

工業所有権法の知識表現システム KRIP†

新田 克己†† 長尾 順太郎††† 水鳥 哲也††††

工業所有権法（特許法，実用新案法，意匠法，商標法等）のエキスパートシステムを構成するためのツール KRIP と，KRIP の中核をなす記述言語 KRIP/L，および，KRIP 上に構築された特許法エキスパートシステムを紹介する。KRIP/L はオブジェクト指向の概念と区間論理式を統合した言語であり，主に，手続きについて定めた法律（手続法）の記述を行うために開発されたものである。手続きをするためには，手続法に規定する特定の条件（手続き要件）が満たされなくてはならない。また，手続きをすることによって，新しい権利の発生や状況の変化（手続きの効力）がおこる。KRIP/L は，手続き要件を区間論理式で表現し，効力をオブジェクトのメソッドとして表現することにより，手続き関係の自然な記述をすることができる。KRIP は KRIP/L で知識ベースを作成するためのエキスパート支援系と，知識ベースを利用するためのユーザ支援系からなっている。パーソナルコンピュータ上の Prolog で実現されており，マウスを用いて，知識の入力や利用が行える。特許法エキスパートシステムを KRIP で作成し，KRIP システムによる違法性のチェックの実例を示した。

1. ま え が き

法律のエキスパートシステムを作成する場合，その知識が法令文という一つの論理体系で表現されているために知識ベースを作成することは容易であると考えられがちである。しかし，実際に法律エキスパートシステムを構築するには，その法律を理解するための予備知識が存在するという問題と，知識表現言語の表現能力が十分ではないという問題を解決しなければならない。

我が国では法令文の知識表現として，法令文を論理式に変換する方法や，フレームとして表現する方法等が研究されてきた^{3)~6)}。しかし，工業所有権法（特許法，実用新案法，意匠法，商標法など）には，性質の異なる条文が混在し，時間の要素が多いことから，他の法律と異なる視点からのアプローチが必要である。

ここでは工業所有権法の知識表現システム KRIP (Knowledge Representation System for Laws Relating to Industrial Property) について述べる。このシステムは工業所有権法の知識ベースシステムを作成するために作られたものであり，Prolog でインプリメントされている。第2章では工業所有権法の特徴をあげ，第3章では手続法記述言語 KRIP/L の概要を

示す。第4章では KRIP/L をサポートする知識表現システム KRIP を説明し，第5章では KRIP 上に構築された特許法エキスパートシステムについて述べる。

2. 工業所有権法の特徴

工業所有権法は，発明を保護する特許法，考案を保護する実用新案法，意匠を保護する意匠法，商標を保護する商標法，および，外国出願のための条約類の総称である（そのほかに政令や省令も含めて工業所有権法を構成する）。

まず，工業所有権法の特徴をあげ，エキスパートシステムを実現するための問題点を明らかにする。

2.1 実体法と手続法の混在

「実体法」は社会生活上の規範そのものであり，「手続法」は規範の実現に国家が力をかす手続きであると定義される。

実体法は物事の判断基準を規定するがこれだけでは「絵にかいた餅」にすぎない。事実関係を実体法にあてはめ，その内容を実現するまでの手続きを規定するのが手続法である。例えば，刑法（実体法）と刑事訴訟法（手続法）のように両者は一体となって用いられる。裁判において，刑事訴訟法の手続きに従って事実認定が行われ，認定結果をもとに刑法の定めによって刑の決定がなされるのである。

工業所有権法においては，ある発明が特許（登録）を受けられるための条件を規定した条文が実体法であり，その条件を満たしていることを審査してもらうための手続きを規定した条文が手続法である。その他に条文を理解するための基本となる知識（用語の定義や

† KRIP: Knowledge Representation and Inference System for Laws Relating to Industrial Property by KATSUMI NITTA (Information Systems Section, Software Division, Electrotechnical Laboratory), JUNTAO NAGAO (Development Laboratory, Nippon Business Consultants Co., Ltd.) and TETSUYA MIZUTORI (Power Technology Section, Substation Division, Meidensha Co., Ltd.).

†† 電子技術総合研究所ソフトウェア部情報システム研究室

††† (株)日本ビジネスコンサルタント開発研究所研究課

†††† (株)明電舎変電技術部電力技術課

手続きをするための資格等)を定めた条文を「総則」という。

手続法では種々の手続きによって、あいまいな事実関係を確定する役割を持つ。実体法のいくつかは手続法によって陽に起動され、このような場合に限れば手続法は実体法のメタ知識と考えることができる。

手続法は、手続きを行うための「前提条件」を定めた部分と手続きの「効力」を定めた部分に分類することができる。前提条件には例外規定が多い。

工業所有権法では、実体法と手続法では違法性のチェックにおける性質が次のように異なる。

1) 実体法の条文は「～ならば～である」の形式で書かれたものが多く、チェックは論理式による定理証明の形で行える。それは定理証明の前件となるデータは手続法によって確定したものであるからである。

2) 手続法の条文は手続きの効力に関し、次のような問題がある。

手続きがなされると、その手続きをするための前提条件が審査され、正当であれば手続き時にさかのぼって効力が発生し、正当でなければ手続きは無効となる。したがって、「手続きをしたという事実」があっても、法的に認められなければ、手続きがなかったものとされる。このように、手続法では、事実をそのままデータとして使うことができず、法的に存在が認められた事実だけをデータとして使うことができる。

また、手続きの前提条件の審査においては、法的に認められた事実だけではなく、その事実によって生じた効力(例えば、ある手続きをした結果、権利や資格等が生じる)も含めて審査の対象となる。

したがって、ある手続きが適法になされたか否かは、その手続きに係わる事実からだけでは判断することはできず、過去からその手続きに至るまでの履歴とその効果の蓄積を考慮して初めて判断できるのである。後述のように、法律的な効果が過去にさかのぼって発生することがあり、事実認定と時間との関係が複雑なものになることがある。

2.2 時 間

工業所有権法には時間に関する規定が多い。例えば、

1) 特許権には存続期間が定められており、それを経過すると消滅する。

2) 手続きによっては手続き期間が定められ、その期間以外はその手続きはできない。

等の規定がある。

また、以下にあげるように手続きの効力が必ずしも手続き時と一致しないという大きな問題がある。

1) 明細書の補正が行われたときには、初めからその明細書で出願が行われたものとみなされる(効力が過去にさかのぼって発生する)。

2) 特許無効の審決が確定すると、その特許は初めからなかったものとみなされる。

さらに、対象物の内部状態が時間とともに変化するため、判断の基準時によって判断結果が異なることがある。

2.3 法律の解釈

法令文は自然言語のなかでは論理的にかかっている方ではあるが、それを解釈するには条文には表れない暗黙の知識が必要である。

1) 自然言語はあいまいな点があり、法律設定時には予想もしなかった解釈が生じることがある。解釈には学説が分かれることがあり、判例が確定しないときには語句の厳密な意味や法目的との関連から解釈される。すなわち法の文書は単なる自然言語ではなく、その裏にそれを解釈するための知識が必要である。

2) 工業所有権法のある部分は民法の特別法となっている。民法の一般原則や慣習、さらには法律家の持つメタ知識(法令文の運用の知識等)を用いなければ法律の解釈はできない。したがって、工業所有権法はそれだけで閉じた法体系ではない。

3) 法令文は局所的に見ると論理的であっても、大局的に見ると論理の整合がとれていないことがある。例えば、民法の規定と特許法の規定が矛盾を生ずるときには特別法である特許法が優先され、また、原則的な規定と例外規定が矛盾を生ずるときには例外規定が優先される。このように、法令文には優先順位がついており、複数の法令文が矛盾を持つときには優先度の高いほうが適用される。この優先度も法律解釈のための一つのメタ知識と考えられる。

3. 工業所有権法表現言語 KRIP/L

工業所有権法の条文は手続法記述言語 KRIP/L で表現される。KRIP/L は手続法の記述を最大の目標とし、あわせて実体法および総則との関係記述をねらうものである。そのために論理型言語を基に、時間と様相とオブジェクトを扱えるように拡張を行い、法律概念と条文を別形式で記述することとした。まず、手続きを扱うモデルを説明し、その後で KRIP/L の概要を説明する。

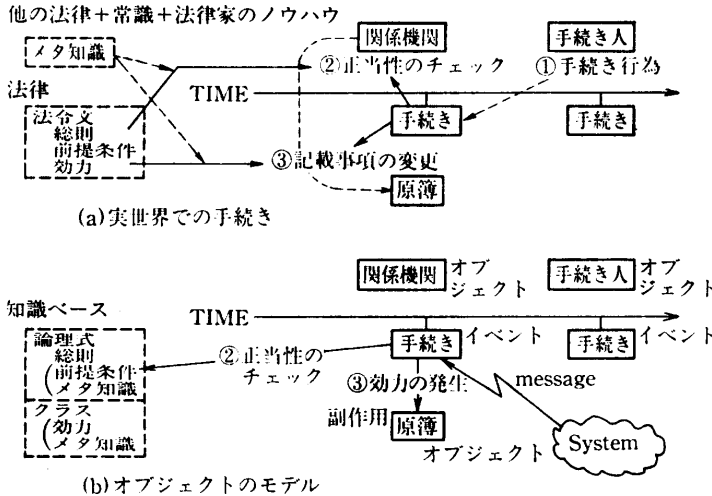


図1 オブジェクトによる手続きのモデル
Fig. 1 Object model of proceedings.

3.1 手続きのモデル

実世界では①手続きがなされると②関係機関によってその正当性が審査され(手続きの前提条件の規定を参照する), 正当であれば③効力が手続き時にさかのぼって発生する(手続きの効力の規定による). 手続きの効力によって新しく権利が発生することもあるし, 他の対象物の内部状態を変更することもある(図1(a)).

これを手続きを「オブジェクト」と対応させたモデルで考える. オブジェクトは人や物, 権利, 手続き等の対象物に対応する. 他のオブジェクトに副作用を及ぼすオブジェクトを特に「イベント」と呼ぶ. イベントはそれが起こった時刻で時間のチェーンにつながれる. 各イベントにはその手続きがなされた年月日や手続き人, 手続き内容等の内部データのほかに, その正当性を裏付けるための手段(どの条文をどの順序で調べるか)やどのような効力を他に及ぼすかがメソッドとして記述されている. このイベントにメッセージを送ると, イベント内のメソッドが起動されて正当性を判断し(手続きの前提条件の参照), 正当ならば新たにオブジェクトを生成したり他のオブジェクトに副作用を及ぼしたりするのである(手続きの効力の発生)(図1(b)).

手続きの正当性を証明するための条文や効力を規定した条文は論理式の形で与えられ, イベント中のメソッド部分から参照される.

オブジェクトの内部状態はイベントからの作用により変化し, それ以外の原因では変化しない. 内部状態

の変更があったときでも, 過去の状態の履歴は保存されるので, 過去の任意の時点での状態を知ることが可能である.

オブジェクトモデルによる手続きの扱いは, 次のことを想定している. まず, 調べたい事例をイベントやオブジェクトの集合として表して事実データベースに格納しておく. 次に, 古いイベントから順にメッセージを送って, その効果をオブジェクトの内部状態の変化として蓄積することによって実際の効果のシミュレーションをする.

3.2 法律概念に関する知識

工業所有権法に係わる概念(手続き, 法律用語等のKRIPで用いられる語い)はクラスを用いて表される. クラス是一群のオブジェクトに共通の性質を記述したものである. クラス間の関係はIS-Aによる階層構造を構成し, 知識の継承が行われる. その構文は以下のようなものである.

```
class <クラス名>.
    super: <上位クラス>    :: <根拠条文>
    スロット: <値> OR {<タイプ>}
                                :: <根拠条文>
    .....
    メッセージ-><Prologのゴール列>
                                :: <根拠条文>
    .....
end class.
```

スロットは内部状態の項目(すなわち, 内部変数)であり, 右辺には実際の値かその値が満たすべきタイプが記述される. スロットは異なった時間に異なった値を持つことができ, 値の履歴をリストとして保持している. メッセージ(セレクタ+引数)は受けつけるメッセージを表し, それが送られると右辺のゴール列が実行される. いずれも上位のクラスからの継承がある. 根拠条文の識別子は

section (特施規 3, 1, 1)

section (民訴 2, 0, 1)

のように<条文番号, 項番号, 論理式の識別番号>の組で与える.

例えば, 出願審査請求という手続きは

```
class 出願審査請求
```

```

super : 特許庁長官への手続き
手続き人 : {手続き能力者} :: section(...).
手続き時 : {年月日}
審査対象 : {出願実体}
constraint(X)→type_match(...), ...
legal_check→legal (self,
                    section (特 48-3, 1, -)), ...
effect→exec (特 48-2, self # time), ...
end_class.

```

のように記述される。出願審査請求は特許庁長官への手続きの一つであり、三つの内部変数（手続き人、手続き時、審査対象）と三つのメソッド（constraint, legal_check, effect で起動されるもの）からなっている。

内部変数にはその値が満たすべきタイプを記述することができる。例えば、「手続き人」の値となりうるものは手続き能力者（未成年者、禁治産者等でないもの）でなければならないこと等が記述されている。

constraint はこの手続きが正当であるために各スロットの値が満たすべき拘束条件を記述し、legal_check はこの手続きが行えるための前提条件はどの条文をどの順序で調べるべきかを記述し、effect はこの手続きの効力を記述している。オブジェクトを発生させたり、副作用によりオブジェクトのスロットの値を変更させたり、ほかのオブジェクトにメッセージを送ったりするためのシステム述語が用意されている。

```

legal_check において、
legal_check→legal (α), legal (β).
legal_check→legal (α); legal (β).
legal_check→not (legal (α)), legal (β).

```

はそれぞれ、「α と β の二つの条文が満たされたときに適法である」、「α と β のいずれかの条文が満たされたときに適法である」、「α が満たされず、β が満たされたときに適法である（α は β の例外規定である）」の意味である。列挙された条文は左から右へ Prolog プログラムと同様に実行され、バックトラックの機能やカットも使うことができる。これらの情報は、この手続きの正当性を調べるのに、どの条文が関連し、どの条文が優先度が高いか（どの条文を先に調べるべきか）を表している。

3.1 節で述べたように、オブジェクトは人や物や手続きのほかに、権利や状況を表すのに用いられる。また、一連の手続きがなされることによる状況の変化は、状況に対応したオブジェクトの内部状態の変化と

して表すことができる。

例えば、借金をすると債権が生じ、返済すると債権が消滅することは、以下のように記述できる。

```

class 借金.
.....
effect→o_create (...債権...).
end_class.
class 返済.
.....
effect→delete (...債権...).
end_class.

```

原則として、法令文に現れるすべての名詞はクラスで記述されるが、その概念が流動的なもの（クラス名そのものが流動的なもの）はクラスと論理式の両方で記述される。例えば、「発明」は「技術思想」の中で特定の条件（発明の構成要件）を満たしたものであると定義されている（特許法2条）。要件を満たしているかどうかは人によって判断が異なることがあり、また、時間とともに判断が変化することがある。したがって、ある発明がなされたことを表現するには、オブジェクトとしては「発明」のインスタンスではなく、「技術思想」のインスタンスを発生させ、その中のスロットの値が特定の条件を満たすときだけそれを発明であるとみなすのである。

3.3 条文に関する知識

条文は Templog⁷⁾で採用された区間論理式で記述する。条文には総則、実体法、手続法の前提条件部分、手続法の効力部分等の性質の異なるものがあるので、それに応じた記述形式が望まれる。

また、法令文は「第2条3項」のように「条」と「項」で区切られているが、一つの手続きの記述が複数の条に分かれて記述されていることがある。条文はなるべく「条」ごとに閉じた表現をとるが、それより細かい「項」や「号」の区切りはあまりこだわらずに記述することにする。

これらを考慮し、各条文の記述は以下のように識別子と節の対で行う。

```

section <根拠条文>.
    条文節.

```

条文節の一般形は、

- 1) Prolog の節
- 2) [A : B]=> P :-<BODY>.
- 3) [A : B]=>~P :-<BODY>.
- 4) T@ P :-<BODY>.

- 5) T@~P :-<BODY>.
 6) [A : B]→may Q :-<BODY>
 7) [A : B]→must Q :-<BODY>
 8) [A : B]→mustnot Q :-<BODY>

のいずれかである。

2) と 3) は「イベント A が発生してからイベント B が発生するまでの期間中、BODY が成立するならば P の内容が成立する (成立しない)」ことを表す。A が無限の過去に発生し、B が無限の未来に発生するならば、[A : B] による期間的制限はなくなり、1) と同じ意味になる。1) の形式の節では、あるゴール G が成立することを証明するには

?-call(G). (または ?-G.)

を実行するが、2) の形式の節では、基準となる時刻 T を指定して

?-call(G, T).

を実行する (4.2 節を参照)。

4), 5) は「BODY が成り立つときに限り、P が成立する (成立しない)」ことを表す。これは 2), 3) の特殊形であり、期間 ([A : B]) を明示しない記述法である。7) と 1) の違いは 7) の BODY には時間的に変化する要素 (時間を引数とする述語) が含まれていることである (内部的には 4.2 節で説明するコンパイルの方法が異なる)。

6)~8) は「イベント A が発生してからイベント B が発生するまでの期間中、BODY が成立するならば手続き Q をしてもよい (しなくてはならない、してはならない)」ことを表す。

主に、6)~8) は手続法の前提条件の記述に用いられ、2)~5) は実体法の記述や総則の記述に用いられ、1) は手続きの効力の記述に用いられる (総則と他の規定とは 3.4 節の ext_object 述語で関係付けられる)。

イベントが発生しているか否かを調べるためにシステム述語 event が用意されており、条文節やメソッド中で使うことができる。event は

event (ID, Class, Condition, Time)

の形で用いられ、「ある時刻(Time)に条件(Condition)を満たし、クラスが Class に属するイベントを探し、そのイベントの識別番号を ID にセットする」働きをする。条件は

スロット : 値, スロット : ~値

のように、対象としているイベントのスロットの値が特定の値と等しい (等しくない) ことを指定したり、

スロット : (オブジェクト # スロット)

のように、対象としているイベントのスロットの値が特定のオブジェクトの特定のスロットの値と等しいことを指定したり、一般の Prolog のゴールを記載したりすることができる (Condition は複数の条件をリストにしたものである)。

引数として時刻 (Time) を与えるのは、スロットの値が時刻とともに変化するからである。この引数が未定義変数ならば最新の値で判定される。以下は「61/3/6 に山田氏によってビデオが発明された」ことを表すイベントを検索するための event の記述例である。

event (ID, 発明, [発明者 : 山田,
内容 : ビデオ], 61/3/6)

イベント以外のオブジェクトについては event とほぼ同等の働きをするシステム述語 object が用意されている。以下は、60/5/7 に体重が山田氏の体重と等しく、身長が 170 以上の男性をさがすときの記述例である。

object (ID, 人, [性別 : 男, 体重 : (山田#体重),
身長 : X, X > 170], 60/5/7)

法律では閉世界仮説が成立し、法律に規定していないことは何も生じない。したがって、手続き Q が適法であるためには必ず根拠条文が存在し、具体的には Q が 6) か 7) の条文を満たす必要がある。

同一の対象について 2) と 3) あるいは 4) と 5) または 6), 7) と 8) のように肯定の概念と否定の概念の両方が主張されているときには、矛盾を生じることになる。2), 4), 6), 7) のように、肯定的概念によって権利や手続きの前提条件を規定し、3), 5), 8) によってその例外を規定するのが普通である。したがって、矛盾が生じたときには、3.2 節で述べた legal_check の記述を使って否定の概念を優先させる。

記述例を以下にあげる。これは特許法 33 条 1 項の一部であり、「特許を受ける権利を持つものは特許出願の前にそれを移転することができる」ことを内容とするものである。

section (特 33, 1, 1).

[event (ID, 発明行為, [], _):

event (ID 2, 出願, [請求の範囲 : (ID # 内容), _])

→may 出願前の移転

([主体 : (ID 3 # 主体), 対象 : ID 3])

:-object (ID 3, 特許を受ける権利,

[客体 : (ID # 内容)], self # 手続き時).

3.4 システム述語

上述の event や object 以外に用意されているシステム述語の例を以下にあげる。

1) o_create (ID, Class, Condition, Time)

クラス Class のインスタンスを作り、その名前を ID とする。内部状態は条件 Condition で指定したものとし、生成時刻を Time とする。

2) o_modify (ID, Condition, Time)

オブジェクト ID の内部状態を時刻 Time の時点で変更する。

3) send_msg (ID, Message)

オブジェクト ID にメッセージ Message を送る。

4) apply (SectionNo, Goal, Time)

時刻 Time に特定の事項 Goal が成り立っていることを特定の条文 SectionNo を使って証明する。SectionNo を指定して実行したときはその条文を使って Goal を証明し、SectionNo を未定義のまま実行させたときはすべての条文を使って Goal を証明し、適用された条文の識別子を SectionNo に代入する。手続法から実体法を参照するのに用いられる。

5) legal (ID, SectionNo)

あるイベント ID は SectionNo で指定された条文に関する限り適法である。

6) object_val (ID, Slot, Time, Value)

特定の時刻 Time におけるオブジェクト ID のスロット Slot の値は Value である。

7) limit (Time1, Period, Time2)

ある時刻 Time1 から特定の期間 Period 経過すると時刻 Time2 となる。期限の計算に用いる。例えば、「P という手続きはイベント A のときから 3 か月以内に行うことができる」ことを、以下のように記述する。

[event (A, ...): limit (A # time, 3 month, T)]

→may P :-...

8) ext_object (ID, Class, Condition, Time)

初めに、(i) クラスが Class に一致し、時刻 Time に条件が Condition の指定にあうオブジェクト ID をさがす (通常の object 述語と同じ)。そのようなオブジェクトがないときには、(ii) Class の下位クラス Class' について同じ条件のオブジェクトをさがす。条件にあうオブジェクトがないときには、(iii) Class を擬似クラスとみなし、Class の定義がされている条文をチェックしてからオブジェクトをさがす。

(ii) の例として、

ext_object (ID, 権利, [所有者: 山田], T)

を実行すると、初めはスロット「所有者」の値が山田であるような権利のインスタンスをサーチするが、見つからないときには、権利の下位クラスである特許権について同じ条件のサーチを行う。

(iii) の例として、

ext_object (ID, 権利能力者, [...], T)

を実行すると、初めは権利能力者またはその下位クラスのインスタンスをサーチするが、見つからないときには、権利能力者を規定している条文をさがす。例えば総則に

T @ 権利能力者 (ID) :-

object (ID, 法人, [国籍: 日本], T).

なる規定 (「法人は国籍が日本である間は権利能力者である」を表す) が仮にあったとすれば、権利能力者のインスタンスのサーチの代わりに、国籍が日本である法人のインスタンスのサーチが実行される。

9) invalidate (ID)

ID で指定されたイベントに「無効」の印をつける。以後、システム述語 event や ext_event を起動しても、この印がついたイベントの存在は無視される。

10) retry (Time)

シミュレーションを時刻 Time からやりなおす。例えば、時刻 T1 になされた手続き P が、時刻 T2 になって取り下げられたとする。取り下げの効果は T1 にさかのぼって発生し、P は初めからなかったものとみなされる。T1 と T2 の間に他の手続き Q がなされていたとき、適法になされたと思われた Q が、P の効力が無くなることにより、違法になることがある。このような状況は、P を述語 invalidate で無効にし、述語 retry によってシミュレーションを T1 からやり直すことで表現される。

3.5 KRIP/L による工業所有権法の記述

KRIP/L によって、第 2 章であげた工業所有権法の問題点がいかに解決されるかを述べる。

1) 実体法と手続法 手続法と実体法は論理式の形から区別される。手続きの流れはオブジェクトを用いてシミュレーションされ、手続きの効力はオブジェクトの内部状態の形で蓄積される。手続法の違法性は述語 legal を使った定理証明の形で検出し、実体法の違法性は述語 apply を使った定理証明の形で検出する。また、述語 invalidate によって、法的効果の生じない事実の表現がされる。

2) 時間 区間論理式により、条文を Prolog で直

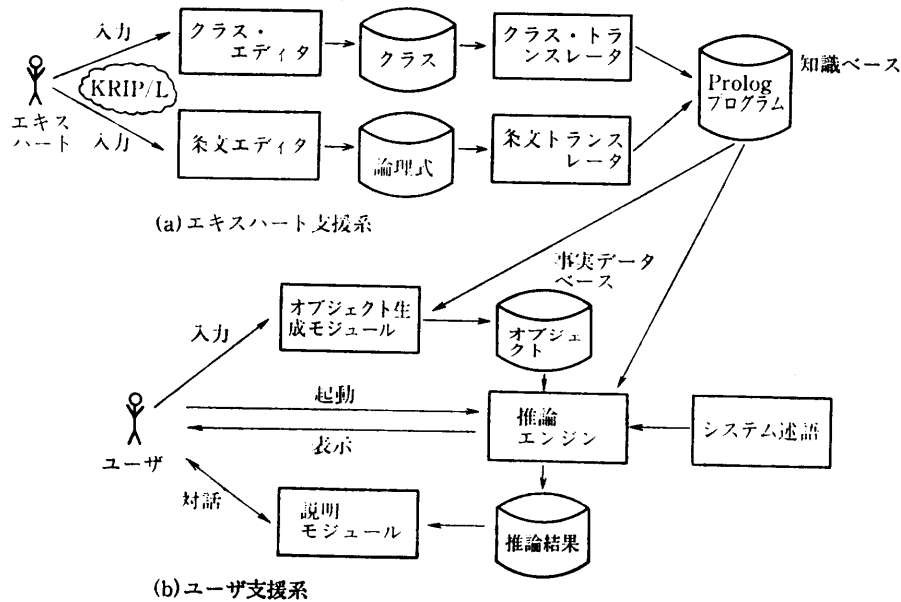


図 2 知識表現システム KRIP の構成
Fig. 2 Organization of KRIP system.

接記述するよりも単純な形で記述できる。オブジェクトの内部状態の履歴が保存されることと、述語 retry によるシミュレーションの再実行機能により、効力がさかのぼって発生する現象の表現が可能である。

3) 法律の解釈 論理式により、条文の厳格な記述ができる(ただし、正確な解釈を行うためには、専門家の助けを必要とする)。また、各クラス中の legal-check の記述により、関連条文と条文間の関係が暗黙のうちに整理されている。

4. 知識表現システム KRIP

4.1 KRIP システムの概要

KRIP システムとは KRIP/L を用いてエキスパートシステムを作るために設計されたソフトウェアツールであり、パーソナルコンピュータ上の Prolog で作成されている。このシステムは、「エキスパート支援系」と「ユーザ支援系」の二つのサブシステムからなっている(図 2)。

「エキスパート支援系」とは、知識ベースを作成するためのソフトウェア群である。専門家が法律の知識を記述する際、KRIP/L の入力には「クラス・エディタ」と「条文エディタ」で行う。これらのエディタは専門用語の辞書を持ち、マウスを用いて入力が高速に行える(図 3)。入力された KRIP/L プログラムは「クラス・トランスレータ」と「条文トランスレータ」

によって Prolog プログラムに変換され、これが知識ベースとして用いられる。

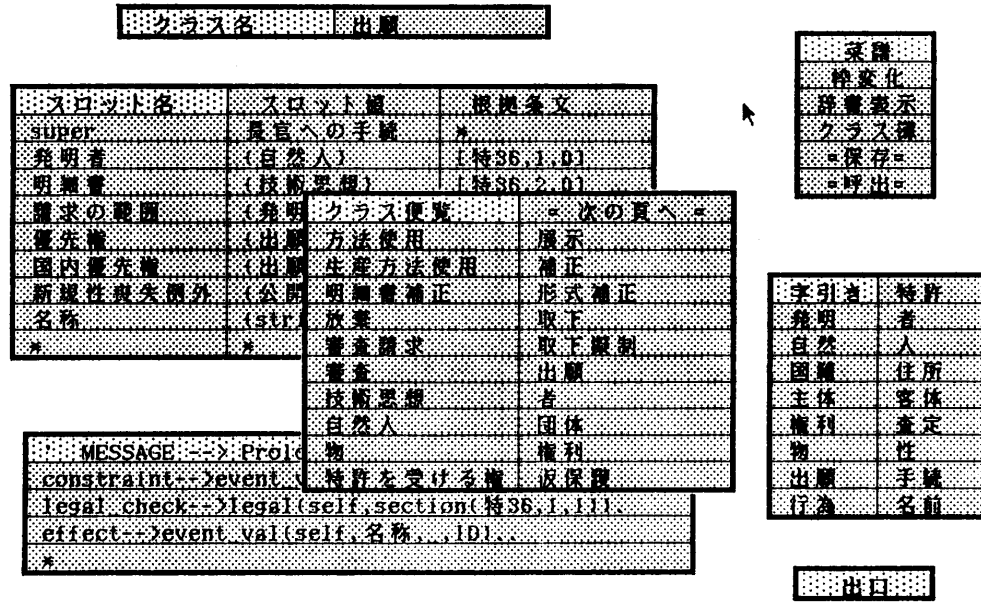
「ユーザ支援系」とは、実際にこのシステムによって事例問題を解決するためのソフトウェア群である。事例を入力する「オブジェクト生成モジュール」、各種の「システム述語」、法的なチェックを行う「推論エンジン」、結果を表示するための「説明モジュール」からなる。システム述語と推論エンジンは Prolog で書かれており、ユーザは機能の追加・変更ができる。

ユーザは、まず、過去に起こった事実を「オブジェクト生成モジュール」に入力するとその結果がオブジェクト(イベント)の集合として「事実データベース」に格納される。事実の入力は(i)日本語(ローマ字)によって入力することも、(ii)エディタを使ってメニュー方式で入力することもできる。日本語による入力においては、文形に若干の柔軟性があり、情報が不足するときにはシステムから補足質問がなされる。

事実入力の後、推論エンジンを起動すると、事実データベースからイベントを一つずつ選択して、メッセージを送って、法的正当性をチェックするのである。

4.2 内部表現

KRIP/L プログラム(クラス記述、区間論理式)はトランスレータによって Prolog プログラムに変換される。区間論理式の変換の仕方は 1 通りではなく、論



21:47

図3 クラス・エディタの画面
 Fig. 3 Screen of 'class editor.'

理式の用い方によって複数の形に変換される。これは、一つの論理式について複数の質問形式が存在するからである。例えば、

[event (ID1, A, ...): event (ID2, B, ...)]
 -->may Q: -C.

は以下のように二つの形に変換される。legal (ID, S) は「イベント (ID) は条文 S によって正当性が証明できる」の意味であり、may (T, Q, S) は「時刻 T に手続き Q をすることができる。その根拠は条文 S である」の意味である。前者はすでに存在しているイベントの正当性の検証に用いられ、後者は現在どんな手続きをすることができるのかをリストアップするのに用いられる。

legal (ID, section (...)) :-
 time (ID, T),
 event (ID1, A, ...), time (ID1, T1),
 ((event (ID2, B, ...), time (ID2, T2),
 between (T, T1, T2)); before (T1, T)),
 call (C, T), event (ID, Q, ...).
 may (T, Q, section (...)) :-
 event (ID1, A, ...), time (ID1, T1),
 ((event (ID2, B, ...), time (ID2, T2),
 between (T, T1, T2)); before (T1, T))
 call (C, T).

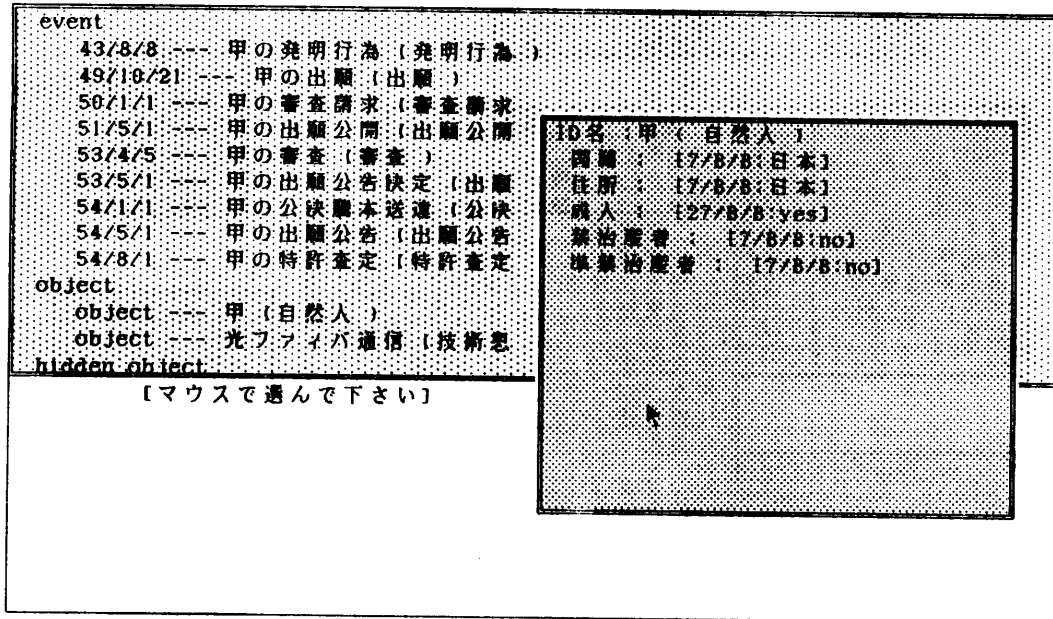
5. KRIPによる特許法エキスパートシステム

KRIP により特許法エキスパートシステムを作成した。このシステムは、特許法の学習者の助けをすることを目的とするものである。ユーザが事例を入力すると、システムは時間の順に一つずつ手続きの正当性をチェックし、違法な手続きを検出する。知識ベースの作成には専門家の助けを借り、文言だけでなく、条文の運用や解釈の知識、民法の知識を反映するようにした^{9),10)}。したがって、条文の KRIP/L 表現はもとの条文とは 1対1には対応していない。

簡単な実行例として図4にあげる事例を実行したとき、「50年1月1日になされた出願審査請求」のチェックの流れを説明する。ユーザが事例を入力したとき、事実データベースにはこの出願審査請求のオブジェクト（その識別子を仮に id1 とする）が作られているはずである（厳密には1月1日は手続きは行えないが、そのチェックは省略する）。

古い手続きからシミュレーションがなされ、id1 の番になったときには、それまでの状況を表すオブジェクト（例えば特許出願の審査状況を表すオブジェクト）がいくつかできている。

出願審査請求の記述および関連条文は次のようになっている。



20:42

図 4 入力された事例の表示
Fig. 4 Display of case problem.

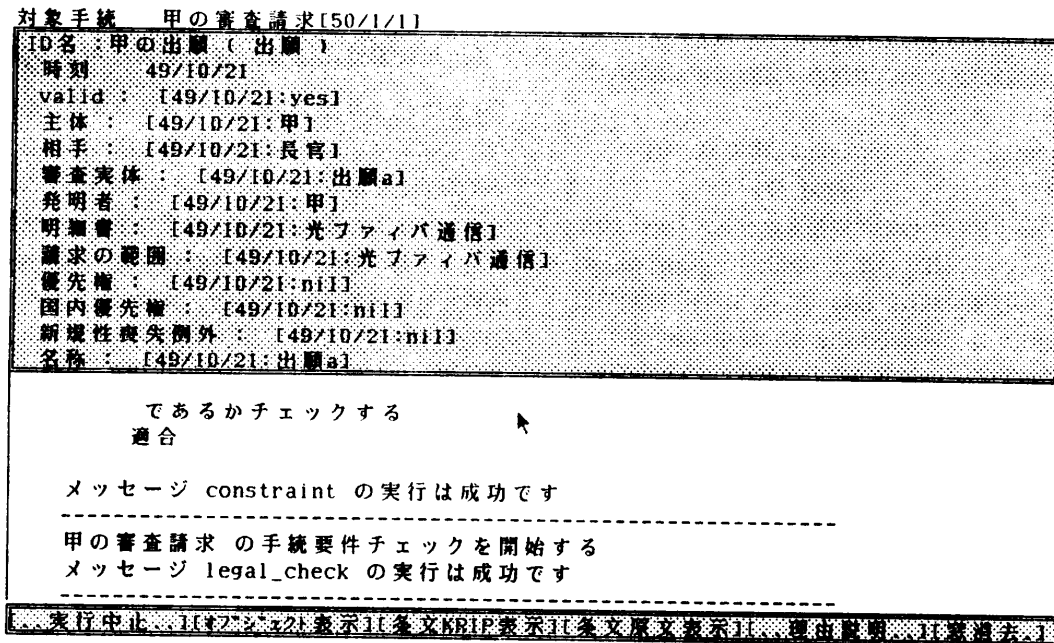


図 5 シミュレーションの結果の表示
(画面の下半分は、シミュレーション結果の表示、画面の上半分は、指定したオブジェクトの内部状態の表示)
Fig. 5 Display of the result of simulation.

```

class 出願審査請求.                                     % 48-3 条をチェックせよ
...                                                       effect→exec (...48-2...), ...
constraint→...% タイプをチェックせよ                    % 48-2 条を実行せよ
legal_check→legal (...48-3...), ...
end_class.
    
```

48-3 条

section (特 48-3, 1, 1).

[event (...出願...): limit (...7 year...)]

→may 出願審査請求 (...):-

object (...審査状況...).

(出願日から7年以内に審査請求ができる)

48-2 条

section (特 48-2, 1, 1).

exec :-modify (...審査状況...)

(「審査の状況」が特許庁係属状態から審査係属状態に変わる)

id1 に対し、推論エンジンはまず、constraint メッセージを送り、「id1の各スロットの値が拘束条件を満たしているか」をチェックし、その結果を出力する。次に、legal_check メッセージを送ると 48-3 条の規定により、「id1が出願日から7年以内に行われているか」、「その時点で適切な審査状況が存在するか」がチェックされ、結果(適法か否か)が出力される。これらのチェックで違法性が検出されなければ、最後に effect メッセージを送ると、48-2 条に規定する id1 の効果として、審査状況オブジェクトの内部状態を変更する。違法性が検出されたときには effect メッセージを送らず、id1 に「無効」の印を付ける。

実行の途中で、ユーザはマウスを使ってメニューから説明コマンドを選択することができ、以下にあげるような細かい説明を画面の上面に表示させることができる(図5)。

- 1) そのような判定をするに至った詳しい説明と根拠条文(条文番号)の列挙。
- 2) その時点で現存する各オブジェクトの内部状態。
- 3) 指定した条文の原文の表示。
- 4) 指定した条文の KRIP/L 表現の表示。

特に2)の機能により、時々刻々変化する複雑な権利関係が表示されるので、過去の出来事の法的なチェックだけでなく、これから何をすべきかという戦略的な目的にも使うことができる。

一つの手続きのチェックに要する時間は、平均で1秒以内である。

6. む す び

工業所有権法表現システム KRIP について述べた。手続法記述言語 KRIP/L は区間論理とオブジェクトの概念を融合したものであり、主に手続法の記述のために開発されたものである。KRIP/L のポイントは

(1)手続法をシミュレーションとして扱ったので、時間とともに変化する状況を扱うのに都合が良い、(2)区間論理式により、期限に関する規定を明確に記述できる、(3)オブジェクトの内部状態の履歴を保存するので、手続きの効力が過去にさかのぼって発生する現象が実現できる、(4)条文に表れる名詞をクラス記述として階層的にまとめることができる、(5)条文参照の順番付けを行って条文の体系化を行うことにより、法令文が大局的に矛盾を生じる問題を解決できる、等である。条文間の矛盾解消のためには(5)の方法のほか、様相論理や暗黙推論による解決も考えることができる。法の厳密な解釈を論ずるときには、これらの拡張論理のほうが適していることもある。しかし、実際に専門家も条文間の知識を用いて法の適用を行っており、ここでの解法は専門家の発想法と似ている。また、条文数が増えたときの実行速度の点からもこの解法のほうが優れている。法律の解釈やメタ知識の導入は知識ベース作成時に専門家の助けを借りざるをえないが、その表現自体は KRIP/L で容易に記述できる。

特許法エキスパートシステムにおいては、(部分的なものを含めて) KRIP/L で記述した条文は、2, 3, 6, 7, 8, 17, 17-2, 17-3, 18, 25, 29, 29-2, 32, 33, 36, 37, 39, 48-2, 48-3, 48-5, 49, 50, 51, 52, 55, 57, 58, 60, 62, 64, 65-2 条である。これらの条文においては KRIP/L の記述力で十分であった。

工業所有権法には第2章に述べたように種々の側面があり、したがって他の多くの法令文記述にも応用できる要素が含まれているものと考えている。また、KRIP システムはオブジェクトと論理式による知識表現を採用しており、この枠組みは法律以外の記述への応用も考えられる。

謝辞 研究の機会を与えられました電総研棟上昭男ソフトウェア部長、佐藤孝紀前情報システム研究室長、藤村是明情報システム研究室長に感謝いたします。また、貴重な御意見をいただいた ICOT 一研・五研の方々、電総研 SIGANY のの方々、吉野一明治学院大学教授を始めとする法的推論研究会の方々に感謝いたします。ご支援いただいた情報システム研究室の方々、人間機械研究室の方々に感謝いたします。

参 考 文 献

- 1) 新田ほか：工業所有権法エキスパートシステムの事例問題解決機能，ロジックプログラミングコンファレンス '85, pp. 375-386 (1985)。
- 2) 新田ほか：手続法記述言語 KRIP/L, ソフトウ

- ェア科学会第2回大会, 3A2 (1985).
- 3) 東条: 法令文検索の機械化の研究, 東大修論 (1983).
 - 4) 吉野: 法適用過程における推論へのコンピュータの応用, 法とコンピュータ, No. 3, pp. 77-94 (1985).
 - 5) 水谷ほか: 記号論理式による法令文事柄検索の一方法, 東芝技術報告書 (1973).
 - 6) 池田: 人工知能言語による法律の解釈と適用, 日経コンピュータ, No. 73, 74 (1984).
 - 7) 米崎ほか: 時間論理プログラミング言語 Templog, ソフトウェア科学会大会, 1E4 (1984).
 - 8) 溝口ほか: オブジェクト指向概念を導入した知識表現言語: 姿と鏡の設計とその応用, ロジックプログラミングコンファレンス '84, pp. 1-12 (1984).
 - 9) 吉藤: 特許法概論, 有斐閣, 東京 (1984).
 - 10) 工業所有権法逐条解説, 発明協会 (1983).
 - 11) Allen, J.F. et al.: Maintaining Knowledge about Temporal Intervals, CACM, Vol. 26, No. 11, pp. 832-843 (1983).
 - 12) Schwartz, R.L. et al.: An Interval Logic for Higher-Level Temporal Reasoning.
 - 13) Cory, H.T. et al.: The British Nationality Act as a Logic Program, Research Memo, Imperial College.

(昭和60年12月3日受付)

(昭和61年8月27日採録)



新田 克己 (正会員)

昭和27年生。昭和50年東京工業大学工学部電子工学科卒業。昭和55年同大学院理工学研究科博士課程修了。同年、電子技術総合研究所に入所、現在に至る。工学博士。論理型言語による知識処理に興味を持つ。著書「プロログ」(昭晃堂, 共著)。日本ソフトウェア科学会会員。



長尾順太郎 (正会員)

昭和23年生。昭和46年京都大学理学部生物物理学科卒業。昭和52年同大学院博士課程修了。昭和52年より(株)日本ビジネスコンサルタント勤務。エキスパートシステムの研究開発に従事。知識表現, 知識獲得, 自然言語処理に興味を持つ。日本ソフトウェア科学会会員。



水鳥 哲也 (正会員)

昭和27年生。昭和52年東京大学教養学部基礎科学科卒業。昭和57年同大学院理学系研究科相関理化学博士課程修了。関数解析学専攻。同年(株)明電舎に入社。電力系統の解析研究に従事。Prologを通して論理型プログラミングの世界に近づく。日本ソフトウェア科学会, 日本数学会各会員。