put price.

The CANDID definition of a put is quite similar to a call; simply, the definitions of Q1 and Q2 are interchanged:

$$\text{PUT!!}(x,y,z,n_1,t_1) ::=$$
$$\text{OPTION!!}(x,y,Q1,Q2,t_1)$$

where,

$$Q1 \leftrightarrow [(\exists w):\ w = \{u|\ \text{Stock}(u,z)\}]$$
$$Q2 \leftrightarrow [(\exists m_1)\ \$\$(m_1)=m\ \&\ \text{OGIVE!!}(x,y,m_1)]\ \&$$
$$\text{Count}(w,\ \text{Stock})=100\ \&\ \text{OGIVE!!}(y,x,w)]$$

Other types of options derive from puts and calls. A "spread" is a combination of a put and a call written on the same stock and running for the same length of time. The put price is below the current market, while the call price is above it. A "straddle" is a spread where the put and call prices are equal. These would be described as conjuncts of a call and a put. A spread has two prices whereas a straddle has only one:

$$\mathrm{SPREAD!!}(x,y,z,n_1,n_2,t_1) ::= \mathrm{CALL!!}(x,y,z,n_1,t_1) \ \& \ \mathrm{PUT!!}(x,y,z,n_2,t_1)$$

$$\mathrm{STRADDLE!!}(x,y,z,n_1,t_1) ::= \mathrm{SPREAD!!}(x,y,z,n_1,n_1,t_1)$$

Insurance

Insurance is a promise contingent upon some change of state in nature, rather than an action controlled by one of the parties to the promise. Let Q1 be a temporally unbound formula describing the event (e.g., Earthquake( ), Fire( ), Flood( )), and let t_1 be the time in which the insurance is valid. Let Q2 be a formula describing the payment by the insurer if the event occurs. Then the general structure of an insurance policy is as follows:

$$\begin{gathered}\mathrm{INSURANCE!!}(x,y,Q1,Q2,t_1) ::= \\ (\exists p)\ \mathrm{PROMISE!!}(x,y,p)\ \& \\ p = [(\forall t_2){:}\ [\mathrm{PT}(t_2,t_1)\ \&\ \mathrm{RT}(t_2)\ Q1] \rightarrow \\ {}^{\wedge}[O(x,y){:}\ \ (\exists t_3)\ \mathrm{Beg}(t_3)=\mathrm{End}(t_2)\ \&\ \mathrm{RD}(t_3)\ Q2]]\end{gathered}$$

Reading: x makes some promise to y that for any time t_2 on t_1 wherein Q1 is realized throughout, then it is obligatory following t_2 that Q2 be realized.

For instance, suppose party x writes insurance for party y against a fire in some building z for the appraised amount of the damage up to a maximum limit of $100,000. We assume a numeric function, Min(nx,ny), which returns the smaller of its two numeric arguments, and another numeric function, Appraisal(z), which returns the dollar amount of the fire damage. Then this fire insurance policy is specified as follows:

$$\text{FIRE-INSURANCE!!}(x,y,z,n_1,t_1) ::=$$
$$Q1 \leftrightarrow \text{Fire!!}(Z)\ \&$$
$$Q2 \leftrightarrow [(\exists n_2)\ n_2 = \text{Min}(\text{Appraisal}(z),\ n_1)\ \&$$
$$(\exists m_1)\ \$\$(m_1)=n_2\ \&\ \text{OGIVE!!}(x,y,m_1)]$$

Easements, Licenses

Easements and licenses are promissory objects involving permission rather than obligation. Easements are the "rights" of persons other than the owner in the use of real property (land). Presumably these rights are restricted to some particular actions or activities. If not, we would characterize the unrestricted right as possession and view the easement as a rental contract or lease.

Typical kinds of easements are permissions to drive on the property, to have a building located on it, etc. These would not constitute full possession in that such other activities as extracting oil or minerals, growing crops, etc. are usually not included in this permission.

Let Q be the allowed activity. Then the granting of an easement by x to y on the property x over the time period t_1 is as follows:

$$\text{EASEMENT!!}(x,y,z,Q,t_1) ::=$$
$$(\exists p)\ \text{PROMISE!!}(x,y,p)\ \&$$
$$p = {}^{\wedge}[P(x,y){:}\ RD(t_1)\ Q]$$

Reading: x makes a promise to y that y may (but doesn't have to) realize (one or more times) during the activity Q during the time period t_1.

A license, at least as we understand it here, is the general case of an easement. That is, it is the licensor's (promissor's) permission to the licensee (promisee) to perform certain actions that normally would be forbidden. This permission is not restricted to rights to use real property.

For instance, a common type of license is for patent rights. In this case, the licensor allows the normal patent protection to be suspended for the licensee.

Again, let Q be the activity permitted, and t_1 be the period of this permission. The general form of a license is then:

$$\begin{gathered} \text{License!!}(x,y,Q,t_1) ::= \\ (\exists p)\ \text{PROMISE!!}(x,y,p)\ \& \\ p = {}^{\wedge}[P(x,y){:}\ RD(t_1)\ Q] \end{gathered}$$

Reading: x makes a promise to y to the effect that y may do Q repeatedly during time t_1.

Debt Instruments

Loans as we discussed them in the earlier section were regarded as a particular promise (to pay cash) from one individual to another. Loans of this type, especially when the period of the promise is less than 5 years, are usually called *notes*.

Bonds are another type of loan. Usually these are for a period longer than five years. The promissor in these cases is generally an economic organization, e.g., a corporation or governmental body, rather than a person. The promisee (bond holder) in these cases may however be either type of economic actor. Also, bonds usually occur as a collection of promises to a number of parties. The collection is referred to as a *bond issue*. The elements of each collective bond issue have a common agent, starting date and terms of payment. They differ in the technicality that different money is promised in each bond, though the *amount* of the money is the same, and that the recipients may be different in each case.

Two major classes of bonds are distinguished based on how the recipients are identified. A *registered bond* is one where the bond issuer maintains a record of each recipient. The bond can only be transferred by the endorsement of the issuer. A *coupon bond*, on the other hand, is payable to the "bearer." This is the more frequent form, comprising 90% of all bonds.

But the concept of "bearer" raises the interesting and potentially knotty question, "bearer of what?". Our treatment of financial instruments thus far has regarded them as abstract objects, what we have called "promissory" objects. The physical representation (document) on which this promise is expressed has so far not been of importance.

If we consider only the promissory object, we would view the promise to be made to some indefinite recipient who is the owner of that promise on some given date. Thus, the promisee would be indicated within the elaboration of the promise as its owner as of some future date:

$$p = {}^{\wedge}[O(x,y): (\forall w): RD(t_1)\ OWN(w,p) \rightarrow$$
$$(\exists m_1)\ OGIVE!!(x,w,m_1)$$

Here the promise p is the obligation that for whoever owns p. x will give them m_1 (some money).

This however is a logical anomaly, a so-called "self-referring" expression. Substitution of p in the argument of OWN here leads to an infinite regress.

In addition, there is a pragmatic problem with this definition. The promissory object, p, is merely an artifice; an abstraction without physical reality. Given that many people might claim to be the owner of this promise on the date t_1, how is the company to identify which is the real one? In the case of coupon bonds (or any bearer bonds for that matter), the issuer generally does not keep a record of the promisees. The whole point of a coupon bond is to be able to trade them without notifying the issuer. How, then, does the issuer know who to pay? The actual mechanism involved is a book containing physical coupons, one for each promised payment. These coupons operate effectively as post-dated checks of specified amounts, but with the recipient left unspecified. After any particular date is reached, the holder of this book removes the appropriate coupon and cashes it at a bank. This physical book is thus an "authoritative document" in that its purpose is not only *informative*, containing information which can copied as is the case with other information objects, but also *performative*, in that the promissory object in this case is identified with this unique physical object. Note that this performative aspect cannot be reproduced in a photocopy (except under false pretense). Designating this book by the variable x, the previous formula would now read:

$$(\iota\ x)\text{: AUTH-DOC}(z,\ p)\ \&$$
$$p = {}^{\wedge}[O(x,y)\text{: } (\forall w)\text{: } [RD(t_1)\ \text{OWN}(x,z) \rightarrow$$
$$(\exists m_1)\ \text{OGIVE!!}(x,w,m_1)$$

Thus, at least in the case of coupon bonds, any change in ownership of the promissory object must also be accompanied by a corresponding change of ownership of the coupon book. This is expressed:

$$(\forall p)\ \text{COUPON-BOND}(p) \rightarrow (\iota\ z)\ \text{AUTH-DOC}(z,p)\ \&$$
$$(\forall x)\ (\forall y)\ \text{OCHANGE!}(x,y,p) \rightarrow \text{PCHANGE!}(x,y,z)$$

Note that here we are describing subsequent trading of the coupon bond. The original issuance of this bond would be as follows. Let us presume that the bond involves n_3 equal size interest payments of size n_1, paid over intervals of length t_2, and that on the final interest payment the principal in the amount n_2 is repaid.

$$\begin{gathered}
\text{COUPON-BOND-ISSUE!!}(x,y,t_0,t_1,t_2,n_1,n_2,n_3) ::= \\
(\iota z)\ (\exists p)\ \text{PROMISE!!}(x,y,p)\ \&\ \text{AUTH-DOC}(z,p)\ \& \\
\text{OGIVE!!}(x,y,z)\ \& \\
p = {}^{\wedge}[O(x,y) \colon [1 \leq i \leq n_3] \colon \\
(\exists t_3[i])\ (\exists t_4[i])\ (\exists m_1[i]) \colon \\
\text{Beg}(t_3[i]) = \text{End}(t_0)\ \&\ \text{Dur}(t_3[i],t_3) = i\ \& \\
\text{Beg}(t_4[i] = \text{End}(t_3[i])\ \&\ \text{Dur}(t_4[i],t_2) = 1\ \& \\
\$\$(m_1[i]) = n_1\ \& \\
(\forall w[i]) \colon [\text{RT}(\text{Beg}(t_4[i]))\ \text{OWN}(w[i],x))] \rightarrow \\
[\text{RD}(t_4[i]\ \text{OGIVE!!}(x,w[i],m_1[i]))]] \\
\& \\
[(\exists t_5)\ (\exists m_2) \colon t_5 = t_4[n_3]\ \& \\
\$\$(m_2) = n_2\ \& \\
(\forall v);\ [\text{RT}(\text{Beg}(t_2))\ \text{OWN}(v,z)] \rightarrow \\
[\text{RD}(t_5)\ \text{OGIVE!!}(X,v,m_2)]]]
\end{gathered}$$

This is read as follows. First there is an ownership change of the authoritative document, x. Next, the obligation decomposes into two major bracketed expressions. The first expression involves an iteration. The reading of each iteration is as follows. $t_3[i]$ is the time preceding the ith iteration. It therefore begins as t_0 ends and is i times t_1 intervals long. $t_4[i]$ is the time span covering the ith interval. It begins as $t_3[i]$ ends and is t_2 long. Then, for some money in the amoung n_1, and for any person who owns the coupon book z *at the beginning point of this time*, x is obliged to give them this money during this time.

The second bracketed expression is structurally similar but without the iteration. Whoever owns the coupon book at the beginning of the kth iteration also gets the princial.

Another important concept with regard to bonds is that of *collateral*. As we have mentioned earlier, promises of the sort considered here are generally enforceable by a legal process of some governmental body. With certain debts, however, a more specific enforcement is included in the terms of the contract, namely the lender's privilege to take ownership of some asset in the event of the borrower's default. Thus the borrower's promise includes an *obligation* to the lender to pay some amount(s) of money, as well as the borrower's *permission* to the lender to take some asset if the payments are not made. Note that the lender does not have to take the asset, but may. To express this, we make use of the connective OE ("or else") developed earlier for deontic expressions.

Let x and y be respectively, borrower and lender. Let Q be a temporally unbound formula indicating x's promised payment actions and let t_1 be the period in which Q us supposed to occur and let x be the collateral object. A collateral promise might be as follows:

$$\text{COLLATERAL-PROMISE!!}(x,y,Q,t_1,z) ::=$$
$$(\exists p)\ \text{PROMISE!!}(x,y,p)\ \&$$
$$p = {}^{\wedge}[[O(x,y)\colon RD(t_1)\ Q]\ \text{OE}$$
$$[P\ (\forall t_2)\colon Beg(t_2)=End(t_1)\ \&\ RD(t_2)\ \text{OTAKE!!}(y,x,z)]]$$

The promise reads as follows: for all times t_2 following t_1 it is obligated to realize during t_1 the action Q; or else it is permissible that y takes ownership of the object z from x.

Equity Instruments

Equity instruments are the various types of corporate stock. The two principal types are *common* and *preferred*. Common stock corresponds most closely with the ordinary concept of "ownership" of the corporation. Each share of common stock permits the holder to one vote in the election of the company's board of directors (usually; there have been exceptions).

Beyond that, however, the stockholder has little direct influence on the firm's everyday operations nor can he/she legally dispose of any of the firm's assets without the permission of the management or board. Common (as well as preferred) stockholders are not responsible for the corporation's debts. If the firm goes bankrupt, creditors have no claim to the stockholder's personal estate.

If the firm is liquidated without bankruptcy, common stockholders have a residual claim to the assets--they get whatever is left after all debts have been satisfied as well as whatever claims preferred stockholder's might have.

We find this to be a quite different form of "ownership" than the others we have considered. For that reason, we have expressly *excluded* it in the definition of our OWN predicate. While stockholders are seen to OWN their stock, they are *not* seen to OWN the corporation itself. Rather, the stock is regarded as a promise, essentially no different than the promises involved in debts, to which the corporation has a commitment.

The details of these promises are rather vague however. Roughly, they are contingent obligations on the part of the firm to eventually distribute cash dividends, and/or accumulate valuable assets within the firm which may be eventually converted to cash on liquidation. Seldom, if ever, are these commitments ever articulated however. (Certainly they exist or else the stock would have no value.)

Given the vagueness and complexity of the corporation's agreement with its stockholders, we are forced (at least for the moment), to accept this as a primitive type of promise, viz. COMMON-STOCK. Thus for a corporation, c, and a stockholder, x, we would describe their relationship as follows:

$$(\exists p)\ \text{PROMISE!!}(c,x,p)\ \&$$
$$p = {}^{\wedge}[O(x,y)\text{:}\ \text{COMMON-STOCK}]$$

(Recall that by the definition of PROMISE!!, x afterwards OWNs p.)

Preferred stock is conceptually something of an intermediate category between bonds and common stock. It often does not have voting privileges, and sometimes is only contingently voting, e.g., only under certain adverse circumstances. In the event of liquidation, preferred stockholder's claims come after those of bond holders but before common stockholders. Also, the nature of the firm's promise is usually more definite with preferred stock than with common, but usually contains contingency provisions not found in bonds.

There is a wide range of variations written into the terms of preferred stock issues. Often there is a fixed dividend rate set, which is payable provided the firm realizes adequate earnings. Sometimes this dividend obligation is made *cumulative*, so that a missed dividend one period is added to the dividend promised for the following period. Other terms are also variously included, such as call and sinking fund provisions allowing the firm to retire this stock if it chooses.

Unlike bond holders, preferred stockholders cannot legally enforce arrearages in dividends, though these dividends do take priority over dividends to common stockholders. This lack of legal enforcement is problematic in CANDID, since we have presumed that our deontic operators have the force of law. To give an example of what a preferred stock might look like in CANDID, let us assume a firm, x, writes a preferred stock to a party, y, promising a cumulative dividend interval t_1 (e.g., every year) in the amount n. Assume the stock is issued in time t_0 and that any dividends paid will be paid within t_2 (e.g., a month) time following the end of the operating interval t_1 (e.g., the fiscal year end). The notion of a dividend contingent on adequate income, would also necessitate an event predicate, Income!(x), which would test for sufficient income.

$$
\begin{gathered}
\text{CUMULATIVE-PREFERRED STOCK!!}(x,y,t_0,t_1,t_2,n) ::= \\
(\exists p)\ \text{PROMISE!!}(x,y,p)\ \& \\
p = {}^{\wedge}[O(x,y)\!:\ [1 \leq i \leq *]\!: \\
(\exists m_1[i])\ \$\$(m_1[i])=n\ \& \\
(\exists t_3[i])\ (\exists t_4[i])\!: \\
\text{Beg}(t_3[i])=\text{End}(t_0)\ \&\ \text{Dur}(t_4[i],t_1)=(i-1)\ \& \\
\text{Beg}(t_3[i])=\text{End}(t_3[i])\ \&\ \text{Dur}(t_4[i],t_1)=1\ \& \\
(\exists t_5[i])\ \text{Beg}(t_5[i])=\text{End}(t_4[i])\ \& \\
\text{Dur}(t_5[i],t_2)=1\ \& \\
(\exists t_6[i])\ \text{Beg}(t_6[i])=\text{End}(t_4[i])\ \& \\
[\text{RT}(t_4[i])\ \text{Income}(x)] \rightarrow
\end{gathered}
$$

$$[RD(t_5[i])\ OGIVE!!(x,y,m_1[i])]\ \&$$
$$\sim[RD(t_4[i])\ Income!(x)] \rightarrow$$
$$[RD(t_6[i])\ OGIVE!!(x,y,m_1[i])]]$$

Reading: on each of an indefinite number of iterations, it is obligatory that for some money in the amount n, and for times t_4 (e.g., the current year), t_5 (a short period following t_4) and t_6 (an unlimited period following t_4), if there is income in t_3 x must pay the dividend during t_5; if there is no income in t_4, x must pay the dividend during t_6.

Convertibles

Certain bonds and preferred stock are "convertible." This means that the holder has the option to exchange them for the issuing company's common stock at some specified exchange rate. This option aspect of convertibles is structurally similar to that of puts and calls. We describe this convertible aspect as a *separate promise taking the form of an option to exchange the* current promise, p_1, by the company (that of the bond or preferred stock) for another promise, p_2, (that of common stock).

Let us assume for issuer x and holder y this option applies for the period t_1 and that y must respond within t_4 amount of time. Then, the issuance of this option would be as follows:

$$CONVERTIBLE\text{-}OPTION!!(x,y,p_1,p_2,t_1,t_2) ::=$$
$$PROMISE!!(x,y,p_1)\ \&$$
$$p_1 = {}^{\wedge}[O(x,y): (\forall t_3)\ (\exists t_4):$$
$$PT(t_3,t_1)\ \&\ Beg(t_4)=End(t_3)\ \&\ Dur(t_4,t_2)=1\ \&$$

$$[RT(t_3)\ OGIVE!!(y,x,p_1)] \rightarrow$$

$$[RD(t_4)\ OGIVE!!(x,y,p_2)]]$$

Reading: x promises y that if for any time t_3 during t_1, y gives back ownership of the promise p, then x is obliged to give to y the promise p_2 within the time t_4 (of length t_2) which immediately follows.

III-E. CONCLUDING REMARKS

This completes our list of sample financial instruments described using CANDID. The preceding was of course only a tutorial survey illustrating how CANDID can be used to represent financial and commercial concepts.

As indicated in the introduction, the motivation behind the development of this calculus was to serve as a representation language for knowledge bases in artificially intelligent managerial decision support systems.

Definitions such as these would therefore serve as the basis for inferencing in decision aiding applications for instance in evaluating a firm's financial statements, evaluating financing alternatives, verification and monitoring of contracts, etc.

Also, the implementation of this language in a deductive computer system would assist in the verification of the definitions. Even at this tutorial level, some of the definitions approached a level of complexity that was difficult to follow. As further, more detailed concepts are included, mental verification would become even more difficult, and the assistance of the computer in this process would be useful.

REFERENCES

Bonczek, R.H., C.W. Holsapple and A.B. Whinston. 1981. Foundations of Decision Support Systems. New York: Academic Press.

Brachman, R.J. and B.C. Smith. eds. 1980. Special Issue on Knowledge Representation. *SIGART NEWSLETTER* No.70. Association for Computing Machinery.

Buchanan, B.G. and Feigenbaum, E.A. 1978. Dendral and Meta-Dendral: their applications dimension. *Artificial Intelligence*, 11(1,2):5-24.

Chang, C.C. and H.J. Keisler. 1973. Model Theory. Volume 73, A. Heyting, *et al.* (eds.) Studies in Logic and The Foundations of Mathematics. Amsterdam: North=Holland, and New York: American Elsevier Publishing Co. Inc.

Cresswell, M.J. 1973. Logics and languages. London: Nethuen & Co. Ltd.

Dowty, D.R. 1978. A Guide to Montague's PTQ. Indiana: Indiana University Linguistics Club.

Enderton, H.B. 1972. A Mathematical Introduction to Logic. New York: Academic Press.

Fick, G. and R.H. Sprague, Jr. eds. 1980. Decision Support Systems: Issues and Challenges. Oxford: Pergamon Press.

Kalis, D., R. Montague and G. Mar. 1980. LOGIC Techniques of Formal Reasoning, Second Edition. New York: Harcourt Brace Jovanovich, Inc.

Keen, P.G.W. and M.S. Scott Morton. 1978. Decision Support Systems: An organizational Perspective. Reading, Massachusetts: Addison-Wesley Publishing Company.

Kripke, S.A. 1963. Semantical considerations on modal logics. F2Acta Philosophica Fennica, Modal and Many-valued Logics pp.83-94.

Lee, R.M. 1980. CANDID—A Logical Calculus for Describing Financial Contracts. Philadelphia, Pennsylvania: Dept. of Decision Sciences, The Wharton School, University of Pennsylvania.

Martin, W.A. and R.J. Fateman. 1971. The MACSYMA system. Proceedings of ACM 2nd Symposium on Symbolic and Algebraic Manipulation. Los Angeles, California. pp.23-25.

Montague, R. 1973. The Proper Treatment of Quantification in Ordinary English. Chapter 8, R.H. Thomason. ed. Formal Philosophy—Selected Papers of Richard Montague. 1974. New Haven and London: Yale University Press.

Rescher, N. 1975. A Theory of Possibility—A Constructivistic and Conceptualistic Account of Possible Individuals and Possible Worlds. Pittsburgh: University of Pittsburgh Press.

Rescher, N. and A. Urquhart. 1971. Temporal Logic. Wien: Springer-Verlag.

Shortliffe, E.H. 1976. Computer-Based Medical Consultations: MYCIN. New York: American Elsevier.

Strawson, P.F. 1959. Individuals—An Essay in Descriptive Metaphysics. Garden Cit, New York: Doubleday & Company, Inc. Anchor Books.

Thompson, B.G. 1981. Ethereal Goods—The Economic Atom of the Information Society. Paper prepared for the EEC Conference on The Information Society, Dublin, Ireland, 18-20 November 1981.

Wittgenstein, L. 1921. Tractatus Logico-Philosophicus. English translation by D.F. Pears & B.F. McGuinness, 1961, London and Henley: Routledge & Kegan Paul.

Wittgenstein, L. 1953. Philosophical Investigations. (Third Edition, 1958) New York: Macmillan Company.

Woodward, J. 1978. Management and Technology. In D.R. Hampton, C.E. Summer and R.A. Webber, Organizational Behavior and the Practice of Management. pp.344-353. Scott, Foresman & Co.

van Fraasen, B.C. 1971. Formal Semantics and Logic. New York: The MacMillan Co.

von Wright, G.H. 1965. And Next. Acta Philosophica Fennica, Fasc. XVIII, Helsinki. pp. 293-301.

von Wright, G.H. 1967. The Logic of Action—A Sketch. in N. Rescher, ed. The Logic of Decision and Action. pp.121-136. Pittsburgh: University of Pittsburgh Press.

von Wright, G.H. 1968. An Essay in Deontic Logic and the General Theory of Action. Acta Philosophica Fennica, Fasc. XXI. Amsterdam: North Holland Publishing.