ICOT Technical Memorandum: TM-0654

TM-0654

知識獲得支援システム
(1987年度 KSS-KAS-SWG報告書)

溝口理一郎(大阪大学), 滝寛和,
椿和弘, 藤井裕一

December, 1988

ICOT
Mita Kokusai Bldg. 21F
4-28 Mita 1-Chome
Minato-ku Tokyo 108 Japan
(03) 456-3191～5
Telex ICOT J32964

Institute for New Generation Computer Technology

ICOT Technical Memorandum

TM－654

知識獲得支援システム

－１９８７年度 知識システムシェルワーキンググループ・
知識獲得サブワーキンググループ報告書－

ＫＳＳワーキンググループ・ＫＡＳ・ＳＷＧ
（溝口理一郎［大阪大学］主査）
（担当、滝寛和「ＩＣＯＴ 第五研究室］）

ＩＣＯＴ

１９８８年 １２月１５日

目次

（序文）

本報告書は、1987 年度KSS-WG KAS-SWGの集中討議、文献調査ならびに、各委員の研究紹介を中心に、知識獲得支援システムに必要な技術について紹介する。

なお、本報告が、ICOTの知識獲得支援システムEPSILON の開発に寄与したことに関して、KAS-SWG の主査ならびに、各委員に感謝いたします。

（ICOT 第5研究室 藤井 裕一、滝 寛和、椿 和弘）

ＫＡＳ報告書

大阪大学　溝口理一郎

１．はじめに

エキスパートシステムの構築は，高機能な汎用のツールを用いて，手探りで行われているのが現状である．その際，タスクの専門家からの知識獲得が知識工学者によって行われるが，それを支援する方法論は確立されておらず，有効な支援システムの開発が待たれている．その意味で知識獲得はエキスパートシステムの構築において極めて重要な課題となっている．

一般に，「専門家は自分が知っていると思っていること以上のことを知っている」と言われるように，専門家にとっては自分の持っている知識を整然と述べることは困難であり，しかも本来体系化されていない，経験的な知識に関してはなおさらである．知識獲得の困難さの本質はここにあると言っても過言ではない．

実社会におけるエキスパートシステムの応用が一段落してある程度のことがわかってきた今日，最も求められているのは，知識獲得の方法論である．専門家が行っている問題解決過程の解析やそこで用いられている専門知識の抽出を有効に支援し，エキスパートシステムの構築を能率よく行うツールの開発はエキスパートシステムの分野に大きく貢献するものと思われる．

近年，知識獲得に関する研究が多く行われているが，殆どが狭い範囲の問題を個別に議論しており，知識獲得支援システムに関する体系的な研究はなされていない．次世代の知識シェルを考えるとき，知識獲得を支援するシステムは不可欠であることは言うまでもない．ＫＡＳサブワーキンググループではこのような認識のもとに知識獲得の問題を幅広く考察し，次世代のシェルにふさわしい高度な知識獲得支援システムのイメージを得ることを目的として検討を行った．

特に，６１年度の知識獲得ＷＧで行われた，動向調査と要素技術の検討結果を踏まえて，具体化した知識獲得支援システムのイメージ作りを目指して，次の３つの観点に重点を置いて検討した．

(1)エキスパートシステム構築の方法論の一環として捉えること．

(2)知識ベースの構築支援と捉え，知識獲得の様々な形態を考察すること．

(3)学習等の他の方法論との関係を考察すること．

これらの方針に基づいて，各委員に次のような担当を定めた．

知識の上流	篠原委員（電力中央研究所）
学習	安部委員（大阪大学）
自然言語インタフェイス	空閑委員（シャープ）
インタビュ	溝口主査（大阪大学）
モデルと汎化タスク	小林委員（日立），田中委員（東芝）
	多田委員（富士通）

さらに，これらの委員の方々に加えて，数名のオブザーバの参加を得て知識獲得に関する最新の研究動向を把握するための調査研究も行った．

本報告書はこれらの委員会活動結果の概要をまとめたものであり，８章より構成されている．２章は知識獲得の諸問題を整理している．３章では知識獲得の上流の問題として位置づけられる知識の整理の問題を，４章では知識獲得への様々なアプローチについてまとめている．５章では問題解決の基本的な構造について述べている．６章では知識獲得に関するいくつかの研究を紹介している．７章では本委員会活動で得られた知識獲得システムのイメージについて述べている．最後に８章において検討の結果をまとめ，今後の課題を述べている．

２．知識獲得における諸問題

知識獲得に関する研究の多くのものは知識獲得のある一つの観点を強調したものであり，知識獲得の全体像が把握し難いのが現状である．そこで本章では，知識獲得支援を考える上で重要と思われるいくつかの視点を整理する．

2.1　知識獲得の形態

知識獲得は広義には知識ベースの構築支援全般を意味するが，そのように考えたとき，その形態は次の６つに分類することができる．

①深い知識からの生成
②例題からの学習
③テキストからの獲得
④インタビュによる獲得
⑤知識ベースの修正
⑥知識ベースの管理

以下に各項目について簡単に述べる．

（１）深い知識からの生成

専門家は基本的な知識を学習した後，様々な経験を通して経験則を修得していく．このような専門家の成長過程に着目して，深い知識と呼ばれるドメインの基本原理や対象物の構造や部品の機能に関する知識から，タスクに直接使われる浅い知識をコンパイルすることができる[2-1]．深い知識は客観的な知識であり，基本的にはドメインに依存するがタスクには依存しないものである．タスク知識は深い推論エンジン（コンパイラ）が持っている．この考えの基本には，有効な経験則（浅い知識）の多くは客観的な知識によってその正当性を説明できるという考えがある．理想的には，深い知識は一つのドメインで共通のものが定義され，その上でのタスクに依存した深い推論エンジンが様々な浅い知識を生成するという形態が望まれる．全ての浅い知識をコンパイルしておくことは不可能であるし，又現実的でもない．ある程度，使用頻度の高い浅い知識だけを事前にコンパイルしておき，残りのものは，実際に困難な問題に出会ったときに動的に生成し，有効に用いられたものを蓄積するという方式が妥当であろう．

（２）例題からの学習

学習は知識ベースの構築の観点から見れば極めて重要な課題である．従来の学習に関する研究はＳＢＬ（Similarity-Based Learning）と呼ばれ，事前の知識は殆ど仮定せず，多数の例題中の類似性に注目して，帰納的に学習を行うものが中心であったが，近

年ＥＢＬ（Explanation-Based Learning）[2-2]と呼ばれる新しい学習方式が注目されている．ＥＢＬではＳＢＬとは反対にドメインに関する豊富な知識（Domain Theory）を事前に用意しており，少数の例題（通常は一つ）から学習を行う方式である．実際，人間の学習過程を考えると，ＥＢＬの方が現実的であるように思われる．しかも，知識獲得の立場から見れば.Domain Theoryは客観的な知識であるため，その獲得は比較的容易であり，それに基づいてＥＢＬによってタスクの解決に有効な知識が得られることは大変望ましいことである．これは弟子が師匠の動作をそばで見ていて，そのKnowhow を学ぶことに類似しており，知識獲得の重要な形態の一つとなる[2-3]．また，分類型の問題で例題が豊富にある場合には，決定木の学習も有効である[2-4]．

（３）テキストからの知識獲得

ドメインによっては専門知識に関する情報がドキュメントの形で整理されている場合がある．特に基本的知識は教科書や解説書に記されている．また，診断の手順書なども文書としてまとめられていることが多い．このような場合には，専門家にインタビュする前に，文書から知識獲得するする方法が有効である．ここでは自然言語処理技術が重要な役割を果たすが，このような方法は知識ベースの第一版の構築には有用である．

（４）インタビュによる獲得

インタビュによる獲得が普通に言われる知識獲得であり，知識工学者が専門家にインタビュすることによって知識を獲得する行為を指す．現状ではこの形態が大部分のエキスパートシステムの構築において採用されており，最も重要な問題となっている[2-5]．一般に，人間は自分の知っていることを客観的には理解していない．特に，経験則と言われるような半分無意識の世界にあるような知識に関しては一層その表明は困難となる．従ってインタビュによる獲得では，専門家が自分の知識を表明し易いような状況を作ることが重要な課題となる．言い忘れているところや矛盾の指摘は勿論のこと，知識を提供する側の心理的負担を軽くする配慮も必要となる．

（５）知識ベースの修正

特定のエキスパートシステムが成熟するにつれて知識の枠組みが固まることを利用して，そのシステムに固有の知識表現言語を開発することによって，記述の概念レベルを高めることができる．汎用の記述言語では記述の際の自由度が大きく，問題領域の専門家が直接知識を記述するのは困難であるが，十分検討された問題専用の言語が設定されればそれが可能となる．このことを利用したシステムの例として，XCON-RIME[2-6]やOPAL[2-7] を挙げることができる．例えば，現在6000以上のルールを持つＸＣＯＮは，毎年半数のルールを修正し，保守を行っている．その作業を軽減するために，RIMEと呼ばれる専用の言語を開発している．

（６）知識ベースの管理

知識ベースの管理は知識獲得が持つイメージとはかなり違いが，知識ベースの構築には不可欠のものである．矛盾や冗長性の管理[2-8]以外に，知識ベースの管理には，Case ＤＢとルールＤＢの２つのＤＢが必要となる[2-9]．Case ＤＢにはタスクの例題（Caseデータ）が蓄えられている．ルールＤＢではルールのドキュメントの管理，バージョンの管理に加えて，ルールの実行履歴が，発火した状況（Caseデータへのポインタ）として管理されている．このような２つのＤＢが提供する情報に基づいて，知識ベース

の洗練を円滑に行うことができる．

このように，知識獲得は学習，インタビュ，生成など様々な技法を結合した形で行われるべきものであり，そのための機構を考えて行かなければならない．

2.2 獲得の戦略

知識獲得の為の戦略は次の２つに大別される．

①トップダウン法

タスクにおける知識の使われ方に関する知識を利用した獲得，即ちタスクに固有のインタビュ戦略を利用する方法．

②ボトムアップ法

タスクの知識を用いず，ドメインにおける基本知識の整理から始める方法で，KJ法などの利用が考えられる．

知識工学者がインタビュして知識獲得を行う場合には，たとえ漠然としていても理解のよりどころとなる基本的知識が必要である．ところが，コンピュータはこのような高度な理解能力は持っていないため，むしろタスクの解決に用いられる知識をトップダウンな方法で直接獲得する方が効率が良い場合が多い．

2.3 対象とする知識

獲得の対象とする知識は次の３つに分類することができる．

①上流:基本概念レベルの整理やドメインモデルの構築

②中流:部分問題の把握やタスク空間の設定

③下流:ルールの構成とインプリメント

知識獲得システムを設計する際には何れのレベルの知識を獲得の対象とするかを明確にしなければならないし，どこに重点を置くか，どの順序で行うかは重要な課題である．

＜参考文献＞

[2-1]山口 他：深い知識に基づく知識コンパイラの設計，人工知能学会誌，Vol.2，No.3，pp.333-340 (1987).

[2-2]Mitchell, T.,M. et al.:"Explanation-based generalization:A unifyingview, Machine Learning, 1, pp.47-80 (1986).

[2-3]渡辺 他：VILLA: VLSI設計知識獲得システム，信学技報，人工知能と知識処理，AI86-1 (1986).

[2-4]竹之内 他: 属性発見機能を持つ決定木の自動生成，電子通信学会知識工学と人工知能研究会資料，51-10 (1987).

[2-5]溝口他：知識獲得支援システム，人工知能学会誌，1988.

[2-6]Soloway, E. et al.:"Assesing the maintenability of XCON-in-RIME: Coping with the problems of a very large large rule-base", Prc. of AAAI '87,

[2-7]Musen,M.A. et al.:Use of a domain model to drive an interactive knowledge-editing tool, J. of Man-Machine Studies, Vol. 26, No.1 , pp.105-121, 1987.

[2-8]北上他：論理プログラミング言語Prologによる知識ベース管理システム，情報処理，Vol.26，No.11，pp.1283-1295 (1985).

[2-9]辻野他：知識ベース構築支援環境を備えた連続音声認識エキスパートシステムシステム: SPREXⅡ，電子情報通信学会論文誌D，Vol.J71-D，No.3，pp.531-542 (1988).

3　知識獲得作業におけるドメインモデルの獲得

知識獲得とは、専門家などの知識源から問題解決に必要な知識の抽出を行うことである。この知識獲得作業は、エキスパートシステムなど知識ベースシステム構築の上で、その核となる重要な作業である。

一般に、知識ベースシステム作成のための知識獲得作業では、最終的に次の4種類の情報とその表現を明らかにする必要がある。

① どの対象物に着目するか？
対象としている問題領域には様々な実体が存在する。これらのうちどれを扱う対象とするかを明確にする必要がある。この情報は、知識ベース上にはフレームとして現れることが多い。

② 対象物間の関係、属性のどれを考慮着目するか？
対象とするものが決っても、例えば機器ならばその重量、大きさ、処理能力や他の機器との接続関係など、対象物毎に様々な属性や関係がある。問題の解決にあたって、どの関係、属性に着目するかを明確にする必要がある。着目する関係についての情報はもれなく持つ、または、推論できる必要がある。この情報は、知識ベース上には、フレームのスロットや述語の種類などとして現れる事が多い。

③ 断片的な判断知識
②で定められている関係、属性に着目して判断を行なっていくための知識である。この情報は、ルールとして知識ベース上に現れる。

④ 問題解決戦略／断片的判断知識の制御知識
③の断片的知識の制御についての知識である。これには、Generate&Testなどのような（大局的）問題解決戦略の決定から、個々のルールの競合解消まで様々なレベルがある。この情報は、知識ベースには、プログラムの制御構造として現れる。メタルールやルールの優先度その他の競合解消として現れる場合が多い。

①②の対象物とそれらの間の関係、属性は、従来のデータベース作成などにおいても共通に必要となる情報である。しかし、従来のデータベース作成と異なり知識ベース作成のためには、これらの情報を使っての問題解決が行ないやすいように、より詳細な分析を行なって構造化をはかっておく必要がある。

③④は、知識ベース作成において固有の情報である。しかし、③個別の判断知識や④制御知識は②対象物間の関係と区別しづらい場合がある。特に、問題領域についての知識が明確になっていない初期の段階では、②対象物間の関係を獲得分析する過程で③個別の判断知識や④制御知識の原型となる情報が明確に区別されずに含まれることが多い。

これらの知識①②③④をどのような順序で如何に獲得し、更に精密化していくかが知識獲得の課題である。

ここでは、（１）①②の基本的な概念や概念間の関係などを整理するフェーズ、（２）④の内、特に（大局的）問題解決戦略を決定するフェーズ、（３）③の個々の判断知識の獲得とそれらの間の優先順位などの詳細な制御知識の獲得を行なうフェーズの３つに分けて考える。即ち、知識獲得作業を、次の３つに分けて考える。

（１）上流工程：基本的な概念や概念間の関係などの整理（ドメインでの活動を表現したドメインモデルの構築）
（２）中流工程：部分問題の把握（問題解決戦略の選択）
（３）下流工程：個々の詳細なルールの獲得とその実現

エキスパートシステム構築においてよく用いられる、知識工学者によるインタビューでは、ほぼ上記の順序で作業が行なわれることが多いので、上流、中流、下流と名付けている。

知識獲得に際しては、（１）（２）（３）のどのフェーズの知識を獲得しようとしているのかに応じてその獲得法は異なってくるので、これを明確にしておくことは重要である。

また、このよう知識をどのような順序で獲得して行くかについては、どのような情報がすでに分かっているかに応じて、大きく３つのアプローチが考えられる。

第１は、（１）問題領域における基本的概念などの整理からはじめて、問題を把握した上で、（２）問題解決戦略の選択や、（３）個々の詳細なルールを獲得していくボトムアップ的アプローチである。

第２は、問題のタイプ（タスクの型）がある程度分かっているとき、（２）その型の問題のための問題解決戦略を同定して、そこでの知識の使われ方に基づいて（１）問題解決に必要な領域モデルの構成や（３）個々のルールの獲得を行なうトップダウン的アプローチで、類型タスク(Generic Task)的なアプローチである。

第３は、領域知識の型もその解決戦略の型も充分に分かっているときに、これら領域固有の知識に基づいて、（１）ドメインモデル、（２）問題解決手順（ルールの適用順序など）、（３）個々の判断規則（ルール）のひな形と専用の獲得インタフェースによって、これらを精密にしていくトップダウン的アプローチで、領域固有的アプローチである。

このようなアプローチの内、どのアプローチをとっているか、どれだけの情報を前提としているかによっても知識獲得法は異なってくる。

本章では、各アプローチにおいて（１）の上流工程／ドメンモデルの獲得について述べる。

３．１　ボトムアップ的アプローチにおけるドメインモデルの獲得

エキスパートシステム開発においては、しばしば知識工学者が専門家にインタビューすることによって知識を獲得することが行なわれる。その際には、このボトムアップ的アプローチが採られることが多い。

ボトムアップ的アプローチにおいては、上流工程は対象領域に登場する基本的な概念や概念間の関係など整理してドメインモデルを構築すると同時に、その問題に適した問題解決戦略を見つけたり、個々のルールの原型を得るという中流工程、下流工程の準備作業と言う意味合いが大きい。

富士通のエキスパートシステム構築技法ＥＳ／ＳＤＥＭ[1]では、概念化、構造化、詳細化、評価という４フェーズに基づいてプロトタイピングを行っている。概念化のフェーズでは、業務内容、システム利用者の把握、解の定義を行い、問題解決方式、戦略、対象モデル、仮説集合などからなる専門家モデル（専門家のドメインモデル）のスケッチを行っている。そして、これに基づいて、構造化のフェーズで、対象モデルの構造を設定したり、問題解決の方式（問題解決の流れの構造）を決定をしたりしている。

ここでいう上流過程には、概念化のフェーズと構造化のフェーズの一部が相当する。概念化のフェーズでは、業務内容を文献調査、専門家によるレクチャやインタビュー、現場観察などに基づいて理解する。また、必要となる答（解）を定義する。そして、この解を導くための過程とそれに関係する諸概念を獲得することで、専門家モデルの概要を理解する。インタビューの際には、「～を～する」などの動詞表現「動詞」や「名詞」に着目することで、処理の流れ（問題解決方式）とその対象となるデータ領域（対象モデル）を明確にするようにしている。また、インタビューの際にコメントしてでてきたような名詞を参考に、「仮説」となる事項を明らかにするようにしている。

ＥＳ／ＳＤＥＭは構築技法で、まだシステム的支援は不十分であるが、問題解決方式の選択に際しては、パラダイムという類型タスク的考え方も取入れるなど、実際的なエキスパートシステム構築システム方法論としては優れたものである。

ボトムアップ的アプローチにおけるドメインモデルの構築作業を行なう時点においては、対象問題の性質については充分な理解がない場合が多く、この場合、（理解／モデル化のための）適切な抽象レベルの選択が一つの問題となる。例えば、「人」という対象概念の下位概念として「既婚者」という概念を設ける事が意味を持つか？　「既婚者」には「配偶者」を、「夫」「妻」という性別を考慮した捉え方すべきか？　などの問題がある。また、様々な対象概念の内、どの対象対象概念のどの関係、属性に注目して問題解決をすべきかなどの対象領域を見る適切な視点の決定も重要な問題である。これらは、③個々の判断規則、④その制御知識とあいまってはじめて正確に決ってくるもので、上流工程ではまだこれらの候補を模索している段階で、視点や抽象度は変化する。このため、適切な視

点や抽象度の模索を助けることができる分析手法が必要となる。

このような観点から、存在論的分析(Ontological Analysis)が注目される。

Alexander[2]らにより提案された存在論的分析は、問題解決のための対象領域について、ドメインタスクの本質的な構造を分析、記述することを目指している。

存在論的分析には３つのレベルがある。第１は、静的存在論(Static ontology)で、これは物理的な対象物や問題空間のプリミティブな対象、更に、オブジェクトの属性、オブジェクト間の関係を定義する。第２は、動的存在論(dynamic ontology)で、これは対象物の行動その他問題空間での状態遷移を引起こすオペレータを定義して、問題解決の最中に変化する動的情報と静的情報を区別する。第３は、認知論的存在論(Epistemic ontology)で、静的分析、動的分析で明らかになった知識が、問題解決でどのように利用されるかを制御する制約、推論方式を定義する。これら分析に際して、Alexanderらは表示的意味論と代数的仕様記述を使用することで、同一物に対する見方の違い（内包と外延）の記述や抽象化の階層の構成などを行なう事ができる。また、表示的意味論による記述によって意味論が記述できるので、意味論的矛盾も捉えることができる。しかし、表示的意味論や代数的仕様による厳密な記述はあまり容易でなく、支援系の充実が望まれる。

また、篠原ら[3]はＫＪ法に関係の階層という考え方を取入れて、適切な抽象度や視点の模索を支援する知識整理支援システム CONSIST を開発している。

ＫＪ法[4]は、川喜多二郎が提案したボトムアップア的整理手法[1,2,3]で、事前には相互の関連性が明らかでない個々の情報（個々の具体的問題、様々の見解など）を整理して、それら間の相互の関連性を見出して行く手法である。ＫＪ法では、まず、ブレインストーミング、ヒアリング、キーワード抽出などにより対象に関係のありそうな情報をできるだけ広く集め、個々の情報ごとにその内容を一行見出しをつけてカード（小項目）に記す。それを机上に広げて全体を眺め、(I)親近感を感じるカードをまとめて小グループをつくる。このようにしてできた小グループ（中項目）に新たに名称を与える。さらに、このグルーピング作業を繰り返して、項目を小項目、中項目、大項目、．．と整理していく。このようにしてサブ問題を明確化した後、ＫＪ法ではこれらの項目を箱で表し、(II)これらの間に関係があれば矢印で結んで見やすい図（構造グラフ）にする。

CONSIST では、このＫＪ法を拡張した整理法をグラフィッカルなインタフェースによって支援する。CONSISTにおける整理法では、グループ化や関係付けに際してその際に着目している関係の種類を指定すると同時に、これらの関係の種類の間に階層関係を認めることで「妻」「夫」と「配偶者」などの関係の抽象度の違いも表現できるようにしている。また、初期の段階では、③個々の判断規則や④それらの制御規則などの原型がはっきりと意識されず未分化のまま概念間の関係として捉えられることが多い。これも「因果的関係」という関係の種類のインスタンスとして捉えることでこれらも統一的に扱っている。

着目する関係が明確になってから、対象概念間の関係を洩れなく整理する上

では、ＩＳＭ法(Intepretive Sturcutural Modeling)、ＤＥＭＡＴＥＬ法(DEcision MAking Trail & Evaluation Laboratory)などの従来からの構造モデル化技法[4]も有用である。

ＩＳＭ法は、サブ問題（項目）の総合的順序付けを行うことを目的とする。ＤＥＭＡＴＥＬ法は、ＩＳＭのように総合的順序付けを狙うのではなく、サブ問題間の（影響）関係を捉らえる構造モデルを作成することを目的とする。これらの手法は、関心をもつ関係を１つ定め、それについての質問（例えば、「サブ問題ｉが解決するとサブ問題ｊの解決を促進するか？」）に対する回答をサブ問題のペアごとに得て、これをグラフ（ネットワーク図）として表わす。このグラフを見ながら総合順位や影響関係についての吟味をする。故障原因を同定するために利用されるＦＴＡ（Fault Tree Analysis）や、故障の波及を調べるためのＥＴＡ（Event Tree Analysis）などでも、構造モデルが故障確率などと結びつけられて利用される。

以上の分析法では、その分析の元となるデータは既にプロトコル解析、教科書、マニュアルその他によってすでに揃っていることを前提にしている。これに対して、システムが主体的に質問を発することで、対象概念や関係を系統的に獲得しドメインモデルを獲得していくことも重要である。

このような場合、インタビューで使用される「動詞」と「名詞」がドメインモデルについての重要なキーを与えていることが多い。

ＫＮＡＣＫ[5]は、デザイン分野での用語の構造についての知識（例えば、「デザイン部品」は「性質」を"持つ"などの、動詞の用法とその時の主語、述語のタイプ情報）を元に、インタビュー戦略に基づいて質問を発する。例えば、「異なるデザイン部品をどのように参照するか？」などの質問をすることで、デザイン部品の名前など新しい概念が導入される。Ｉ³Ｓ[6]は、さらに、インタビューによってドメインで使用される動詞についての情報を得たり、名詞の概念階層を精密化する質問戦略を用いることで、ＫＮＡＣＫでは初めから持っているドメイン固有の情報を学習しながら、より柔軟にドメインモデルを拡張するようにしている。また、類似の他分野で獲得されたドメインモデルを利用した、質問をすることで、より効率の良いドメインモデルの獲得を支援している。

これらのボトムアップ的アプローチは、分析者（知識工学者）が問題を理解して、問題解決戦略を決定したりする上では有効であるが、結果的に問題解決に直接必要のない情報を獲得していたり、逆に、後に問題解決に必要となる情報が充分に獲得されていないおそれもある。あらかじめ、問題のタイプが分かっている場合は、知識獲得の効率と言う点から言えば、次に述べる類型タスク的アプローチの方がよい。

３．２　類型タスク的アプローチにおけるドメインモデルの獲得

類型タスク的アプローチでは、問題解決戦略は既に決定しており、その実現の為に必要となる要素をタスク固有のインタビュー戦略によって獲得していく。

ＥＴＳ[7]は、分類型のタスクについての知識獲得システムである。ＥＴＳでは、KellerによるＰＣＴ(Personal Contruct Theory)[8]に基づいて、分類問題のための知識を獲得している。分類型の問題は、結局、観察される属性(construct)と選択項目（分類、仮説など）の対応関係を明確にすれば、それに基づいたルールによる判断するという問題解決戦略をとればよい。このような問題解決戦略に基づいて、ＥＴＳの上流工程では、選択項目のリストが既知になったとき、これらの選択項目を提示して質問を発することで、分類に必要な属性を獲得していく。即ち、任意に選ばれた３つの選択項目（例えば、虎、犬、人）を提示して、これらの２つ（虎、犬）に共通で残りの１つ（人）に欠けている属性（２本足／４本足）が尋ねられる。このような質問に対して専門家は属性を述べることで、属性が付加される。更に、このような各選択項目についての各属性の評価が尋ねられることで、ドメインモデルが獲得されていく。この際に、同時に③個々の判断ルールも獲得されていることは重要である。ＥＴＳではこのようにして獲得された結果を元に、属性の従属関係を分析してその結果から、判断ルールを生成することができる。

ＥＴＳは、タスクの型に基づいたインタビュー戦略に基づいてドメインモデルを拡張していく典型的な例である。分類型問題を以外についても、タスクの型に基づいてドメインモデルを拡張して行くことは有効であると考えられる。しかしながら、現在のところ分類型問題を除いてはタスクの型に基づくドメインモデルを拡張して行く有効なインタビューはあまり見つかっていない。今後は、類型タスクの明確化と、それに基づくインタビュー戦略の明確化は重要な課題である。

類型タスク的アプローチでのドメインモデルの獲得では、ボトムアップ的アプローチでのドメインモデルの知識獲得とは異なり、問題解決方式が決っておりそれに必要な項目のみを獲得していくため、無駄になる知識の獲得がなく効率的な知識獲得が行なえる。しかしながら、対象問題の型がはっきりしていることが前提となっているため、対象問題がどの型の問題かが比較的容易に判定できることが重要である。

特に、対象問題が単一の類型タスクによらない複合型の場合などについても、効率よく判別するためのインタビューなどのタスクの問題構造の把握の戦略の研究が重要である。

プリポスト法[9]などによる作業手順の構造に基づく分析に基づいて、タスクを類型化することなども考えられる。

３．３　領域固有なアプローチにおけるドメインモデルの獲得

類型タスク的アプローチでは、タスクの型／問題解決方式が共通なものについては同じ方式で知識を獲得しようと言うのに対して、領域固有なアプローチでは、その知識獲得において扱う対象問題自身の構造を利用して、対象問題に特化したドメインモデルのひな形とその拡張の為の専用の獲得インタフェースを準備するものである。このひな形や獲得インタフェースは、領域の専門家が直接入力

可能なように、完全に問題領域の用語や問題領域でよく用いられる表示方法（例えば、表形式）で設計されている。

ＯＰＡＬ[10]は、ＯＮＣＯＣＩＮというガン治療計画を選定するエキスパートシステムの知識ベース作成／訂正用の知識獲得ツールである。

ＯＰＡＬでは、治療法や薬などは対象領域での実体で、これらの実体についての情報を獲得するために、ＯＰＡＬでは専用の書式（表）が用意されておき、専門家はその表の穴埋めをすることで、個々の薬やその属性値を得ることができる。また、テスト結果に基づく次の診断の決定などの判断ルールにあたる知識も、テスト結果についての表に値の条件とその時に行なう行動のスロットを埋めることによって行なわれる。また、手続的知識はフローチャート風のグラフィカルインタフェースによって設計できるようになっている。

ＯＰＡＬでは、このような専門家向きのモデル化、特化したインタフェースの設計によって、１つの治療計画についての知識の入力に２年近くかかったのが、２～３日でできるようになりかなり効率が向上した。

このように、その対象領域で整理する必要のある同じ様な知識の件数が非常に多い場合は、領域固有なアプローチは有効である。

このような領域固有のドメンインモデルの枠組みや専用のインタフェースの設計は、１つの治療計画を２年近くかかって分析して知識ベースを構築し、それを分析した結果得られたものである。このように、領域に特化したアプローチは、モデル化やインタフェース設計のためコストまでを含めて考えた場合、類型タスク的アプローチと較べ、必ずしも効率がよいとは言えない。特に、コストをかけて作った領域固有の専用インタフェースなどが他の分野では役立たないおそれが強い等の汎用性の問題がある。

類型タスクの分析の進歩などによってドメインモデルの構築が行いやすくなり、それ元にして、対象領域の専門家がより容易に知識の投入ができるようにする専用インタフェースの設計を行えるようになれば、開発コストの低減が期待される。

3.4　効率的ドメインモデルの獲得における問題点

現在のエキスパートシステム開発においては、類型タスクとして研究の進んでいる分類型タスクなどを除いては、業務分析や対象物間の関係の整理などのボトムアップ的アプローチによって、ドメインモデルを獲得することが一般的である。このようなボトムアップ的アプローチの場合には、そのドメインの登場物や登場物間の関係、さらには、それに関わるタスクの種類をインタビューなどで使用される「動詞」や「名詞」に着目して獲得して、ドメインモデルの全体像を明らかにして行くことが重要である。また、インタビュー等で取り出された登場物やそれらの間の関係などを構造化する際には、その適切な抽象化のレベルの選択も重要である。

一方、領域固有のトップダウンアプローチでは、一般に効率的な知識獲得を

行うためのドメイモデルの形成およびその獲得法の設計を行うために、対象領域の問題を充分分析してその対象領域用にカスタマイズを図らなければならない。このためのコストは、現状では多くの場合非常に大きな物となる。

知識獲得の効率化の観点から見ると、このようなボムアップ的アプローチや領域固有のアプローチよりも、類型タスク的なトップダウンアプローチの方が望ましい。類型タスクによるアプローチでは、問題解決の型に基づいてその実現に必要な情報のみを獲得できるように適切な質問を発することにより効率的なドメインモデルの確立が行える。しかし、このような効率的ドメインモデルの獲得が行えるためには、対象問題がどの類型タスクに属しているか判断が正しく行われる必要がある。

今後は、対象問題における類型タスクの分析までも含めた知識獲得においてどのような観点にたったタスクの類型化が良いかの研究が一層重要となる。

［参考文献］

[1] Matumoto et al.:ES/SDEM-Software development and engineering methodology for expert systems",Proc. of AI'87 Japan, pp.527-531,1987.

[2] Alexander,J.H. et al.:"Ontological analysis: an ongoing experiment," Int. J. Man-Machine Studies,Vol.26,No.4,1987

[3] 篠原靖志,寺野隆雄：「関係の階層を利用した知識ベース構築支援システム」,信学技報,AI86-32,1986.

[4] 寺野寿朗：システム工学入門,共立出版,1985.

[6] 川口他：「知的インタビューシステムＩ２Ｓにおける学習機能について」,知識工学と人工知能56-13,1988.

[7] Boose,J.H.: "Personal construct theory and the transfer of human expertise," Proc. of AAAI'84, pp.27-33,1984.

[8] Keller,G.A.:The Psychology of Personal Constructs, New York:Norton, 1958.

[9] 滝　寛和：「解釈型知識獲得システム－知識解釈の問題を中心に－」,知識工学と人工知能56-13,1988.

[10] Musen, M. A. et al.:"Use of a domain model to drive an interactive knowledge-editing tool," Int. J. Man-Machine Studies,Vol.26,No.1, 1987

4.1 自然言語（テキスト解析）

コンピュータによる処理という制限を取り払い、我々が知識をどのような形で獲得しているかを考えた場合、言葉（自然言語）の存在は大きい意味を持っている。このことは、幼児の学習過程や障害者などの教育においてしばしば述べられている。知識の多くは、自然言語で記述できるので、自然言語から直接あるいは間接的に、知識をコンピュータに獲得することができれば、知識処理システムの普及につながっていく。

本節では、自然言語で記述されたテキストの中から、知識処理システム用の知識を獲得して行くためのアプローチの方法と、そのような方法を取り込んでいる幾つかのシステムについて述べる。

4.1.1 テキスト作成の方法

専門書、論文などは、特定分野の新しい知識、体系付けられた知識が取り扱われている。各種機器のマニュアルでは機器の操作方法や注意などが記述されている。オフィスでは、整理した知識や断片的な知識が報告書やノウハウ集として記述されている。知識の種類やレベルはこのように対象分野、目的、記述の方法などの違いで様々に書き分けられる。ここでは、専門書、マニュアル、ノウハウ集などという書かれた媒体の種類やレベルを問わず、まとまった知識を記述したものをテキストと呼ぶことにする。

最近、作文の仕方、文章の書き方の類いの書籍が多数出版されている。これらの中で樺島（樺島　1980）は次の５つのステップを踏めば、うまい文章が書けると報告している。①主題を発見する　②内容に価値付けをする　③表現順序を決める　④表現方法を決める　⑤推敲、校正をする。

テキストには、著者の訴求点、知識などの内容にかかわる情報と、用語、構造などの表現方法にかかわる情報がある。上の考え方は、これらの情報を分離し、各作業を進め、後で、構造化し、テキストを作成していこうとするものである。

実際、このような考えに基づいた、文書処理の装置も開発され出している。例えば①、②を実現するためにはアイデアとネタを記述したアイデア着想ノートが有効であることが指摘されているが、ハイパーテキストは、まさに、それを実現するための装置である。また、③にはアウトラインプロセッサ、④、⑤には文書推敲、校正支援システムが日本でも開発され使われ始めている（本文章もそのような装置を用いて作成したものである）。

思考や、情報収集までを含めた文書作成の方法が工学の対象になっており、今後、テキスト作成時に作った各種の情報が利用できることからテキスト解析の手法も変化していくことが考えられる。

4.1.2 テキスト解析の方法

テキストからエキスパートシステムの知識を獲得するためには、次のような過程が必要である。

①知識文章の抽出

エキスパートシステムの知識になる文章をテキストの中から抽出する。

②知識ベース化

抽出された知識を、エキスパートシステムの知識表現の枠組みに合うように変形する。

従って、作者の頭の中にある知識をテキストで表現し、テキスト解析によって知識を獲得する過程は次の図のように表すことができる。

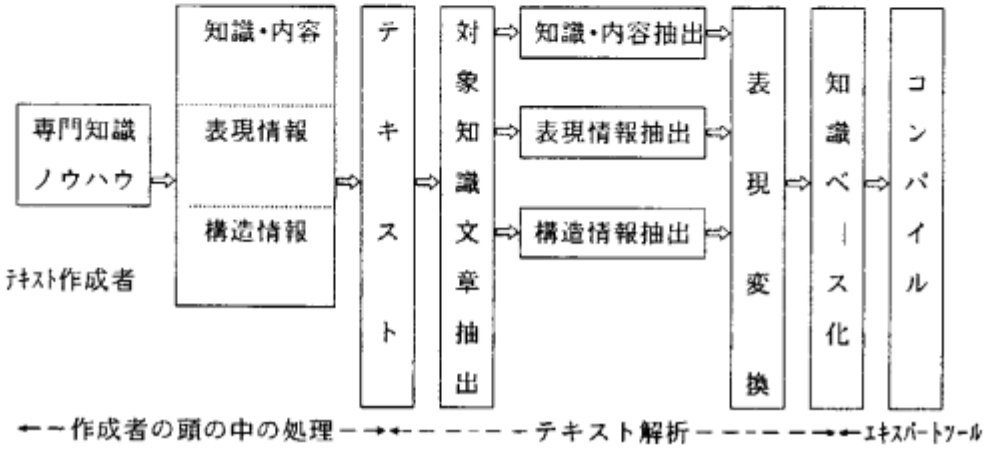


図４．１＿１　テキスト解析のプロセス

狭義には、テキストから各種の情報を抽出する過程をテキスト解析と呼ぶことができるが、ここでは、図のように、テキストから得られる知識を対象とするエキスパートシステムの知識表現の構造に変換し、知識ベースとして蓄える過程までをテキスト解析の範疇に含める。次に、上の図にあるような情報を抽出する方法について考える。

対象知識文章抽出は、テキストの中から、エキスパートシステムに必要な対象知識を記述した部分のみを抽出する部分である。これには、動詞や名詞を中心としたキーワードを用いる方法、文章の構文的なパターンを用いる方法などがある。エキスパートシステムの知識獲得のためには、基本的なドメイン知識の利用も有効であろう。

世論調査の内容分析に、ＫＪカードを用いた対象知識文章の抽出が行われている。ＫＪカードの場合は、一つのカードに一つの内容を記述するのが、原則である。抽出された文章には、二つ以上の事柄を一つの文章で記述する場合も多いので、そのような場合には、２枚以上のカードに分け、カード間の関係を記述するような工夫が行われている。また、文章抽出には、内容分析の要素の上位に当たるような要素を設定したほうが整理が付けやすいと言われている。

次の情報抽出の過程は、エキスパートシステムの知識ベースに加工するまでの内部情報を得る所である。一番上に位置する知識・内容の抽出を目的をした部分の解析方法として内容分析というものがある。これは、世論調査の解析などでよく用いられる。例えば、L.Guttmanによれば、人間の行動は、attitude, intensity, closure, involvement, intelligenceの5つの要素に分類される。ここで、attitudeはある対象に対して、好意的か非好意的かという態度の方向である。intensityはその態度が強いか弱いかという強弱のレベルである。closureはある対象に対する態度が既に決まっているか否かを表す尺度である。記述された内容から、上の各要素のレンジを抽出することにより、人間が対象にどのような行動を取るかを調べることができる（真鍋　1985）。ここで、人間の行動以外の対象に対し、上のような要素と要素の値が決定でき、それらをテキストから抽出できれば、知識・内容の抽出ができる。このような考えに基づいた、システムを後で紹介する。

テキストからの各種情報の抽出には、少なくとも自然言語処理の過程が関与するが、純粋に自然言語処理の延長線上で文章の内容を把握しようという考えもある。これは、文章の理解といわれ、通常、　形態素解析⇒構文解析⇒意味解析⇒文脈解析
過程で情報が抽出され、理解が実現される。理想的には、言語の種類に寄らない概念レベルの内部構造を抽出する。しかしながら、現在の、言語処理のレベルは、構文解析と意味解析の中間付近にあるといわれ、文章理解のレベルは十分ではない。これらは、自然言語処理の研究の進展を待たねばならない（最近の自然言語処理の概要については、人工知能学会誌　88）。内容分析においても、自然言語の処理過程を必要とするので、内容分析手法における要素の抽出に視点を置いた自然言語処理が必要であると思われる。そのような要素には、二つの動作の間の関係、時間的な順序、動作や状態の継続性、程度などが上げられる。

図4.1_1のテキスト情報の中の、中段に書かれた表現情報は、自然言語の記述の恣意性の代償として発生するものである。これには、同一内容の対象に対する、漢語、和語、カタカナなどによる用語の差異、語順の差異、構文の差異などが含まれる。表現上の差異を抽出し、表現の統一を行う目的で利用できるテキスト処理システムの一つに校正システムがある。これは、本来、誤字などを見付け、修正する装置であるが、その機能を積極的に利用すると、表現の統一が可能になる。現在、自然言語処理のフロントエンドプロセッサとして、このような機能を持った日本語用の校正システムが市販されている。自然言語と知識処理との接点に立ち、これらの差異の統一を図ろうとする試みもある。それによると、条件節を真理関数的解釈し、無理のある部分に対して語用論で解釈する（坂原　85）。そして、暗黙の前提知識と言語化の対象領域における目的志向性によって条件節が、条件文、理由文、譲歩文など様々に変化していくことを明らかにしている。

次に、構造情報は自然言語の記述、発話の手続きにかかわる情報のことである。インタビューを用いた知識獲得の手法の中にプロトコル分析に基づくものがある。その分析手法の主なものに、発話プロトコル、行動プロトコル、目の動きのプロトコルがある。テキスト解析の場合は、この中の発話プロトコルに相当するものである。文章においては、全体の構成を形成する部分と、各構成要素の順序、要素内での順序、記述対象の順序などが問題になる。この情報は、現在、自然言語からの知識獲得で利用されていないが、例えば、テキストを利用した上流問題の整理のときには、見出し、アンダーライン、表などの情報を利用した、テキスト内容の概念抽出、タイトリングなどに利用できる。従って、上流問題への展開とともに重要さを増してくるものと思われる。

エキスパートシステムの知識の枠組みが確立しており、上の方法で得られた知識がこの枠組みの中で処理できるものであれば、そのままあるいは、若干の変更を加え、新しい知識として取り込むことが可能である。また、内部の知識の体系に合わない場合は、新たな知識表現の体系を作成する必要がある。

4.1.3テキスト解析を利用した知識獲得システム

テキストから知識を獲得しようと試みている2つのシステムを紹介する。一つは、内容分析の手法を利用し、新幹線の計画に関する記述文章から、人間行動、社会現象を把握し、正しい意思決定を行おうとするシステムである。本システム（楢木　87）では、因果連鎖構造におけるすべてのノードを“決定”D、“知覚”P，“価値あるいは信条”Vの3種類に限定し、これらの間の有向枝として、2つの因果関係“生起”C，“集結”T，2つの心情的関係、“好ましい”＋S，“好ましくない”－Sを定義している。例えば、新幹線建設計画に対し、騒音公害を懸念する文章の部分は次のように表されている。

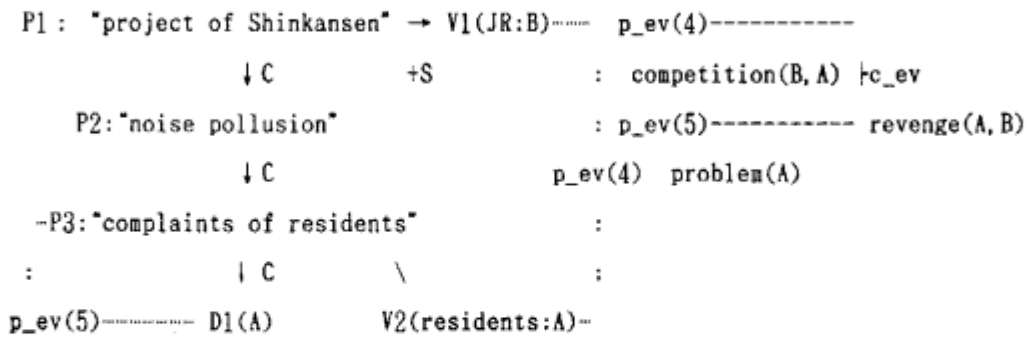


図4.1_2 因果連鎖の多重構造表示

上図では、左の方の図で表した単位事象系列を右の方に表したように複合事象系列へ集約し、複合事象系列を支配する目標干渉関係を抽出することで、内容の把握を行う。

もう一つのシステムは、知識の基になる文章知識プリミティブを用いた知識獲得の

方法（空閑　87）である。対象は、秘書の知識の獲得であり、例えば、「出社直後の１時間と昼食時の１時間と帰宅前の１時間は外来客を避ける。」、「休日明けと休日の前は面会を避ける。」などが、テキストから抽出された文章知識プリミティブの例である。知識獲得までの概略の手順は次のとおりである。まず、抽出のための基準に従い、秘書の業務やノウハウなどを記述したところから、文章知識プリミティブを取り出す。それを、ＫＪカードに書き出し、表現の統一を行う。この中には、１件１内容への統一、用語、構文などの統一の処理が含まれる。次に、ＫＪカードの内容を似ているもの同士でグルーピングし、大中小の代表概念をつけて整理する。これらの文章に対し、ドメイン知識、辞書を利用して、動詞を基準にモードに分類する（上の文章例は、制約モードに属する）。最終的に、プリミティブ文章は自然言語処理の分野で用いられているＣＩＬの形式に変換される。質問文やスケジュール文も自然言語で記述できるようになっており、これらの結果から、自然言語による質問応答を行うことができる。スケジュールの調整は、プランナーとスケジューラと呼ばれるモジュールで実現され、制約条件を満足するスケジュールが結果として得られる。この方法は、実際の秘書のスケジュールを行った結果、エキスパートシステムの初期知識を獲得するのに有効なことが確認されている。現在、制約条件のタイプの分類、対話的な知識リファイン、ＫＪエディタによる知識の整理機構の導入が試みられている。

《参考文献》

（樺島　80）樺島 忠夫 "文章構成法" 講談社 1980

（真鍋　85）真鍋 一史 "世論の研究－内容分析と質問紙による接近－" 慶應通信 1985

（坂原　85）坂原 茂 "日常言語の推論" 東京大学出版会 1985

（人工知能学会誌　88）"次世代自然言語処理技術" 人工知能学会 VOL.3,NO.5 1988

（椹木 87）椹木他 "社会現象の深層構造把握のための因果連鎖知識の多重構造化" 計測自動制御学会 VOL.23,NO9 1987

（空閑 87）空閑他 "文章知識ﾌﾟﾘﾐﾃｨﾌﾞを用いた知識獲得の方法-条件節の統一化" 情報処理学会第35回全国大会5P-10 1987

4.2 インタビュ技術

インタビュの本質は，専門家に自分自身が持つ知識に気付かせることにある．そのためにはいくつかの注意すべき点が存在するが，ここでは個々の技術ではなく，インタビュを考える上で重要と思われる観点を以下に述べる．

①タスク分析と知識獲得のインタビュの区別：専門知識にはタスク固有の推論の枠組みと推論エンジンが用いるデータとしての知識との２種類の知識がある．インタビュに際してはこの２つの知識の区別を明確に意識しなければならない．
②タスク知識の利用：タスクにおける知識の利用のされ方に関する知識を用いれば，質問するタイミングや不足する知識の示唆を行うことができる．この意味でタスクに依存するトップダウンなインタビュ戦略の研究は本質的である．
③専門家に具体的な状況を提供すること：人間は自分が知っていることを順序立てて整然と表現することはできない．従って，専門家に知識を一般的な形で述べさせるのではなく，具体的な問題の解決を通してそこで使った知識に注目してインタビュを行なわなければならない．
④プロトタイプの早期作成とその利用：専門家の思考を活性化するためには，プロトタイプをできる限り早く稼働させて，ケースデータを用いたタスクの試行を通して専門家に刺激を与えることが有効である．

インタビュには大きく分けて初期知識ベースの構築とその洗練との２つの段階に対応するものがあり，従ってインタビュ技術もそれぞれに対応するものがある．特に，後者で用いられる知識の利用のされ方に関する知識に基づくインタビュ戦略は，不足する知識の獲得を可能にする．この意味でインタビュシステムは問題解決システムでなければならない．即ち，インタビュシステムは獲得した知識で問題を解いてみる事により知識の不足を検出し，必要な知識をその獲得に適切なコンテキストの中で獲得することができる．

以下に具体的なシステムにおけるインタビュ戦略を見てよう．

4.2.1 MORE/MOLE

MORE[4-2-1] はいくつかのインタビュ戦略を用いて専門家にインタビュすることにより，知識の獲得を支援するシステムである．MOREにおける知識獲得は次の３段階の処理に従って行われる．

STEP1. 簡単なインタビュによる初期モデルの構築．
STEP2. いくつかの質問戦略に基づいたインタビュによる診断知識の洗練．
STEP3. 診断ルールの生成．

初期インタビュでは，利用者が新しい仮説や兆候を入力するたびに，さらに有用な関連情報の入力を促す．また，兆候の確かさや顕著な特徴などを質問する．次に，このようにして得られた問題領域のモデルから診断ルールを生成し，その弱点を探索して，それを解消するために必要な情報の入力を要求する．そして最後に，完成したルールを生成する．

MOREの大きな特徴は診断知識の抽出に特化されたインタビュ戦略であるが，その基本

的な特徴には次の５つがある．
(1)問題領域のモデルを構築する．
(2)知識ベースから診断ルールを生成する．
(3)戦略知識に基づき専門家に助言を与えることができる．
(4)ルール・ベースをテストする診断用シェルを持つ．
(5)様々な兆候の役割や背景となる条件を説明できる．

以下では，MOREの本質的な思想を担っている(1)と(3)について，その詳細について述べる．問題領域のモデル表現はMOREで重要な働きをしており，以下に示す５つの基本要素から構成される．

(1) H（仮説）：診断の結果となる事象
(2) S（兆候）：仮説から生じる事象および状態
(3) C（条件）：仮説の兆候とは言えないが，診断に深く関わり合う事象および状態であり，以下の５種類のものがある．
ＦＣ(Frequency Condition)：兆候ではないが，Ｈに直接関連する事象および状態（他のＳに依存せずに，常に成立する事象および状態．関連するＨに直接リンクされる）
Ｔ(Test)：Ｓを観察するための手続き（関連するＳに直接リンクされる）
ＴＣ(Test Condition)：Ｔの環境（ＴとＳのリンクに付加される）
ＳＡ(Symptom Attribute)：Ｈに対するＳの詳細な記述（ＨとＳのパスに付加される）．
ＳＣ(Symptom Condition)：Ｓの発生条件（ＨとＳのパスに付加される）
(4)リンク（実線）：モデル中の実体（Ｈ，Ｓ，Ｃ，リンク，パス）を関係づける．リンクで結ばれるＨとＳ，ＳとＳは隣接する（直接関連する）もののみに限られる．
(5)パス（破線）：ＨとＳを関係づける特別なリンク．Ｈからリンクを辿って到達できる全てのＳにそのＨから直接パスが張られる．パスはＨとＳの因果関係のみを表現する．

図１に条件の表現の基本的な形を示す．また，図２に問題領域のモデルの例を示す．モデルの構成要素が示すように，MOREにおける知識表現は診断に特化されたものであるが，診断型の問題でさえあれば，診断の対象には依存しない一般的な手法で知識が表現されていることが分かる．簡単なインタビュにより，まず問題領域を図１に示したプリミティブを用いて表現する．その後，８つの戦略知識に従ってモデルを洗練する．以下に代表的な戦略を４つ示す．

戦略１：仮説の区別(differenciation)：必要な兆候を獲得するための戦略である．まったく同じ兆候を持つ仮説の対に対して，それらを区別するための兆候を専門家に質問する．

戦略２：パスの分割(path division)：同じく兆候を獲得するための戦略である．ある仮説からある兆候へのパスの因果関係が弱い場合に，その間に因果関係がもっと強い別の兆候が存在しないかを尋ねる．

戦略３：兆候の区別(symptom distinction)：ＳＡを獲得するための戦略である．ある兆候(Ｓ)が複数の仮説(H1-Hn)と関連している時，各仮説が識別されるようなＳの詳細な属性を専門家に尋ねる．

戦略４：テストに関連した環境(test conditionalization)：テストの使用環境を獲得するための戦略であり，テストの信頼性に影響する条件を専門家に尋ねる．

これら４つの戦略は，仮説と兆候とそれらの間の形式的な因果関係だけに基づいており，仮説や兆候が何であるかには依存していない．このことから，これらは診断というタスクの型にのみ依存し，タスクの問題領域から独立した，汎用性の高い戦略知識であることが分る．MOREにおけるモデル表現はルールよりは深い知識と言うことができるが，問題領域そのものを理解するという観点から見れば，限りなくルールに近い「深い知識」である．しかしながら，診断にタスクを限定すればこのようなレベルの表現でも知識の獲得や洗練を行うことができることを示したことは意義深いと考えられる．

MOREはさらに改良がなされてMOLE[4-2-2] と呼ばれるシステムに発展している．MOREでは仮説や兆候に対応する事象，及びリンクの型，ＣＦ値等を入力することを直接専門家に要求していたため，専門家自身が自分の持つ知識を曖昧性のないものにし，一貫性のとれたＣＦ値を与えなければならず負担が少なくなかったが，MOLEではこれらの情報に関して曖昧な入力を許すように拡張されている．また，MOREで行われていた知識ベースの洗練は静的なものであったため，戦略としては（適切な質問の生成としては）成功したが専門家がそれに答えられないという問題が生じた．そこでMOLEでは，静的な解析に加えて実際の問題を専門家と共に解決を試み，その中で対話をするという動的な洗練を行っている．このことにより，専門家の注意を必要な部分に集中させることが可能となり，獲得の効率が向上している．

4.2.2 ETS/AQUINAS

MORE/MOLE では，いかにして専門家に自分自身が持つ知識を意識させるかということに努力が払われていた．このことに心理学的な立場から本格的に取り組んでいるシステムにETS[4-2-3]がある．ETS の最大の特徴は心理学における Personal construct Theory[4-2-4] を援用しているところにある．これは，人間が経験を概念に抽象化し，それを用いて新しく出会った環境を分類する（理解する）心理的な活動に関する理論であり，分類に必要となる属性(constructと呼ばれる）とそれらの間の依存関係を抽出するインタビュ技術が開発されている．ETS はこのインタビュ技術を用いて他のシステムと同様，分類型のエキスパートシステムを想定した知識の獲得を行う．

処理は分類に有効な属性の獲得とルール生成の２段階に分かれている．米国からの海外旅行相談システム構築のためのインタビュを例にとってその概略を述べる．（図３参照）ETS における初期のインタビュでは，任意に選ばれた３つの対象（この例では推薦すべき都市名）に対して，その中の２つに共通して，残りの１つが持たない属性の有無が次々に尋ねられる（例えば，「パリとミュンヘンには共通するがマイアミにはない性質を挙げて下さい」）．次に，得られた各属性に関する各対象の評価値が１から５の整数の範囲で求められる．図３の上部にはパリやロンドンなどの５つの都市と旅費や気候などの３つの属性が獲得された後の，各都市の各属性に関する評価結果が示されている．この行列はRating grid と呼ばれ，後で様々な解析に用いられる重要な情報である．その後，結果を総合的に解析して求められた属性間の従属関係が図３の下部に示すような形で提示される．太い矢印は強い依存関係を示している．この中の「よい博物館があるところは寒い」という依存関係に疑問を感じた専門家は，自分がアテネ（暖かくてよい博物館がある）を言い忘れてることに気付けば，新しい情報を追加することになる．次

に，OPS5のルールが生成され，テストデータを用いて簡単な検証をして，その結果を参考にしてルールを洗練する．ルールの生成ではＣＦ値の生成も行われる．ＣＦ値はEMYCINにおけるＣＦ値の結合方式を想定して，専門家が入力した値の相対的な大きさを参考にして決定されるが，ここでも専門家が入力するＣＦ値が信用されていないのは興味深い．

ETSはNeoETS[4-2-5]を経て現在ではAQUINAS[4-2-6]と呼ばれるシステムに発展している．AQUINAS は専門家にインタビュすることにより，彼らが知識ベースを解析し，テストし，洗練するのを支援する統合的環境として機能するように拡張がなされているが，インタビュシステムとしての本質は変わらないのでここでは省略する．

4.2.3　SALT[4-2-7]

SALTはVT[4-2-8] と呼ばれるエレベータの設計支援エキスパートシステムの知識獲得システムとして開発されたものである．VTはPropose&revise戦略に基づいて適切な近似解を提案してはそれを修正して行き，行き詰まればドメインに依存する知識を利用して適切な時点に後戻りするという処理を続けながら解を構成する．VTが必要とする情報には

(1)パラメータの値を求めるための手続き
(2)パラメータが満たすべき制約
(3)制約が満たされなかったときの処置法

の３種類がある．SALTは初めの二つに関しては，予め定められた順序に従って手続きの名前やその内容に関するインタビュを行う．手続きに関しては依存関係を表すネットワークが自動的に作成され，それに基づきループの検出などを行いながら適切な示唆を行う．３つ目のbacktrack に関する知識の獲得は最も重要な課題である．SALTの獲得戦略ではVTが用いるPropose&reviseに基づいて，システムが後戻りをするときに必要な知識が獲得される．　専門家は必要な知識の断片を述べるのには苦労を感じないが，Backtrack を含む様々な制御知識を体系立てて述べるのは得意ではない．SALTでは獲得された手続きのネットワークを利用して，制約が満たされなかった時のBacktrack 先（変更すべきパラメータ）を質問して必要な知識を獲得する．さらに，修正がループしないようにパラメータの依存関係が錯綜している場合には，依存関係の様子を表示し入力を支援する．

4.2.4　知的インタビュとそのアーキテクチャ

これまで眺めてきたシステムが示すように，インタビュによる獲得では，タスクにおける知識の利用の仕方に関する知識に基づき問題を解いてみることによって，不足する知識や矛盾の示唆等が行われている．このことから，上にも指摘したように，タスクに依存するインタビュ戦略の研究が本質的であること，また，インタビュシステムはその一部に問題解決システムを含まなければならないことがわかる．

知識獲得支援システムは知的にインタビュを行うシステムであるということができる．インタビュとはそもそも何であろうか？　上手なインタビュを行うにはどのような知識と機構が必要なのであろうか？　獲得しようとする知識に依存しない，汎用の枠組は存

在するか？　存在するとすればどのようなものか？　インタビュにはこのように多くの興味ある課題があるにもかかわらず，今まで議論が少なかった．

近年，インタビュを「専門領域の異なる者同志が一方の専門領域における知識の伝達を目的として行っている対話において，聞き手が行っている行為」と定義し，知識獲得のための基礎的研究として，知的インタビュ・システム[4-2-9] (I^2S : Intelligent Interview System)，及びインタビュシステム構築のためのシェル[4-2-10] (SIS: Shell for Interview Systems)の研究が行なわれている．そこでは，特にインタビュシステムを駆動するためのエンジン，トップダウンなインタビュ戦略，及び学習機能に重点おいてインタビュシステムの枠組みの開発が検討されている．

SIS は，インタビュシステムに関する経験を活かして抽象化をさらに進めることによって作成された，ネットワーク表現でドメインのモデルが構成できるタスクにおける知識獲得支援システムを構築するためのシェルということができる．

C-Prolog上のフレームとプロダクションシステムとによって実現されているが，さらにその上にインタビュ用のエンジンを設計してインタビュの戦略を記述するだけで，インタビュシステムが構築できるようになっている．システムはInterface, Domain model, Attention manager（インタビュエンジン），Supervisorの４つのサブシステムから構成されている．Attention manager はSIS の心臓部であり，SIS が提供する７つの質問戦略に基づいて記述されたインタビュ知識を解釈し，インタビュを実行する．７つの質問戦略は，MOREとデータベース論理設計支援におけるインタビュを詳細に検討することによって抽出されたもので，ネットワークで表現されるドメインモデルを洗練するための基本インタビュ戦略として構成されている．このように，SIS はインタビュに固有のアーキテクチャと基本知識を持っており，一種のドメインシェルと考えることができる．

<参考文献>

[4-2-1] G.Kahn, S.Nowlan and J.McDermott : "MORE: An Intelligent Knowledge Acquisition Tool,"Proc. of IJCAI'85, 1, pp.581-584 (1985).

[4-2-2] L.Eshelman and J.McDermott: "MOLE: Aknowledge acquisition tool that uses itshead," Proc. of AAAI '86. pp. 950-955(1986).

[4-2-3] J.H.Boose:"Personal construct theoryand the transfer of human expertise," Proc. of AAAI '84, pp. 27-33 (1984).

[4-2-4]Kelley, G.A. : The psychology of personal constructs," Norton, New York (1955).

[4-2-5]Boose, J.H., et al.: NeoETS: Capturing expert system knowledge inhierarchical rating grids, Proc. of Expert Systems in Government Symposium, pp.34-45 (1986).

[4-2-6]Boose, J.H., et al.: Expertise transfer and complex problems: using AQUINAS as a knowldge acquisition workbench for knowledge-based systems,J. of Man-Machine Studies, Vol. 26, No.1, pp.3-28 (1987).

[4-2-7]Marcus, S.: Taking backtracking with a grain of SALT, J. Man-Machine Studies, Vol.26, No.4, pp.383-398 (1987).

[4-2-8]Marcus, S. et al.: VT: An expert elevator design that uses knowledge-based backtracking, AI magazine, Vol. 8, No.4, pp.41-58 (1987).

[4-2-9]川口 他：インタビュ形式でユーザーと対話するデータベース論理設計支援システム，信学論，J70-D, No.11, pp.2243-2249, (1987).

[4-2-10]川口 他：インタビュシステムのためのシェル，ＳＩＳの開発，人工知能学会全国大会論文集，6-21 (1987).

４．３．ドメインモデルと深い知識

ドメインモデルは、ある問題領域に固有の知識を一定の形式で構造化したものである。知識獲得支援システムには、ドメインモデルを積極的に利用したことにより成功を収めているものが多い。まず4.3.1.節では、ドメインモデルを利用した知識獲得支援システムを紹介した後、ドメインモデルの利用形態についてまとめる。

深い知識は、より浅い知識を生成できるものとして、相対的に定義される。実際には、ある問題で対象にしているものの構造や機能を表現したものを深い知識とし、問題解決の効率を高める経験的知識を浅い知識とする場合が多い。次に4.3.2.節では、知識獲得における深い知識の利用について論じる。

ドメインモデルも深い知識も、知識獲得過程においてシステムが利用するものであるから、システムはこれらを予め内蔵している必要がある。最後に4.3.3.節では、知識獲得に先立ってシステムは何を知っているべきかについて論じる。

４．３．１．ドメインモデル

ドメインモデルは、ある問題領域に固有の知識を一定の形式で構造化したものであると定義することができる。しかし、一定の形式といってもネットワーク表現もあれば述語表現もあり、ドメインモデルの意味する範囲はひろい。さらには、その表現形式そのものをドメインモデルと呼んでいる場合もあるようである。ここでは、知識獲得支援システムや学習システムが獲得や学習過程において用いる、問題領域に固有の知識をひろくドメインモデルと呼び、その表現構造を（ドメイン）モデル構造と呼ぶことにする。

まず最初に、ドメインモデルを知識獲得や学習に利用して成功したシステムの代表例を紹介し、以下の議論の題材としたい。

(1) ＭＯＲＥ／ＭＯＬＥ [Kahn85][Eshelman86]

診断用知識の獲得を支援するＭＯＲＥ／ＭＯＬＥが、ドメインモデルを用いたシステムの典型例であろう（ＭＯＬＥはＭＯＲＥの発展版）。これらのシステムは、特定分野の診断用知識を表わすドメインモデルを構築するための枠組みを提供しており、専門家から知識を抽出しながらドメインモデルを構築していく。ＭＯＲＥ／ＭＯＬＥの提供するモデル構造は、仮説、徴候、および各種条件を因果関係のリンクで結付けたネットワーク構造になっている。システムは、獲得過程においてドメインモデルのネットワーク構造を静的に解析し、診断を行なう上で不足している知識の提供を専門家に促す。

(2) ＥＴＳ [Boose84][Boose87]

ＥＴＳは、分類型システムを構築するための知識獲得を支援するシステムで、分類すべき項目に、複数の観点から重み付けを行なったものをドメインモデルとして構築していく。分類型はモデル構造が単純なので、ＥＴＳがドメインモデルを構築

しているとは認めにくいが、一定の形式で知識を表現しているのであるから、ここではドメインモデルと考える。ただし、ＭＯＲＥ／ＭＯＬＥのようにドメインモデルを解析してインタビューを生成するといった積極的な利用はしていない。

(3) ＥＢＬ [Mitchell86][DeJong86]

説明ベース学習（Explanation-Based Learning, EBL）は、単一の具体例（training example）から概念を学習する演えき的概念学習で、有効な学習方式として最近注目されている（4.5.節参照）。複数の具体例から概念を学習する従来の帰納的概念学習（Similarity-Based Learning, SBL, とも呼ばれる）とＥＢＬが大きく異なる点は、ＥＢＬでは、領域理論(domain theory）と呼ばれる、システムが学習対象としている分野に固有な知識を持っており、学習過程においてこれが重要な役割を果たしていることである。

ＥＢＬの学習方式を簡単に説明する。一つの具体例が与えられると、システムはまず、その具体例がなぜ成立つかを領域理論を用いて解釈（説明）する。次に、その説明の仕方（説明構造）を基にして、与えられた具体例を汎化／特化し、与えられた具体例から概念を生成する。典型的なＥＢＬシステムでは、領域理論は所与のものとしてシステムに備わっている。

ＥＢＬは、知識獲得システムというよりは学習システムの分類に入るが、専門家から抽出した知識をより有効なものに変換する手段として、知識獲得支援システムに組込んで使うことができる。この場合、専門家から抽出された知識が一つずつ具体例としてＥＢＬサブシステムに与えられるというシステム構成になろう。

以上のシステム例に基づいて、知識獲得におけるドメインモデルの利用形態を整理すると、次のようにまとめられる。

(1) インタビューの生成

システムが専門家に対してインタビューをしながら知識を抽出する知識獲得方式においては、いかにして専門家の記憶を刺激して有効な知識を引出すかが問題である（4.2.節参照）。そのような刺激的なインタビューを生成するためにドメインモデルを利用することができる。例えば、ＭＯＲＥ／ＭＯＬＥでは、ドメインモデルの構造を静的に解析するとによって不足している知識の提示を専門家に促す質問を生成する。

(2) 知識の中間表現

専門家から抽出された知識は、システム内に保存され、最終的にはターゲットとする推論エンジン上で実行可能な知識表現に変換される。ドメインモデルは、抽出された知識を保存しておくための中間的な知識ベースとして利用できる。ＥＴＳおよびＭＯＲＥ／ＭＯＬＥでは、このような使い方をしている。

(3) 知識ベース管理の基盤

専門家から抽出された知識は、知識獲得支援システム内部の中間表現または推論

エンジン上で実行可能な表現で保存される。システムによっては、このような中間表現または実行可能表現の知識ベース上で一貫性・完全性などの管理が行われる。ここで、一貫性は、知識相互間に矛盾が無いこと、完全性は、所定の範囲の知識が網羅されていることである。

ドメインモデルは、中間表現として知識ベース管理の基盤として利用することができる。ＭＯＲＥ／ＭＯＬＥにおけるドメインモデルの解析による不足知識の指摘は、中間表現における完全性のチェックと考えることができる。

４．３．２．深い知識

専門家は、経験したことのない未知の問題に直面したとき、その分野の原理・原則に立戻って問題を解決していく。すなわち、表層的・経験的な知識がない問題に対しては、原理的・常識的な知識から問題解決に有用な知識を生成していると考えられる。これが、深い知識と浅い知識という概念の原点である。ここで、前者が浅い知識であり、後者が深い知識である。この観点によると、深い知識は浅い知識を生成できるものと定義することができる。

この２種類の知識は、エキスパートシステムの問題解決過程において利用される知識のレベルの深さに関する Hart [Hart82]、Michie [Michie82] による議論の中で初めて言及されたもので、その後、活発な議論がなされている [Chandra84] [山口87] 。これらの議論の中で、浅い知識のもつ問題点として次のものが指摘されている。

(1) 浅い知識の欠如

経験的知識が蓄積されていない新しいドメインでは、エキスパートシステムの構築は容易でない。例えば、次々と新しい機種が登場するような機械の故障診断や、次々と新しい製造法が開発されるＬＳＩ分野における設計支援などがある。

(2) 不完全性への対処

ある程度、経験的知識が蓄積されている分野でも、専門家の持っている全ての知識がエキスパートシステム化されているわけではない。したがって、知識が完全に記述されていないような状況に直面すると、システムは何ら対処できなくなる。

(3) 不十分な説明

浅い知識による説明は、基本原理に基づくものではないので初心者には理解しにくい。エンドユーザ向けの説明は深い知識を使った解説的なものが必要である。

知識獲得における深い知識の典型的な利用方法としては、深い知識を与えておいて浅い知識を生成することを支援することが考えられる。浅い知識の生成機構を組込むことにより上記(1)(2)の問題が解決される。 [山口87] は、深い知識を浅い知識を生成するためのものとしてとらえ、構築容易性および汎用性の観点から深い知識を分類整理し、浅い知識を生成する方式を提案している（知識コンパイラ）。

4．3．3．知識獲得支援システムの構成との関連

上述のように、ドメインモデルと深い知識は、知識獲得において重要な役割を果たす。これらを知識獲得に利用する上で重要な問題は、システムが予め何を知っているべきかということである。これによって、システムの構成が大きく変わってくる。システム構成のイメージについての包括的な議論は、７章に譲ることにし、ここでは簡単に次の２つの場合を検討する。

(1) 特定構造のモデルを与える場合

ＭＯＲＥ／ＭＯＬＥは診断タスクに、ＥＴＳは分類タスクに特定したドメインモデルの構造を提供しているが、診断タスク、分類タスクであれば、分野に関係なく利用できる。このように、システムには分野に関係なくタスクに特定した構造を与えておけばよい場合がある。

この場合、特定のタスクに適したモデル構造の設定が課題となる。今後の研究方向としては、a)現在提案されているモデル構造の精密化、b)タスクの類型的な分類・整理（4.4.節参照）が重要になろう。

(2) 特定分野のモデルを与える場合

ＥＢＬや浅い知識の生成を知識獲得に応用する場合、ベースとなる領域理論や深い知識を予め知識獲得支援システムに与えておく必要である。このように、ある特定分野の知識を与えておかなければならない場合がある。

この場合、ベースとなる知識をいかにして獲得するかが問題となる。一方法として、自然言語風に記述されたテキストを解析して特定分野の基礎知識を獲得する方法がある（4.1.節参照）。

[参考文献]

[Kahn85] G.Kahn, S.Nowlan and J.McDermott : "MORE: An Intelligent Knowledge Acquisition Tool", Proc. of IJCAI'85, pp.581-584 (1985).

[Eshelman86] L.Eshelman and J.McDermott : "MOLE: A Knowledge Acquisition Tool That Uses Its Head", Proc. of AAAI'86, pp.950-955 (1986).

[Boose84] J.H.Boose : "Personal Construct Theory and the Transfer of Human Expertise", Proc. of AAAI'84, pp.27-33 (1984).

[Boose87] J.H.Boose and J.M.Bradshaw : "Expertise Transfer and Complex Problems: Using AQUINUS as a Knowledge-Acquisition Workbench for Knowledge-Based Systems", J. of Man-Machine Studies, Vol.26, No.1, pp.3-28 (1987).

[Mitchell86] T.Mitchell, R.Kellar and S.Kedar-Cabelli : " Explanation-Based Generalization: A Unifying View", Machine Learning, Vol.1, pp.47-80(1986).

[DeJong86] G.DeJong and R.Mooney : "Explanation-Based Learning: An Alternative View", Machine Learning, Vol.1, pp.145-176 (1986).

[Hart82] P.E.Hart : "Direction for AI in the Eighties", SIGART, Vol.79, p.79 (1982).

[Michie82] D.Michie : "High-Road and Low-Road Programs", AI Magazine, Vol3, No.1, pp.21-22 (1982).

[Chandra84] B.Chandrasekaran and S.Mittal : "Deep versus Compiled Knowledge Approaches to Diagnostic Problem-Solving", Development in Expert Systems, M.Coombs, eds., pp.23-34, Academic Press (1984).

[山口87] 山口他 : "深い知識に基づく知識コンパイラの基本設計", 人工知能学会誌, Vol.2, No.3, pp.333-340 (1987).

4.4 ジェネリックタスク

(1) モジュール型知識処理システム

知識処理システムは，現状においては，ソフトウェア開発の効率が必ずしも高くない。この理由は，専門技術者の知識を取り出して知識ベースを作る工程があい路となるため，システム開発工数が大きくなることにある。現在の知識処理技術の最も大きな課題の一つに知識獲得があることは，応用分野を問わず共通の認識となっている。最近，システム構築ツールの普及により，知識処理システム作成の難しさは部分的に改善された面もあるが，知識獲得の難しさは克服されてはいない。

知識獲得のための技術開発においては，知識獲得を直接支援する手法やツールを実現することだけでなく，知識獲得が容易となるような知識処理システムを設計することが重要であるとの指摘が多い。米国Ohio州立大学のChandrasekaran教授は，上記の知識処理の問題点が，問題のレベルと知識表現のレベルのギャップが大きなことに起因すると分析し，ジェネリックタスク方法論[1]を提唱している。彼の言によれば，現在の汎用システム構築ツールが提供するルール，フレーム等の知識表現レベルは，従来の数値計算の例で言えば，アセンブラのレベルであり，せめてFORTRANのレベルのツールにする必要がある。ジェネリックタスク方法論は，その意味で，問題のレベルと汎用システム構築ツールのレベルの間に，新たな表現手段を提供することになる。ジェネリックタスク方法論の提唱以来，知識処理システム構成方法に関する理論が研究されているが，基本的な考え方は類似しているので，このような方法論に基づくシステムをシステム構成上の特徴に注目して「モジュール型知識処理システム」と総称する。

モジュール型知識処理システムの特徴は，知識表現のレベルに現われる。知識表現の階層構造として，言語(Lisp, Prolog, C, …)のレベル，汎用システム構築ツール(ルール，フレーム，…)のレベル，問題(診断，スケジューリング，…)のレベルを考えると，ジェネリックタスクに相当する知識処理モジュールは，問題のレベルと汎用システム構築ツールの間に位置する。知識処理モジュールは，言わば，問題解決のための標準部品であり，これらを組み合わせて与えられた特定の問題を解決するシステムを構成する。したがって，このレベルを応用分野向きシステム構築ツールのレベルと特徴づけることも可能である。

モジュール型知識処理システムでは，与えられた特定の問題を解くために，複数の知識処理モジュールを組み合わせた分散協調的なシステムを構成する。このとき，個別の知識処理モジュールには，そのモジュールの担当する情報処理に適した制御ストラテジ，知識表現・構成を持たせることができる。その結果，知識処理モジュールで使われる知識の獲得やその処理内容についての説明を，汎用的な形式ではなく，そのモジュールで行なわれている情報処理プロセスを反映した自然な形式で行なうことができる。

(2) ジェネリックタスク方法論

ジェネリックタスク(Generic Tasks)は，問題解決の手順に現われる自律的かつ汎用的な知識処理モジュールである。知識処理システムの作成に当たっては，これらのモジュールをビルディングブロックとしてライブラリー化しておき，それらを部品として組み合わせて目標とするシステムを作成するアプローチを採る。ジェネリックタスクは，知識処理システムを組み立てるための標準部品である。機械部品に対して，場当たり的に作ったり，似たようなものを作ったりしないよう工業規格の形で基準があるのと同様に，ジェネリッ

クタスクにも基準が必要となる。Chandrasekaranのグループでは，次のような基準を基に選択した知識処理モジュールとしてジェネリックタスクを規定している。

①入力と出力の対応によって記述した情報処理タスク仕様

②専門領域の知識の基本的な部品とその記述形式，および，これらの部品の構成方法

③推論の制御戦略

ジェネリックタスクの摘出は，慎重かつ厳密に行なわれている。現在の所，ジェネリックタスクは，公式には後述する６種類に限られている。

知識処理システムを作成する方法として，ジェネリックタスク方法論は，問題のタイプあるいは性質よりも，問題解決に必要な情報処理のタイプあるいは性質を重視している。このような立場は，ジェネリックタスクに与えた名称からも読み取れる。(ただし，中には「ルーチン設計」のようにまぎらわしいものもある。)上記の二つは，簡単には，問題の構造としてまとめられそうである。しかし，ジェネリックタスク方法論では，ジェネリックタスクは，情報処理タスクであり，問題の構造よりも，問題解決の構造を反映すべきであるとしている。ここでも，両者を区別し，問題構造という言葉だけでなく，「問題解決構造」という言葉も用いることにする。ジェネリックタスクの大きな特徴は，各タスクが問題解決構造の抽象レベル，すなわち人間による情報処理の抽象レベルのモジュールであることにある。端的に言えば，問題解決構造ではなく問題構造の抽象レベルでは，問題それ自体の抽象レベルに近すぎるということである。その結果，前述の「問題のレベルと知識表現のレベルの間にある大きなギャップ」をうまく埋めることができず，システム作成方法としての効力が限られると見る。Chandrasekaranの「正しい」抽象レベル("right" level of abstraction)という言葉は，この点を主張している。

(3)　ジェネリックタスクの種類

以下に６種類のジェネリックタスク[2][3][4]を紹介する。名称については，複数の呼び方をするものや文献によって異なる呼び方をするものが有り，必ずしも確定していない。ここでは，情報処理のタスクとして良く表現されている名称を中心に用いることにする。

(a) Classification

階層的に表現した分類項目の中で，対象がどれに当てはまるかを分類する。分類項目に関する知識を表現した各ブロックは，スペシャリストと呼ばれる自律分散型の知識ベースである。各スペシャリストは，分類の対象が自分の所に分類されるか否かを判定する知識を持っている。知識処理モジュールは，スペシャリストの階層をトップダウンに使って分類の対象が首尾良く分類できる項目を調べていく。このとき使われる推論の戦略は，Establish and Refineである。いま，あるスペシャリストが対象は自分の所に分類されると判定したとする。これは，対象を分類すべき分類項目(スペシャリスト)がその階層において決まった(Establish)わけであり，次は，その項目の下の階層の分類項目(スペシャリスト)を調べて行けばよい。したがって，そのスペシャリストは，自分の下にいるスペシャリストに制御を渡すことになる。この結果，分類は詳細化(Refine)される。

(b) Selection and Refinement

機械，回路，プログラム等の設計には，従来の設計例を引き出し(Selection)てきて，それを手直し(Refinement)して新たな設計案を作成する「ルーチン設計」と呼ばれるタイプの設計が少なくない。この場合，対象の構成要素は前もって分かっているので，設計の仕

様に合うように要素間の接続関係を変えたり，パラメータの値を修正していけばよい。このタスクは，「ルーチン設計」で使用する。

Selection and Refinementで用いる知識ベースは，Classificationの場合と同様にスペジャリスト階層となる。各スペシャリストは，特定の部品に関する情報を持ち，その部分の設計を担当する自律分散型の知識ベースである。スペシャリストの階層関係は，設計対象の部品の間の階層関係を反映したものに他ならない。

知識処理モジュールは，スペシャリストの階層をトップダウンに使って部品の情報を検索し，設計仕様に合うように詳細化していく。このプロセスで，上の階層にある部品を設計する処理に失敗すると，失敗処理機能を用いて，下の階層にある部品を設計する処理に自動的に再起動をかけ，バックトラックを行なう。

(c) Knowledge-Directed Data Retrieval

知的データベースの原型とも言える知識処理モジュールである。データに関する質問に対して回答を生成する。回答すべきデータが直接与えられていない場合には，(i)デフォールト値の使用，(ii)"if-needed"タイプのデモンによる付加手続きの実行，(iii)継承メカニズムを用いた検索，(iii)ヒューリスティックな推論等により，回答すべきデータを生成する。このデータに関する質問回答のやり取りは，メッセージ交信に他ならない。したがって，データを要求する側のモジュールと，データを提供する側のモジュールとは，全く自律分散的に構成することができる。

以上に説明したことから想像されるように，この知識処理モジュールは，フレーム・システムとして実現されている。

(d) Abductive Assembly of Hypothesis

説明すべき現象が与えられたときに，それを最も確からしく説明できる仮説の組を選び出し，これらを組み合わせて現象の説明を作成する。現象自体は，複数のデータ項目を用いて記述される。最も確からしい仮説の組を検索するプロセスは，Assembly and Criticismの推論制御により行なう。すなわち，仮説の組を作り(Assembly)，それではうまく説明できない問題点を見つけ(Criticism)，関連のあるデータ項目についての仮説を修正するという処理を繰り返す。このとき，知識ベースの中に格納した仮説の集合の間の因果関係，排反関係，包含関係等を利用する。

(e) Hypothesis Matching

与えられた仮説が，一般的には複数の条件により記述される目標とすべき状況，仕様等に一致するかどうかを検証する。目標のレベルから仮説のデータ項目のレベルまでにわたって，仮説を階層的にブレークダウンして表現しておく。階層の一番上は，目標の状況や仕様に対応し，それを達成するためには，その下にある条件を満足させる必要がある。階層の一番下のレベルでは，仮説のデータ項目とその内容を直接照合することができる。したがって，ボトムアップの検索を実行することにより，目標のレベルでの仮説の妥当性を検証することができる。

このタイプの知識処理モジュールは，状況駆動あるいは事象駆動的なプロセスを自然に記述することができるため，エンジニアリングの分野では応用範囲が広いと言える。

(f) Prediction by State Abstraction

サブシステムのレベルでの変化がシステムのレベルにどのような影響を及ぼすかを予測

する。もう少し詳しく言うと，次のようになる。サブシステムの状態の変化からサブシステムの機能の変化を導びく。サブシステムの機能の変化は，システムにとっては状態の変化となるので，次に，システムの状態の変化からシステムの機能の変化を導びく。

知識ベースは，予測の対象とするシステムの構造を反映して，階層的に構成する。ボトムアップの推論によって，部分から全体への影響波及を追跡する。この知識処理モジュールの行なう予測は，「もし何かが起こると，次に何が起こるか」というwhat-will-happen-ifタイプの予測であると指摘されている。

[参考文献]

[1]B.Chandrasekaran：Generic Tasks in Knowledge-Based Reasoning:High Level Building Blocks for Expert System Design, IEEE EXPERT, Vol.1, No.3, pp.23-30 (1986).

[2]T.Bylander and B.Chandrasekaran：Generic Tasks in Knowledge-Based Reasoning: The "Right" Level of Abstraction for Knowledge Acquisition, Int. J. of Man-Machine Studies, Vol.26, No.2, pp.231-243 (1987).

[3]市吉：ジェネリックタスクに関する説明資料，ICOT知識システムシェルワーキンググループ資料，KSS5-1 (1986).

[4]斎野：注目を集めた"Generic Tasks"――ＡＩアプリケーションの新しい枠組みが登場，日経ＡＩ，1986.9.15, pp.13-20 (1986).

4.5 説明に基ずく一般化を利用した戦略学習

知識の獲得に学習を用いようという考え方は一般的である。しかしながら、システムが自立的に学習できるためには、多くのことを予めシステムが知っていなければならない。そこでシステムに学習を行わせるために、基本的な知識を与えておかなくてはならない。

多くの問題分野において問題解決システムを構築しようとするユーザが、容易にかつ明確にシステムに与えることのできる知識は、問題の状態を変化させる公式的かつ基本的な書換えルール（オペレータ）である。しかし、この基本オペレータはその条件部が一般的過ぎる場合が多く、これをそのまま用いていたのでは前向き後向きに限らず全く無意味な展開が数多く生じ、探索空間が爆発的に拡大してしまい到底解にたどり着くことは不可能である。また基本オペレータは非常に断片的な知識なので細かいステップ毎に探索を行わねばならず、無意識にマクロオペレータを使っている人間の問題解決と比べると非常に効率が悪い。したがって、従来の問題解決システムでは、必要な時にのみ基本オペレータが適用されるようにその条件部を精錬するか、それともマクロオペレータを与えるか、または競合解消を工夫したりすることにより、事前に問題解決の戦略知識をシステムに与えていた。しかし、一般にこの様な戦略知識を与え、かつ知識全体として整合性を保つことはかなり困難なことである。比較的入力が容易な基本オペレータと、実際の解法例の二つを与えれるだけで、システム自身が問題解決の戦略知識を獲得することができればユーザの負担は極端に軽減する。

我々は、以上のようなパラダイムの基に問題解決における戦略知識の学習を行うシステムＰｉＬ(Paradigm-based inference Learner)の基本的動作と一次方程式、不等式などを含んだ初等数学の分野におけるケーススタディについて研究を行った。又、ＰｉＬの一般化の能力をさらに向上させるため、直接解決可能性に基ずく一般化についても研究を行った。

ＰｉＬは人間の教師に１次式の基本オペレータ及び解法例を与えてもらい、その解法例を肯定例として一般化することにより、書換えルールの条件部を

精製し戦略知識の学習を行う。図1にＰｉＬの構成図を示す。

図2に示すように、ＰｉＬは解法例が与えられると知識管理モジュールがそれを一般化していき、その結果がヒューリスティックオペレータとして蓄えられる。さらに、得られたヒューリスティックオペレータの重み付けを行うことにより絶対オペレータを抽出して、次に完全因果性という基準の基に特定のシーケンスをマクロオペレータとして取り出しす。

一般に、人間はある特定のオペレータのシーケンスをマクロオペレータとして持っており，それを用いて問題解決を行っている．そこでＰｉＬも解法例から特定のシーケンスをマクロオペレータとして抽出することにより問題解決の効率を上げてやる必要がある．しかし，どの様な基準でマクロオペレータを選別して抽出するかが重要である．なぜなら，解法例におけるオペレータのシーケンスの任意の部分的シーケンスがマクロオペレータの候補となるが，その中には全く無意味なものが多く含まれているからである．そこでＰｉＬでは，「人間はオペレータ間に閉じた強い因果関係があるときにそのオペレータのシーケンスをマクロオペレータとして取り込む」という仮定に基づいた完全因果性という基準でマクロオペレータの抽出を行う．

さらに、抽出されたマクロオペレータの内で終了条件まで含んでいるもの，つまりそのマクロオペレータが適用されれば直接的に問題が解決されてしまうものには，直接解決可能:DS(Directly Solvable)というラベルを付けておき、この直接解決可能なマクロオペレータ（ＤＳマクロオペレータ）の条件部の論理和を用いて直接解決可能性に基づく一般化を行う。

ＰｉＬでは、条件連鎖に基づく一般化を用いることにより学習の効率が帰納的一般化に比べてはるかに向上した。しかし，一次・二次方程式からさらに分数方程式，対数・指数方程式へとその解法が複雑になるに従って教師が与えなければならない解法例の量も増え，かつそれらの中には教師からみると冗長なものがかなり含まれてくる．そこで、過度に一般化しない範囲でさらに十分な一般化を行うことにより学習の効率化を図る必要がある．

それが直接解決可能性に基ずく一般化によって行われる。直接解決可能性をもつ問題状態とは，過去において直接解決可能であった問題状態であり，直接解決可能な状態とは前述の様に目標状態を含んでいるマクロオペレータ（

ＤＳマクロオペレータ）の条件部の和集合に含まれる問題状態を意味する．具体的には，直接解決可能性の有無を判別する述語ＤＳをＤＳマクロオペレータの条件部の論理和で表わす．つまり，ある問題状態に対して現在蓄えられているＤＳマクロオペレータの何れかが適用可能なとき，その問題状態は直接解決可能であるとかんがえる．そのため、この直接解決可能性という概念は学習の過程で新たなＤＳマクロオペレータが得られる度に逐次的に更新されねばならない．

いま，図３に示すようにあるマクロオペレータ：ＭＯ＿１が抽出され，それがＭＯ＿２というＤＳマクロオペレータを包含しているとき，ＭＯ＿１についてＤＢＧ（Directly solbable Based Generalization）が適用可能となる．ＤＢＧは解法シーケンスにおけるＭＯ＿２の領域を切り離して残りの領域Ｒ（＝ＭＯ＿１＆notＭＯ＿２）内のみで条件連鎖に基づく一般化の伝搬を行うこのとき，一般化伝搬の最適経路を改めて行う．これはＤＢＧでは目標状態から条件連鎖した場合とは異なった一般化経路が選択される場合があるからである．このような一般化を行い，実際の問題解決時には解法シーケンスＲを通過して得られた結果を今まで得られた全てのＤＳマクロオペレータの条件部の論理和である述語ＤＳでチェックする。従って，一般化伝搬の際にＭＯ＿１の条件部にはＭＯ＿２の拘束条件が直接的には伝搬されないので，ＤＢＧはＭＯ＿１の条件部を条件連鎖に基づく手法だけの場合よりもさらに一般化することが可能であり，さらに述語ＤＳによるチェックをしているので一般化され過ぎることはない．また，述語ＤＳを構成している各ＤＳマクロオペレータの条件部を全てＲを通して後方連鎖させ，その結果を条件部に選言としてもつことにより，問題解決時にＭＯ＿１の上層部で効率的にＭＯ＿１の適用可能性を調べることができる。

このようにして、解法例から条件連鎖に基づく一般化よりもさらに十分かつ一般的な一般化がＤＢＧにより実現され，その結果学習の効率化が実現された．

少数の解法例を解析することにより問題解決の戦略を学習するシステムＰＩＬの基本動作及び直接解決可能性に基づく一般化について報告した．今後は問題解決モジュールを述語表現に対応可能なものに交換することにより、

様々な分野の問題解決に応用可能であると考えられる.

参考文献

[1] Mitchell,T.M., Utgoff,P.E. & Banerji,R. : Acquring and Refining Problem-Solving Heurristic, pp.163-190, in Machine Learning-An artificial intelligence approach vol.1, Tiago Pub. Co., Palo Alto (1983)

[2] 山田, 安部, 辻: 問題解決に於ける戦略学習システム: ＰｉＬ１方程式、不等式でのケーススタディー, 人工知能学会誌, pp.206-215 (1988)

[3] A. パンディ: メタレベル推論と意識, pp.187-203,「ＡＩと哲学」, 産業図書 (1985)

[4] Mitchell,T.M., Keller & Kadar-Cabelli : Explanation-Based Generalization : A Unifying View, pp.47-80, Machine Learnig, 1-1 (1986)

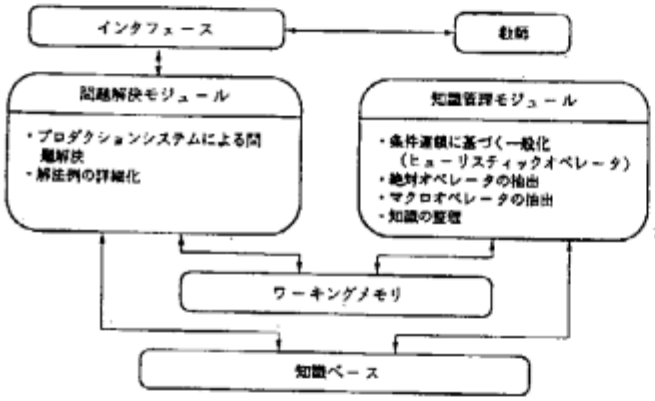


図1 PIL構成図

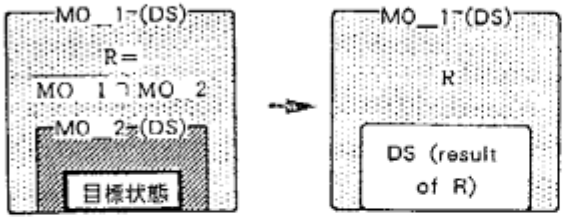


図3 DBGの適用対象

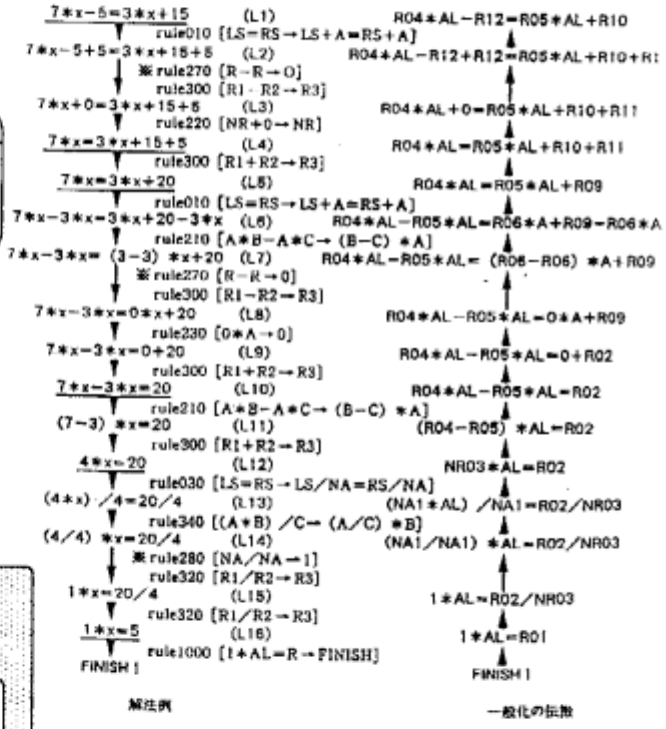


図2. 解法例の一般化の過程

5　問題解決構造の分析

5.1　問題構造と問題解決構造

診断あるいは設計というような応用分野の特徴に注目して，知識処理システムを作成するべきであるとの指摘は，必ずしも少なくない。エキスパートシステムのテキストにおいても，いくつかの分野に分類して論じられることが多いようである。[1][2] 例えば，上野(電機大)は，データ解釈，診断，監視，制御，設計の５種類に分けており，小林(東工大)は，データ解釈，診断，監視・制御，計画・設計の４種類に分けている。このような形で自然に，問題のタイプや性質に従って，エキスパートシステムの作成方法を整理していく方法論が発展している。ここでは，このような立場を問題構造アプローチと称する。

一方において，エキスパートシステムの作成方法に関しては，Chandrasekaran[3]のジェネリックタスク方法論がある。ジェネリックタスク方法論の基本的アイデアは，以下のように要約できる。

(ｉ)知識ベースシステムにおける問題解決機能は，インプリメンテーションのレベルではなく，より上位のタスクレベルで捉えるべきである。

(ii)知識の表現法およびその構成は，タスクレベルで捉えた問題解決のための制御ストラテジに依存すべきである。

(iii)複雑な問題解決は，それぞれ(ii)に規定した，ある種の知識と制御ストラテジから構成されるジェネリックタスクの組合せで処理することができる。

ジェネリックタスク方法論については，既に4.4節でも紹介している。そこでは，問題構造と問題解決構造の相違についても触れた。問題構造アプローチに対比して，ジェネリックタスク方法論は，問題解決構造アプローチと言える。問題解決構造アプローチの例としては，Clancy[4](Stanford大)がＭＹＣＩＮの診断プロセスをHeuristic Classificationというタイプの問題解決構造に帰着されることを明らかにしたことを挙げられる。

ジェネリックタスク方法論は，問題解決のプロセスにおける人間の情報処理的側面に注目する。問題の構造自体は，論理あるいは数理の世界の存在であると言ってもよく，必ずしも人間が行なう情報処理には関連を持たない。ジェネリックタスク方法論は，問題解決における人間の情報処理から出発している点，あるいは実際に知識処理システムを作成する過程を通じてジェネリックタスクを発見していく点から，ボトムアップ的であり，そこに知識処理システムの作成方法としてのパワーがある。

問題構造アプローチとしては，小林(東工大)の提案したもの[5][6]が，詳細であり，整理されている。応用分野あるいは問題の特徴に注目して知識処理システムを作成する方法論の代表例であると言える。対象とする問題を大きく解析型と合成型にタイプ分けし，合成型をさらに三つのクラスに分けている。これら四つの問題タイプは，①システム構造，②システム特性，③構成要素特性の間の入出力関係によって，系統的に分類したものになると説明している。さらに，解析型問題を，解釈問題，診断問題，制御問題に細分化し，合成型問題を，計画問題，設計問題に細分化する。

表１は，上記の５種類の問題に対し，基本タスクと問題解決機能を整理したものであり，問題構造アプローチによる知識処理システム作成の方法論の基本的な考え方を表わしている。ここで，基本タスクとは，問題タイプに共通する部分問題であり，問題解決機能とは，その部分問題を解決する上で有効な手法であると見なすこともできる。

ジェネリックタスク方法論に代表される問題解決構造アプローチに比して，問題構造アプローチは，トップダウン的であり，分かりやすい。これは，一言で言えば，知識処理システムの知識表現の階層において，ジェネリックタスクよりも問題の側に近く位置するためである。ジェネリックタスクが，問題解決における人間の情報処理から出発しているのに対し，問題構造アプローチは，問題解決はかくあるべしという所から出発している。両者はきわめて対照的である。ただし，基本タスクや問題解決機能には，当然のことであるが，ジェネリックタスクと同じ抽象レベルにあるものが現われている。

5.2 既存システムの分析

ジェネリックタスクの考え方とは独立に既に数多くの知識処理システムが開発されている。６種類のジェネリックタスクがカバーしている範囲を把握するため，既存の知識処理システムを対象として，問題解決構造の分析を試みた。今回の分析では，その目的からして，欲張らず広く浅くを旨とした。分析の対象としたシステムは，下記の日立評論の知識処理関連特集号に掲載された論文の中に含まれる応用システム25件である。

・日立評論，Vol.67，№12 (昭60-12)

・日立評論，Vol.69，№３ (昭62-3)

ここでは，ある意味で統計的に知識処理システムの問題解決構造を摘出することを試みている。したがって，次のような点から，個々のシステムについては，必ずしも確定的なものとは言えない。

(ｉ)論文の記載内容だけから判断した。システムによっては，多くの機能の一部しか記してないものも有りうる。またシステムのプロトタイピングの過程で手法を変更したものも有りうる。

(ii)複数の問題解決構造を読み取れるものもあったが，代表的なものだけに限定した。どちらとも取れるようなケースについて，現状のジェネリックタスクに含まれていない問題解決構造がある場合には，それを重視した。

当然のことであるが，システムの問題解決構造は，システム自体の技術レベルや完成度とは無関係である。

分析結果を表２に示した。この結果より，以下のようなことがわかった。

(イ)ジェネリックタスクに含まれるものは，18／25で７割程度である。中でもHypothesis Matchingは，制御の分野を中心に８件あった。

(ロ)ジェネリックタスク以外のものは，Generate and Test，Constraint Modification，Constraint Synthesis，Constraint Satisfactionの４種類であった。ただし，これらが，ジェネリックタスク相当の単位となるべき問題解決構造である否かについては議論の余地がある。

ジェネリックタスク以外の問題解決構造について以下に簡単に紹介する。

(a) Generate and Test

解の候補を生成するステップと，それらをテストするステップから成る。テストを満足するものを求めることにより問題の解を得る方法である。この方法を，階層的に使用する場合もある。問題解決戦略としてよく研究されている。Generate and Testは，はっきりと二つのステップに分けられることからも，より基本的な問題解決構造に分けて考えるこ

ともできる。

(b) Constraint Satisfaction

問題を制約条件を満足する可能解を求める形式に定式化し，変数の間の関係を利用して変数の取りうる値の範囲を絞り込み，すべての制約条件を満足する変数の組を探索する方法である。

(c) Constraint Modification

問題の解を探索する際に制約条件として定式化する条件の中に緩和可能なものが含まれる場合がある。制約条件が厳し過ぎるために解が見つからない場合，これらの条件を緩めることにより，解が見つかる可能性を高めた状態で，再度，解の探索を繰り返す。このようなプロセスにより，最終的には，問題の解を得ることができる。

このような問題解決構造は，何らかの目標がある場合に，まず，制約条件の組によって目標を設定し，それを解としたときに制約条件を緩め，目標の妥協をはかるプロセスを繰り返し，初期の目標に近い所に解を得るプロセスと見ることができる。

(d) Constraing Synthesis

問題解決のプロセスで，有効な制約条件をダイナミックに決定して，解を効率的に探索できる場合がある。特に，制約条件の数が一般的には多くても，解の探索点の局所的な状況から，効力をもつ制約条件が限定できる場合に効果が大きい。

このような問題解決構造は，何らかの意味でフォーカス・メカニズムが介在し，全体を見渡すのではなく，ある部分に注目することにより，ダイナミックに問題を部分問題に分割していくプロセスと見ることができる。

5.3 知識処理システムに組み込まれたジェネリックタスク

ジェネリックタスク方法論に基づく知識処理システムの開発例として，プラント運転ガイダンスシステムについて紹介する。なお，この例は，昭和63年3月 に情報報処理学会にて発表された山田，Chandrasekaranによる研究事例[7]である。

(1) 背景

原子力発電プラントのように複雑で大規模なシステムの運転には，高い安全性と信頼性が求められる。この分野では，知識処理の技術を応用してヒューマンエラーの低減を図ることが期待されており，時々刻々変化する動的な環境で行なう問題解決を支援する知識処理システムの開発が注目されている。知識処理システムDPMS(Dynamic Procedure Management System)はそのようなシステムであり，特にプラントの運転を対話的に支援することを目的としている。

DPMSは，運転法合成を目的としたシステムDPSRL[8]を発展させ，動的環境で使用するために，変化する状況に迅速に対応できるようにしたシステムであり，知識処理システム作成のためのジェネリックタスク方法論に基づいて構成されている。ここでは，DPMSのタスク構成，各タスクで使用する知識の形式，構成，および問題解決ストラテジについて紹介する。

(2) 運転ガイダンス

ここで，運転ガイダンスとは，特に緊急時にプラントの運転員が実行すべき一連の運転操作をプラントの状況および運転履歴から判断して提示することを指す。これを行なうた

めには，プラントの状況判断，適切な運転操作の決定，操作不具合時の回復操作の決定等を時々刻々変化する状況のなかで実施することが要求される。このようなタイプの問題解決は，人工知能の分野では計画(Planning)に分類されるが，通常の計画と比較して以下のような特徴がある。

(a)事前に決定された豊富な運転操作手続きが利用できる。

例えば，原子力発電プラントにおいては，各種の事前に想定した事象に対する一連の運転操作法(Enent-orientd Procedure)や，発生事象に関係なくプラントの安全性，健全性を維持するために設定された運転操作法(Function-oriented Procedure)が利用できる。したがって，一連の運転法を各種の単一の運転手続きから構成する必要がない。

(b)問題解決を行う環境が時々刻々変化する。

時間とともに変化する状況に応じて実行すべき運転手続きを選択決定し，それらが利用可能であるか否か，またそれらが正しく実行されたか否かを動的に判断する必要がある。さらに，新たに発生した状況に対して，問題解決のフォーカスを即座に変更して柔軟に問題解決を進めることが必要である。

(c)維持すべき目標(Maintaing Goal)と達成すべき目標(Achieve Goal)が存在する。

緊急時のプラント運転では，安全性の確保が第一目的である。したがって，安全性，健全性を維持するために設定された運転操作法が，運転法の決定や不具合時の回復操作において優先的に処理される(Defense in Depth)。また，達成すべき目標は，一連の運転手続きにおいてそれらの実行結果の判断や，代替手続きの指定に利用する。

以上の特徴を考慮して，DPMSではEvent-Oriented ProcedureとFunction-Oriented Procedureを組み合わせて利用し，安全上の重要度から決められる安全目標に関する階層(Safety Goal Hierarchy)を基に，それらを制御する方法を採用している。

(3) DPMSのタスク構成

問題解決のプロセスは，ジェネリックタスク方法論[9]に従ってタスクレベルの処理により実現している。DPMSでは運転ガイダンスを図１に示すように次の４タスクから構成している。

(a) 安全目標同定タスク

プラントから得られる各種データおよびそれらの解釈結果に基づいて，その状況で脅かされている安全目標を同定する。使用する知識は，観測結果およびその解釈結果の組合せと，その状況で脅かされている安全目標仮説に対する確信の度合との関係である。安全目標階層の中にこれらの安全目標を重要度にしたがって配置しておき，トップダウン探索により安全目標を同定する。

(b) 事象類別タスク

プラントから得られる各種データおよびそれらの解釈結果に基づいて，その状況で発生していると考えられる事象を選択する。Classificationのタスクである。使用する知識は，観測結果およびその解釈結果の組合せと，その状況に適合した事象仮説に対する確信の度合との関係である。事象類別は，事象の仮説階層をトップダウンに探索して事象を同定する。この制御は，Establish and Refineとして知られている。

(c) 運転ガイダンス制御タスク

安全目標同定および事象類別からの指示を受け，実行すべき運転法を決定する。このタ

スクは，制御タスクであり特別な知識を持たない。問題解決の制御は，事象階層中の各事象を，その事象の発生により脅かされる安全目標との関係を考慮して安全目標階層に埋め込んだ拡大安全目標階層を利用して行う。

(d) 運転ガイダンスタスク

選択された安全目標あるいは事象に対応した運転法を指示する。操作不具合時には代替操作や回復操作を指示する。Selection and Refinement[10]に類似したタスクと見ることができる。このタスクでは，上述した各Event-Oriented ProcedureおよびFunction-Oriented Procedureを，それを管理実行する単位であるスペシャリストの下で，プランおよびプロシジュアとして階層的に表現する。

・プランは，一連の操作を具体的操作とプロシジュアの順序付けられた集合の形式で持ち，実行の前提条件，実行タイプ，対応する安全目標，実行結果の判定条件を属性として持つ。

・プロンジュアは，より下位のプロシジュアや具体的操作の順序付き集合で定義し，実行の前順提条件，実行タイプ，想定される安全目標，実行結果の判定条件等を属性として持つ。

問題解決の制御は，これらの階層をトップダウンに探索するとともに，実行できない操作や，効果が期待通りでない操作に対しては，ボトムアップに代替操作や回復操作を探索することにより行われる。

[参考文献]

[1]上野：知識工学入門，オーム社 (1985).

[2]小林：知識工学，昭晃堂 (1986).

[3]B.Chandrasekaran：Generic Tasks in Knowledge-Based Reasoning:High Level Building Blocks for Expert System Design, IEEE, Vol.1, No.3, pp.23-30 (1986).

[4]W.J.Clancy：Heuristic Classification, Artificial Intelligence, Vol.27, No.3, pp.289-350 (1985).

[5]日本情報処理開発協会：知的情報処理システムに関する調査研究報告書——知識処理システム開発方法論，62A-001 (1987).

[6]小林：問題解決と類型的タスク，ICOT知識表現サブワーキンググループ資料 (1987).

[7]山田，B.Chandrasekaran：ジェネリックタスク方法論に基づくプラント運転ガイダンスシステム，情報処理学会第36回(昭和63年前期)全国大会，3P-6，pp.1429-1430 (1987).

[8]D.Sharma：PhD. Dissertation,the Ohio State University (1986)

[9]B.Chandrasekaran：Toward a Taxonomy of Problem-Solving Types, AI Magazine, Vol.4, No1, pp.9-17 (1983)

[10]D.Brown and B.Chandrasekaran：Knowledge and Control for Mechanical Design Expert System, Vol.19, No.7, pp.92-100 (1986).

表1　個別問題領域の基本タスクと問題解決機能〔文献5より引用〕

		基本タスク	問題解決機能
解析型問題	解釈問題	1．特徴の抽出	1．分類階層によるクラス分け
		2．モデルとの不完全な照合	2．モデル検索機能
		3．システム構造の同定	3．部分構造の評価機能
		4．システム状態の推定	4．階層的生成検査法
		5．あいまいさの処理	5．不確実性推論
		6．不完全性の処理	6．仮説推論
		7．解釈の多様性	7．協調型推論
	診断問題	1．異常事象の分類階層的表現	1．事象駆動型推論
		2．システム構造の階層的表現	2．目標駆動型推論
		3．解釈問題のタスクを内包	3．不確実性推論
		4．計測点の選択的決定	4．仮説推論
		5．異常原因の同定	5．協調型推論
		6．浅いモデルによる効率性	6．効率性と完全性の調和
		7．深いモデルによる完全性	7．費用／効果分析
	制御問題	1．システム構造の表現	1．状態の診断機能
		2．システム動特性の表現	2．シミュレータによる予測
		3．診断問題のタスクを内包	3．定性推論による予測
		4．モデルによる状態の予測	4．制御則の多目的評価
		5．安定化操作の優先的実行	5．アクションガイダンス
		6．安定操業下での省エネ化	6．デッドロックの解析
		7．オペレータガイダンス	7．インターロックの回避
合成型問題	計画問題	1．計画過程の階層化	1．トップダウン精密化戦略
		2．組合せ的探索	2．拘束最小化戦略
		3．制約条件の相互作用	3．網羅的探索機能
		4．環境の予測	4．仮説推論による知的探索
		5．上位レベルへの知的後戻り	5．類推による計画事例の利用
		6．計画事例の検索と利用	6．計画の多目的評価
		7．計画評価の多用性	7．リスク下での意思決定
	設計問題	1．構成要素とその関連の表現	1．トップダウン精密化戦略
		2．設計問題の階層的表現	2．拘束最小化戦略
		3．代替案の自動生成	3．最適化機能
		4．部分システムの評価	4．検証機能
		5．上位レベルへの知的後戻り	5．仮説推論による知的探索
		6．設計事例の検索と利用	6．類推による設計事例の利用
		7．並列的問題解決	7．協調型推論

表2　エキスパートシステムと問題解決構造

システム	問題解決構造（記述内容からの推定）
1 故障発生区間判別 エキスパートシステム	Generate and Test
2 加熱炉燃焼制御 システム	Constraint Modification
3 運転ガイダンスシステム	Classification
4 事象駆動型制御システム	Hypothesis Matching
5 配管ルーティングシステム	Constraint Synthesis
6 変電所機器レイアウト システム	Selection and Refinement
7 未知物体の認識システム	Classification
8 環境変化の検知システム	Hypothesis Matching
9 物流ＦＡシステム （工場内物流システム）	Hypothesis Matching
10 鉄鋼ＦＡシステム （整備自動運転システム）	Hypothesis Matching
11 パッケージ組立ＦＡシステム （無人搬送者制御システム）	Hypothesis Matching
12 計算機室機器レイアウト エキスパートシステム	Constraint Satisfaction

13 計算機システム構成設計支援エキスパートシステム	Selection and Refinement
14 半導体プロセス診断システム	Generate and Test
15 対話形論理設計支援エキスパートシステム	Selection and Refinement
16 回路系自動翻訳エキスパートシステム	Hypothesis Matching
17 論理回路検証エキスパートシステム	Hypothesis Matching
18 地図情報エキスパートシステム	Knowledge-directed Information Retrieval
19 知的ファイリングシステム	Knowledge-directed Information Retrieval
20 ワークスケジューリングエキスパートシステム	Constraint Satisfaction
21 フェーシングコントロールエキスパートシステム	Constraint Satisfaction
22 出店計画支援エキスパートシステム	Selection and Refinement
23 融資エキスパートシステム	Classification
24 相続相談エキスパートシステム	Classification
25 資金運相談エキスパートシステム	Hypothesis Matching

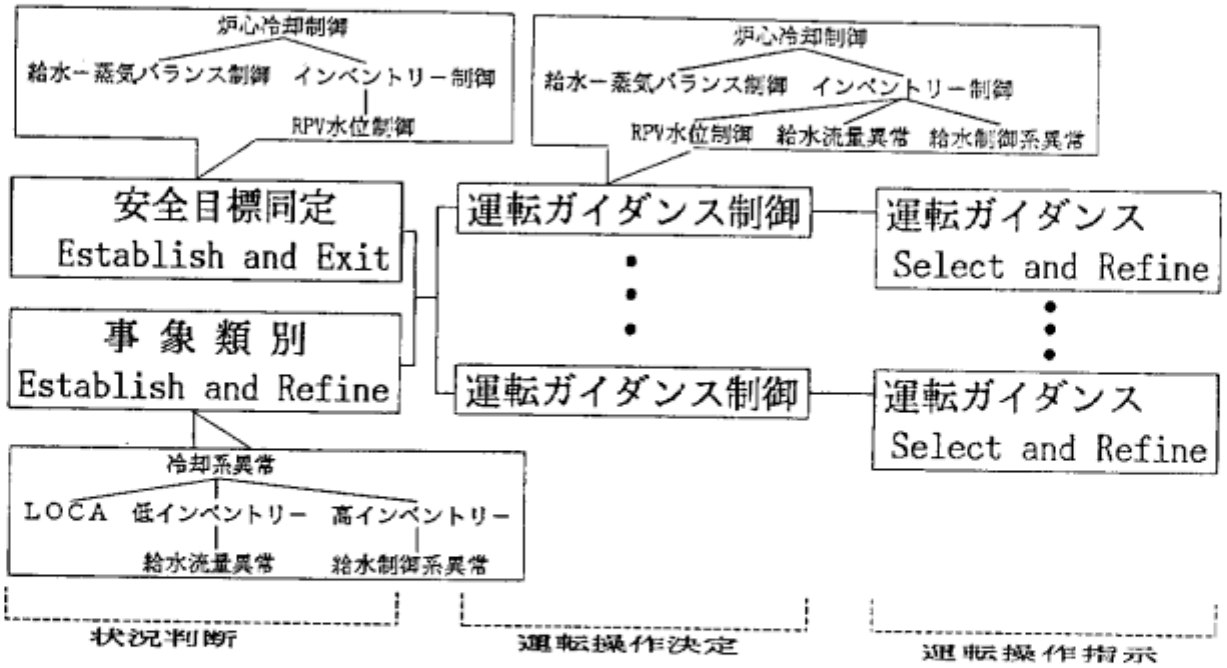


図　運転ガイダンスシステムのタスク構成

6.1 AQUINAS

AQUINASは，分類型の問題向け知識獲得支援システムETSの拡張システムである．ETSは，G.KellyのPersonal Construct Theoryに基づいたシステムで，以下のような順で，知識の抽出，洗練，知識ベースの生成を行なう．最初に分類すべき項目を専門家から聞き出す．そして，入力された項目群に対して三つ組法を適用することによりそれらの項目を分類するのに必要な特徴の組（Constructと呼ばれる）を抽出する．次に項目を列方向に，特徴を行方向に持つ表を作成し，成分として項目と特徴との関係度を表わす評点（1-5，N，?）の入力を促す．　この表に対しクラスタ解析，特徴間の因果関係の解析を行なうことにより，項目，特徴の組，評点について洗練を促す．その結果，最終的に得られた表から特徴と項目間，特徴間の関係を表わすルールを生成する．

しかし，ETSでより複雑な分類型問題を扱おうとした場合，次のような問題が生じる．

- 一枚の表だけでは、項目と特徴間の対応関係しか表現することができず、関係の連鎖や多項関係が表現できない。
- 一枚の表において、項目や特徴は、同じレベルの抽象度に統一されていなければならない。
- 複雑な問題を対象とする場合、表が大きくなり表の理解が困難になる。
- 複数の専門家、いくつかの推論戦略、領域モデルのそれぞれが階層構造を持つような知識を一つの表で表現することはできない。
- VAX,IBM,･･･という値を持つcomputerという属性を表現するためには、特徴の組からなる列(VAX/NOT-VAX,IBM/NOT-IBM,･･･)で表現せざるをえない。
- 順序尺度上の１から５までの評点でしか表現できず、他の尺度による評点ができない。　あるいは，属性の値を直接に表現できない。

そこで，AQUINASではこれらの問題を解決するために，項目と特徴の２次元に加えて専門家、caseの次元を設け，さらにそれぞれの次元について階層表現を導入すると同時に，評点の型として複数のものを採用している．以下，AQUINASの特徴について解説する．

(1)階層的知識表現（図6.2-1参照）

項目（分類問題における解）の階層は，段階的詳細化やクラスタ解析を行なうことにより，Super-Sub,Instance-ofの関係からなる階層構造を持つ．項目の階層における各レベルに対応して特徴を抽出することにより、特徴の階層が作られる．

項目，特徴，評点は，専門家個人毎に異なるものであるので，個人毎に表をもたせている．これにより専門家間の知識の比較が行なわれる．さらにGroup-Subgroup，

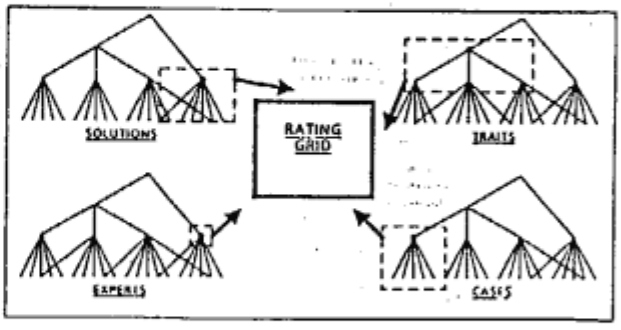


図6.2-1　階層的知識表現

Individual-Groupの関係による階層構造を導入することにより，複数専門家からなる判断を可能としている．

問題解決における視点により，扱う知識の種類は異なる．そのため，異なる視点に応じて表が持てるようになっている．さらに，階層構造を持たせることにより個別の視点から一般的な視点まで対応している．

(2)評点の型の拡張

1から5の順序尺度以外の尺度として，名義尺度(nominal)，間隔尺度(interval)，比率尺度(ratio)が扱える．評点は，知識の投入者により直接与えられるか，上位階層の知識から継承されることにより得られる．　また，必要に応じて異なる尺度間の変換が行なわれる．（例えば，テスト推論時に，一対比較を用いて名義尺度から比率尺度に変換する．）

(3)テスト推論法

知識の階層化にともない，問題解決法も拡張される必要がある．AQUINASでは，Clanceyのabstraction-refinement modelに基づいた方法を採用している（図6.2-2参照）．つまり，特徴の階層についてdata-abstractionを行い，項目の階層についてrefineを行なう．

また，評点のデータ型の拡張にともなう推論の種類として，確定的判定を行なうabsolutely reasoning，不確定な判定を行なうrelativistic reasoning（ex. MYCIN流の確信度，Fuzzy logic，AHP），確率理論に基づいたProbablistic reasoning（ex. Bayesean，Dempster-Shafer）がある．

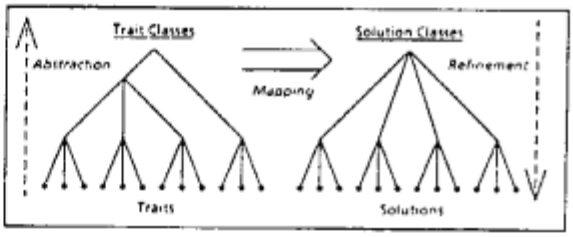


図6.2-2　問題解決モデル

(4)その他の特徴

以上紹介したように，AQUINASは多数の機能を持つ．このため初心者のユーザにとっては，必要な機能の選択にとまどう場合がある．AQUINASは，初心者に対してシステム側から適切な誘導を行なうようなDialog Managerと呼ばれる機能を持っている．AQUINASのシステム構成を図6.2-3に示す．

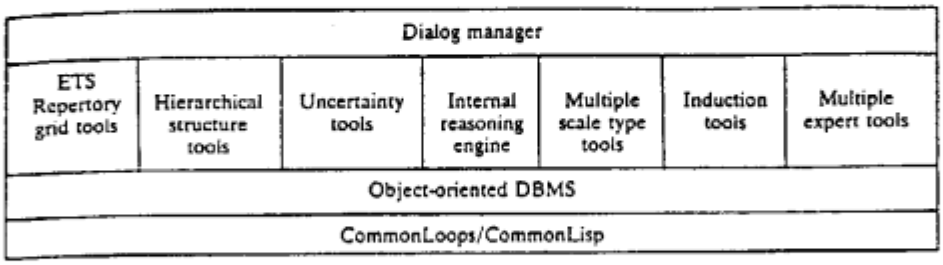


図6.2-3 システム全体構成

6.2 KNACK

KNACK[Klinker 86][Klinker 87]は異なるクラスのシステムデザインの評価用エキスパートシステムを作成するための知識獲得支援システムである。KNACKは作成される知識システムの処理するタスク(デザイン評価)の問題解決方法と評価タスクに基づく詳細なドメインモデルを使用することによってデザイン評価の専門家から知識獲得を行う。

KNACKによって作成されるエキスパートシステムはWRINGERと呼ばれる。WRINGERの処理するタスクは、システムデザイナーからシステムのデザイン情報と使用される環境の情報を収集すること、その情報を基に評価すること、及びデザイナーに対してデザインの訂正と改善の提案を行うことである。そして評価タスクの主要な問題解決方法は、

(i)有効なデザイン記述を作成するためにデザイナーから必要な情報を収集する方法

(ii)構成的な手法で一貫性、完全性、デザインフローの点からデザインを評価する方法

である。

ドメインモデルは、システムデザインが未知の環境(システムが使用される状況)を考慮して評価されなければならないため、一般的な評価タスクの解析結果から、評価ドメインに共通の概念で構成されている。KNACKにはこの問題解決方法とドメインモデルに関する知識が組み込まれている。図1にドメンモデルの雛型を示す。

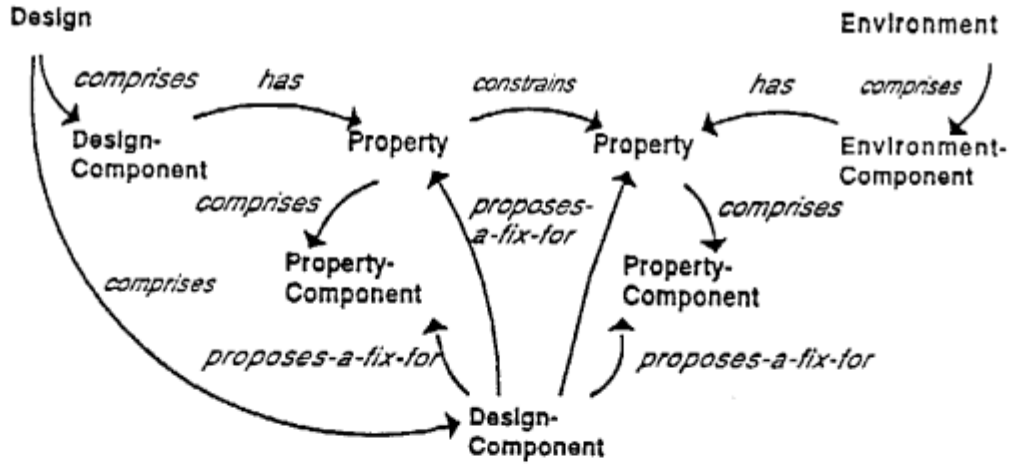


図 1 評価ドメインモデルの雛型

KNACKのコサルテーションは初期ドメインモデルの獲得とドメインモデルのリファインの2つの段階からなる。まず初期ドメインモデルの獲得では、KNACKは専門家との初期インタビューとレポート形式(サンプルレポート)での入力が行われる。この初期対話により入力された知識をカスタマイズしてドメインの初期ドメインモデルを構築する。そのドメインモデルはある評価タスクにおいて専門家が使用する概念と語彙を表現したものである。またサンプルレポートによってデザインの記述、与えられる環境、その環境に関しての詳細なデザインの評価、及びデザインフローが発見された時にそのデザインを

改善する提案に関する知識がレポート形式で獲得される。図2にhardeningドメインにおける初期ドメインモデルを示す。

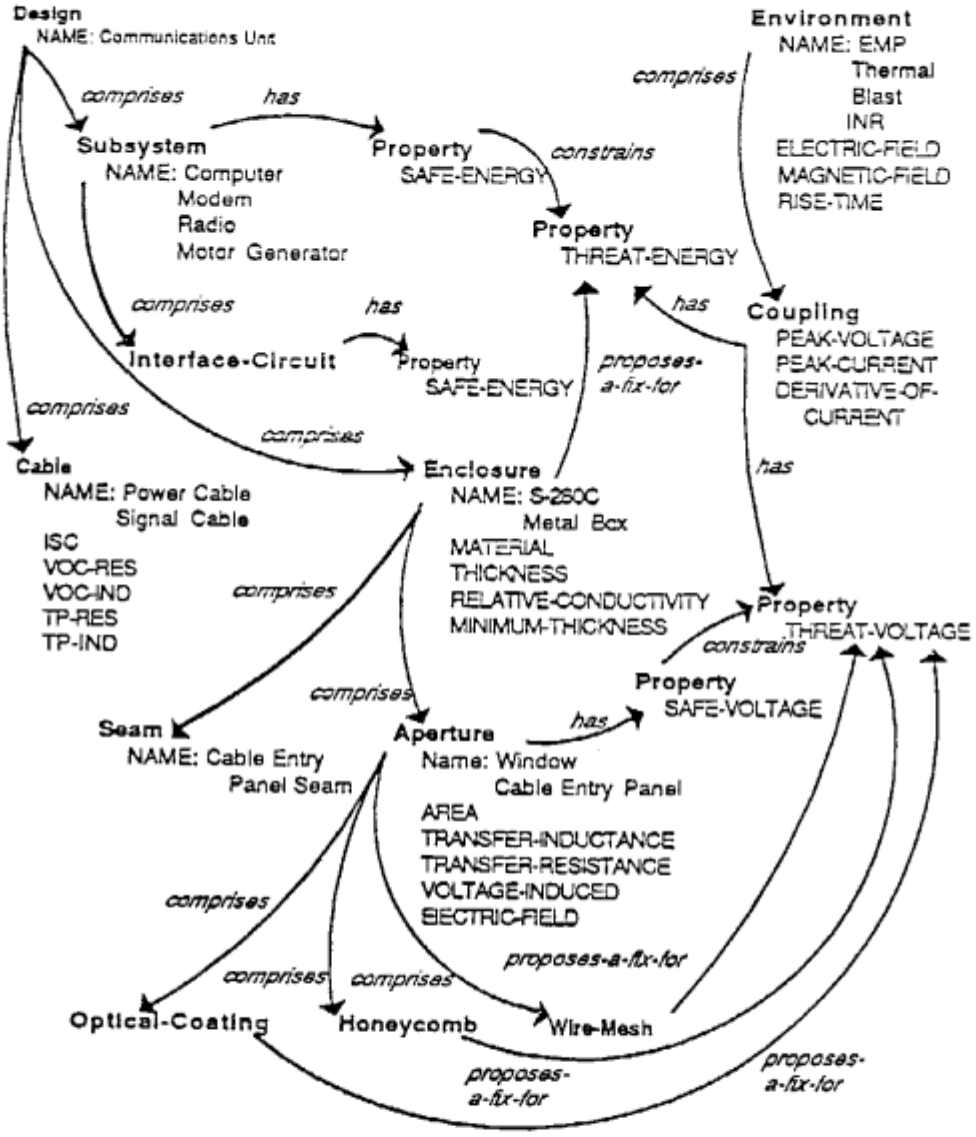


図2　hardeningドメインにおける初期ドメインモデル

ドメインモデルが定義されると、KNACKはヒューリスティックを使用して専門家とのインタビューにより初期ドメインモデルで使用される概念・語彙の詳細化と機能的な知識の獲得を行う。ここで機能的な知識とは、ある特定のアプリケーションについて評価タスクを処理するために必要な情報を獲得するための知識である。具体的には、専門家が評価の際に用いるデザインと環境のパラメータの獲得方法・伝播方法・比較方法に関する知識のことである。KNACKは専門家が提供する必要のある情報の量を最小にするためのヒューリスティックと、先に獲得した知識から獲得すべき知識や付随的な知識を推論するヒューリスティックを使用することによって効率的に知識獲得を行う。図3にリファインの行われたhardeningドメインにおけるドメインモデルを示す。

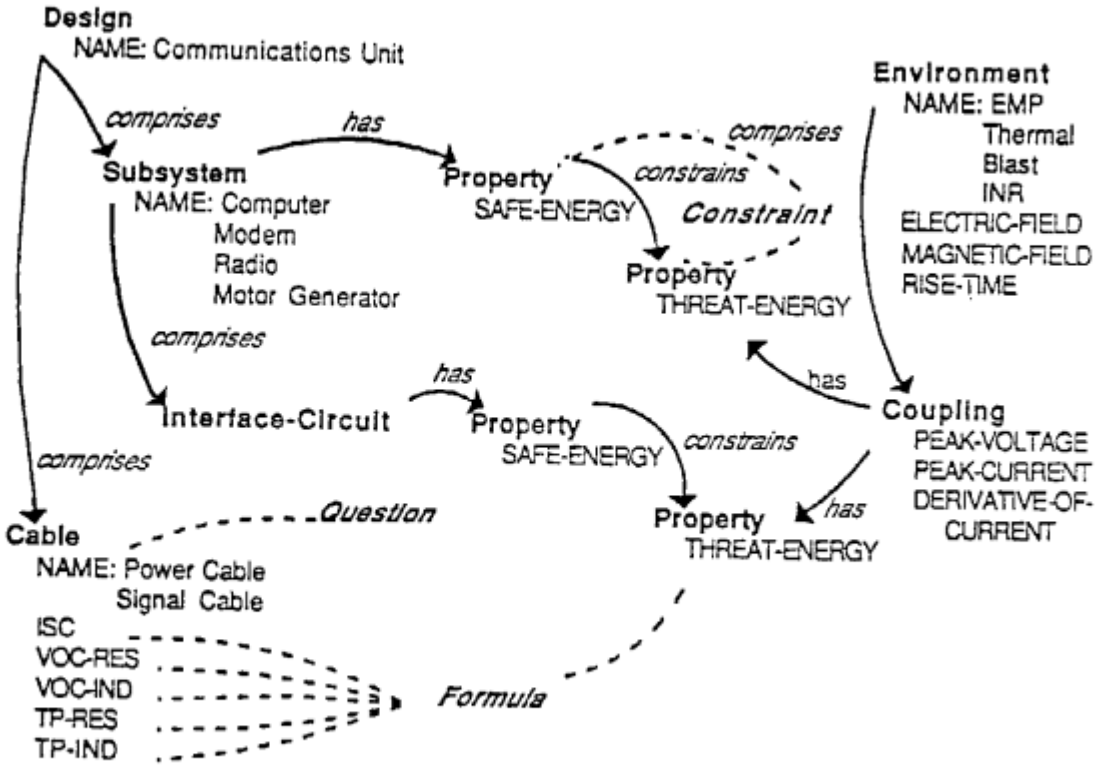


図3　hardeningドメインにおけるドメインモデル

KNACKの特徴はナレッジエンジアが介在すること無しに専門家からインタビューとレポートによる対話によって、ドメインモデルの獲得、あるアプリケーション用のドメインモデルの詳細化を行うことである。KNACKのパフォーマンスの限界はKNACKの持つ評価問題における一般的なドメインモデルと、そのドメインモデルをある特定のアプリケーション用のドメインモデルへとリファインする能力に依存する。

「参考文献」

[Klinker 86] George Klinker, Joel Bentolia, Serge Genetet, Michael Grimes, and John McDermott: KNACK-Report-Driven Knowledge Acquisition , Proceedings of the KNOWLEDGE AQUISITION FOR KNOWLEDGE-BASED SYSTEMS WORKSHOP, 23-0, 1986.
[Klinker 87] George Klinker, Serge Genetet, and John McDermott: Knowledge Acquisition for Evaluation Systems, Proceedings of the 2ND KNOWLEDGE AQUISITION FOR KNOWLEDGE-BASED SYSTEMS WORKSHOP, 13-0, 1987.

6.3 OPAL

OPAL[Musen 86]は癌治療計画を選定するエキスパートシステムであるONCOCIN用の知識ベース構築とメンテナンスを行うための知識エディタである。従来のエキスパートシェルはフレームやルールといった最終的なインプリメトレベルの知識表現しか提供していない。このため知識ベース開発・メンテナンスは、専門家自身が直接知識ベースの構築・修正を行うことができないために、ナレッジエンジニアの手によるしかなく、非常な労力と時間がかかった。そこでナレッジエンジニアにできるだけ依存せず専門家自身がONCOCIN用の知識ベースの構築ができることを目的としてOPALが開発された訳である。

OPALのアプローチとしては知識ベースの中味を専門家が直接入力する作業を支援する知識エディタを開発するというものである。そのためOPALは問題に特殊化したドメインモデルを使用している。このドメインモデルを使用することによって専門医師の理解に近い表形式の表現や視覚的な表現を用いて知識獲得を行う。

OPALのドメインモデルは対象問題のタスク(癌患者に対する治療計画)の解析に基づいて作られており、宣言的な知識、手続き的知識、制御知識を表現する枠組みが用意されている。宣言的知識はENTITYとRELATIONと呼ばれる表現を用いてフレームとして表現される。手続き的な知識はDOMAIN ACTIONと呼ばれる表現を用いてプロダクションルールとして表現される。そしてメタ知識の表現はDOMAIN PREDICATEとPROCEDUAL KNOWLEDGEと呼ばれる表現を用いて、それぞれプロダクションルールとgenerator(時間と共に変化する治療計画を表現するための特殊なオブジェクト)として表現される。そしてドメインモデルの構成要素は表・ダイアグラムによって抽出・表現されることになる。

OPALのコンサルテーションではまずOPALの提供する表によってENTITYの抽出が行われる。抽出されたENTITYは階層構造を作り、各ENTITYに対してその内部に含まれるパラメータの値を決定する知識が表によって抽出され、プロダクションルールが作成される。図1にENTITYの抽出例を、図2にプロダクションルールの抽出例を示す。

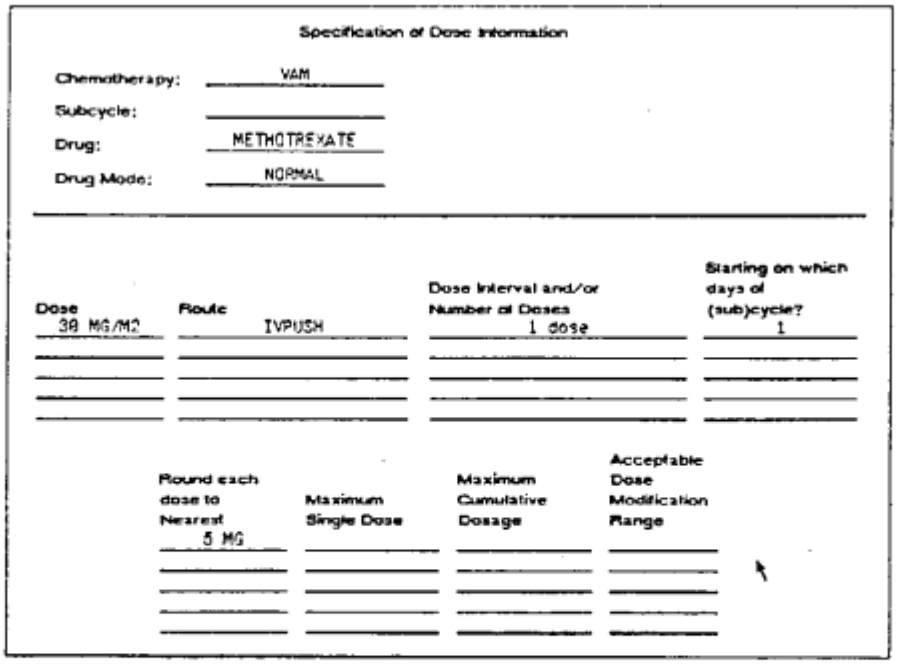


図1 ENTITYの抽出例

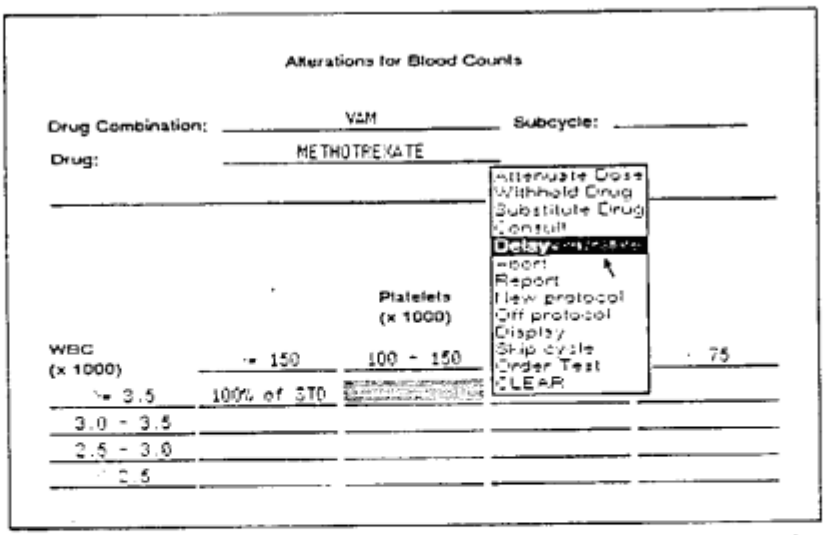


図2 プロダクションルールの抽出例

そして個々の治療時間の順序を決定する知識がグラフィックエディタによってダイアグラム形式で抽出される。獲得された知識はOPALの提供する中間表現に自動的に変換され、中間表現をONCOCINの知識表現形式に変換し、既存知識ベースとのマージが行われることによって知識獲得が終了する。OPALによる知識獲得イメージを図3に示す。

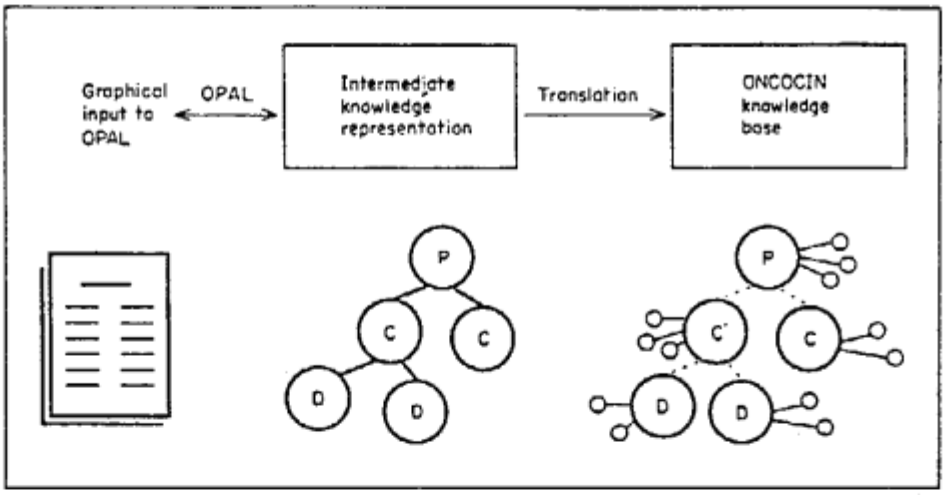


図3 OPALの知識獲得イメージ

OPALの特徴としては対象問題に特殊化したドメインモデルとグラフィックエディタを用いることによって、専門家自身による知識構築を容易にしたことである。このためONCOCIN用の新しい知識を獲得するに要する時間が大幅に短縮された。OPALは従来の知識獲得支援システム(TEIRESIAS、INKA、ROGET、ETS等)と比較して対象問題に関する膨大な知識を持っており、対象問題の知識獲得が収集しやすいようにモデルが作られている点が異なる。その反面、ドメインに特殊化しぎているため他の問題への応用は難しいと考えられる。

［参考文献］

[Musen 86] Mark A. Musen, Lawrence M. Fagan, David M. Combs, and Edward H.Shortliffe : Using A Domain Model to Drive an Intreactive Knowledge Editing Tool, Proceedings of KNOWLEDGE AQUISITION FOR KNOWLEDGE-BASED SYSTEMS WORKSHOP, 33-0, 1986.

6．4　　KRITON：a knowledge-acquisition tool for expert systems

6．4．1　概　要

種々の獲得法に基づいて人間の持つ異なった種類の知識を統合的に獲得できるような環境が知識獲得ツールには必要である。ＫＲＩＴＯＮはそのような設計思想を元にして作られたハイブリッド型の知識獲得システムである。ＫＲＩＴＯＮでは、知識源として専門家の持つ手続き的知識と宣言的知識、及び静的な文書知識を設定しており、抽出方法としてプロトコル分析、インタビュー、テキスト分析を用意している。抽出したそれぞれの知識は分析・統合されて、中間知識表現の形にまとめられた後、システムの表現形式（フレーム・ルール・コンストレイント）へと変換される。

6．4．2　知識抽出

（1）インタビュー
ユーザとのインタラクションにより、対象領域の概念項目が階層関係と属性を持つ"構造化オブジェクト"として抽出され、中間知識表現へ送られる。インタビュー技法としては、トップレベルにレパートリーグリッドを、それの切り替えモードとしてラダリングを用意している。また、シミュレーション可能な状況下で問題解決のステップを聞く、フォワードシナリオシミュレーションも別途用意されている。
（2）プロトコル解析
問題解決のための手続的な知識を獲得する。専門家の、問題解決に関する発話部分をテープに録音したもの（プロトコル）を元情報として用い、命題となる部分を抽出する。その手順は、プロトコルのセグメント化、各セグメントに対する命題作成、選択したオペレータ、引数の正当性のチェック、変数の具体化、アレンジから成る。
（3）テキスト解析
マニュアルやドキュメントのテキストを対象としてインクリメンタルな内容分析を行い、ＫＥ自身の知識獲得作業を軽減する。キーワードの頻度情報から必要と思われるキーワードを選択し、それを含む前後のテキストを抜き出した後、プロトコル解析と同様の手法で解析を進める。

6．4．3　中間知識表現と知識統合手法

（1）中間知識表現
中間レベルでの知識表現が必要である理由として以下のような項目が上げられている。知識の拡張可能性、異なるシェルへの適用可能性、獲得中の不完全知識の格納、知識源寄りの情報のメンテナンス、既獲得知識ベースの統合・利用。
インタビュー及びテキスト解析の結果は、中間知識表現のレベルでは意味ネットにより表現される。意味ネットには基本概念に対する、階層関係・素性・項目間関係が記述される。プロトコル解析の結果は命題算法（propositional calculus）により表現される。これは、概念間の基本的関係を sematnic primitive を使って記述したもので、問題解決プロセスにおける、オブジェクトの変換パスを表現するものである。
（2）既獲得知識の利用
Aquisition Knowlege Base に蓄えられている獲得済みの知識は、新たな獲得過程においてガイドの役割を果たすとともに、ドメインモデルの完全化や解釈モデルとして他の場面で利用される。
（3）不完全な知識の取り扱い
中間知識表現の欠落要素を監視するための機構が用意されており、インヘリタンスパス、part-of relation，instance relation等のリンク情報が欠落したオブジェクトがないかをチェックし、欠落要素が存在すれば情報抽出のためのメソッドの利用をユーザに薦めるかインタビューレベルへの切り替えを行う。
（4）知識ベースの生成
・フレームジェネレータ　単純なシンタックストランスフォーメション
・ルールジェネレータ　ルールの対象となる、プロトコル分析結果の命題クローズのセットから必要な命題を選択して、ルールの条件・アクション部を作成する。
・コンストレイントジェネレータ　グローバル変数に対する制限を記述する。

6．4．4　処理ステップ

ＫＲＩＴＯＮの知識獲得過程の処理ステップを以下に記す。

(1) ドメインの定義　・・・　インタビューによる。
(2) 宣言的知識の抽出　・・・　インタビュー、テキスト分析により重要な用語や概念が聞き出される。意味ネットが十分な大きさに達するまで行われる。
(3) プロトコルの記録　・・・　問題解決時の、専門家の発話内容を録音する。一定の言葉で表現させるための注意深いガイダンスが用意されており単一の問題解決パスに絞って行う。
(4) プロトコルの書き写し
(5) プロトコルのセグメント化　・・・　番号を割り振りながら、スピーチポーズを単位として、書き写したプロトコルをセグメントに自動分割する。
(6) 分割プロトコル上の知識要素(knowldge element)検索
(7) セグメント化されたプロトコルの意味分析　・・・
セグメント上の単語間に次の関係がないかどうかを調べる。
・順序関係 (A is smaller than B, A is equal to B etc.)
・傾向 (The state of X is stable, The value of Y continues to increase. etc)
(8) 命題化 (Propositionalization)　・・・　知識要素が命題化される。
(9) 命題の完全化　・・・　自分自身に近いセグメント順に、(8)で作った命題を完全化するような知識要素を見付けようとする。
(10) 知識ベースマッチング　・・・　既に抽出された命題を参照して、引数の欠落がないかどうか確認する。
(11) 中間知識の表現　・・・　プロトコル分析の結果の全命題を中間表現言語の形式にまとめる。その結果以下の要素を持つ中間表現による命題が生成される。
・オペレーター　　概念間の基本的関係を表す意味的深層格
・セグメントマーカ　　自然言語のオリジナルな命題へのポインタ
・関係する概念項目
(12) 構造化オブジェクトの完全性チェック
(13) 知識ベースへの変換

6．4．5　インプリメンテーション

開発はＸＥＲＯＸ－１１０８上でＩＮＴＥＲＬＩＳＰ－Ｄ、ＬＯＯＰＳを使用して行われている。中間知識表現における構造化オブジェクトの表現にＬＯＯＰＳのオブジェクトを使っている。プロトコル分析及びテキスト解析は閉じた語いと主題化（lemmatization）を利用した、語いベースの解析手法を取っている。知識ベースにおけるフレーム表現には既存のフレーム表現言語が、ルール表現には中間知識で得た命題の組み合わが、制約表現にはある種の制約表現言語がそれぞれ使われている。

アプリケーションの展開としては事務設備の設計計画エキスパートシステムが試作されている。

6．5．Ontological Analysis

エキスパートシステム構築は複雑で労力を要する活動である。特に、適切な抽象概念を選択し、対象を分類・整理するのが困難である。このような知識工学の過程を支援する手法として、知識レベルの分析が提唱された[Newell82]。その考えを受けて提唱されたのが、問題空間の知識レベル分析を行なうための"存在論分析"(ontological analysis)と呼ばれる方法論である[Alex86]。

6．5．1．存在論分析の概要

存在論（ontology）とは、抽象的・具体的な実体、およびあるタスクを遂行するのに必要な物理的・認知的な実体を表現する関係・変換の集りである。６章は獲得ツールの調査が中心になっているが、本節で述べるAlexander らによる存在論分析は、方法論＋言語を提供するものである。まず、言語について説明した後、方法論について述べる。

存在論を構築するための言語として、Alexander らは、 SPOONS (SPecification Of ONtological Structure)と呼ばれる一群の言語ファミリを提供している。SPOONSファミリのうち最も簡潔で有用なのが、SUPE-SPOONS (SUPErstructure SPOONS) である。この言語は、a)表示的意味論 (denotational semantics) におけるドメイン方程式、および、b)代数的仕様記述 (algebraic specification)、に基礎を置いている。SUPE-SPOONS には、a)ドメイン方程式、および、b)ドメイン関数定義、の２種類のステートメントがある。

ドメイン方程式の演算子としては、次の５種類がある。

(1) 構造化和集合：Ｄ＋Ｅ

ドメインＤとＥの構造化和集合(discriminated union) とは、ＤとＥの各要素から構成されるドメイン（集合）で、もともとＤとＥのどちらに属していたかを識別できるもの。データモデルにおける汎化 (generalization) に相当する。

(2) 直積：Ｄ×Ｅ

ドメインＤとＥの直積(cross product) とは、第１要素がＤの要素、第２要素がＥの要素になっている全ての順序対から構成されるドメインである。これは、データモデルにおける統合(aggregation) に相当する。

(3) 写像：D->E

ＤからＥの上への全ての関数からなる集合。

(4) 巾集合：２**D

Ｄの全ての部分集合の集合。

(5) 閉包：D*

Ｄの要素からなる全てのシーケンスの集合。

Alexander らの方法論によると、存在論は、静的存在論、動的存在論、認識的存在論の順序で構築される。以下で、各段階の分析を説明する。

(1) 静的存在論 (static ontology)

静的存在論は、物理的実体や問題空間を構成するプリミティブな実体、各実体の属性、実体間の関係などを定義するものである。存在論分析は、実体を列挙し、それらに固有の属性・関連を明確化する静的存在論の分析が第１段階である。静的存在論のレベルの分析は、Chenの実体－関連モデルによるデータモデルの構築に似ている。

(2) 動的存在論 (dynamic ontology)

動的存在論は、a)対象分野の問題空間、および、b)問題空間内の状態遷移をもたらすアクション、を定義するものである。動的存在論の分析は、まず静的存在論分析で得られた個々の要素がどのように組合わさっているかを明確にし、それから問題空間の状態遷移をもたらすオペレータを定義するという手順を取る。オペレータは、問題空間を構成する状態の集合（ドメイン）上の変換として定義される。動的存在論の分析は、問題解決過程を通して変化しない知識と問題解決が進につれて変化していく知識を区別するのが目的である。

(3) 認識的存在論 (epistemic ontology)

静的および動的存在論分析で明らかになった知識が、問題解決においてどのように利用されるかを制御する制約条件および推論方式を明確にするものである。認識的存在論は、a)実行すべきオペレータを選択するもの、および、b)あるオペレータの実行を制御するもの、の２種類の知識構造を持っている。

以下に、存在論分析を知的電子予定表の開発に適用した際の、SUPE-SPOONS による記述の一部を各分析ごとに掲げる（[Alex86][Alex87]に完全な記述あり）。

(1) 静的存在論分析

```
Meeting - <atomic>
Person  - <atomic>
Scheduled__Meeting - ( Meeting × Person )
```

(2) 動的存在論分析

```
create__new __meeting
        - ( Purpose ×  Required__Participation ) -> Meeting
```

(3) 認識的存在論分析

```
Arbitrator__Selection __Rules
        - 2 ** ( Purpose  ×  Person__Description )
```

6．5．2．存在論分析の実践

Alexander は、存在論分析をオシロスコープのトラブルシューティングを行なうエキスパート・システムの構築[Freiling86]などに応用して、存在論分析を実践する上で役に立つノウハウを得ている[Alex87]。

(1) マニュアルなどにある専門用語は、静的存在論分析の一次情報源になる。

(2) 実体を原子的なものと導出しうるものに区別することは、実体間の関係を明確にする上で役に立つ。特に、内容の把握しにくい実体は、原子的なものとして表現しておくことによって、たとえその内容について何もわからなくても、存在を認識しておくことができ、エキスパートシステムの開発を容易にする。

(3) 内延的表現と外延的表現を明確に区別することにより、実体の外延的表現に対して構造の持たせ方を最少限にとどめることができる。

(4) 抽象化の大部分は般化と統合によって行なえる。

(5) 存在論分析により知識構造を構築すると、下位レベルの構造の矛盾／無矛盾が明らかになる。つまり、意味論やモデル論の研究者がよく使う構成的な分析を知識工学者でも利用できるようになった。

(6) ドメイン方程式を多重に定義している場合は、それなりの情報が表現されている。例えば、同一の実体を構文、動作、意味、抽象度といった異なった視点から見ている場合である。

また、存在論分析の欠点として次のものが認識された。

(1) 存在論分析においては、ドメイン方程式で知識構造における"型"を定義するが、型は、ドメイン方程式で定義するような値ではなく、記述に一貫性がない。

(2) 型の階層を調べることができない。型の階層は、現時点でどのような抽象化がなされているかという知識レベル分析の環境を定義しているので、どのような抽象化がなされているかを調べたり、現時点の抽象化を修正できる必要がある。

(3) 操作的な要素を抽出することができない。

型の定義を完全にするには、その型に対する操作を定義する必要がある。現在のSUPE-SPOONでは、実体の構造を定義することはできるが、その振舞いを定義することはできない。

(4) 継承関係を明確にすることができない。

通常、下位概念を明確にして、操作、構造などを継承できるようにするが、SUPE-SPOON では継承関係を明確にすることができない。

(5) 記述に厳密さが要求される。

知識レベル分析を行なっている過程では、矛盾した型を持ってしまう場合がある。表現上、そのような矛盾が許される必要がある。

6．5．3．次世代の存在論分析用言語

知識レベル分析の究極の目的は、対象領域の適当な抽象概念を明確にすることである。知識レベル分析用言語のプロトタイプともいえる SUPE-SPOON によって、その目的にかなった次期言語を開発するさいの方向付けが得られた[Alex87]。

(1) 計算機支援の必要性

抽象概念の形式的理論による方法論と計算機支援を組合わせることが必要である。

(2) 変更・修正の管理の支援

抽象化のプロセスには試行錯誤がつきものなので、抽象概念の変更・修正、バージョン管理、既存の抽象構造の分析などをシステムで支援する必要がある。

(3) 分析段階からインプリメンテーションへの移行の支援

分析段階で得られたタイプ階層を最終的なインプリメンテーションに用いる知識表現言語による記述への変換の支援が必要である。

6．5．4．まとめ

存在論分析は、記述言語の表現力や理論的基礎、支援ツールの欠如など、またまだ問題は多く、実用への道は遠い。しかし、これらの問題が解決されれば、知識の整理・分類を知識レベルで行なうための方法論として、中心的な位置を占めるようになると思われる。

[参考文献]

[Newell82] A.Newell : "The Knowledge Level", Artificial Intelligence, Vol.18, pp.87-127 (1982).

[Freiling86] J.Freiling, et al. : "The Ontological Structure of a Trouble-shooting System for Electronic Instruments", First Int. Conf. on Applications of AI to Engineering Problems, Southampton University, U.K., April (1986).

[Alex86] J.Alexander, et al. : "Knowledge Level Engineering: Ontological Analysis", Proc. of AAAI'86, pp.963-968 (1986).

[Alex87] J.Alexander, et al. : "Ontological Analysis: An Ongoing Analysis", Int. J. of Man-Machine Studies, Vol.26, No.4, pp.473-486 (1987).

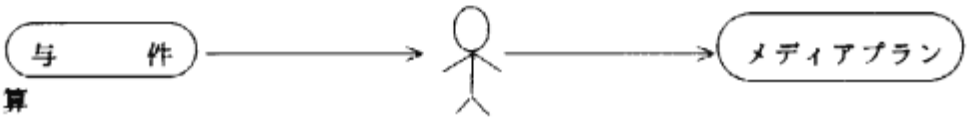
6.6 A Knowledge-based Media Planning System

（1）概要

筆者は知識ベースに基ずいた「メディアプランニング」を行うエキスパートシステムを開発中である。　その際に使われている知識獲得手法が本論文中で論じられている。

メディアプランニングエキスパートシステムを開発する過程で，先ず手初めに「ディシジョンフレーム」と呼ばれるシステムが作られた。　ディシジョンフレームは，エキスパートのメディアプラン作成をサポートするシステムであり，これを利用すれば素人にもメディアプランが作れるという訳ではない（従ってエキスパートシステムとは呼んでいない）。　現在，ディシジョンフレームに専門知識を追加して，エキスパートシステムとしても働くシステムの開発を行っているが，メディアプランナーから専門知識を抽出するために，後述の４つの手法が使われている。

（2）メディアプランニングとは



・予算
・目標購買者層
・マーケットのシェア
・競合製品の情報
・メディアの情報　etc.

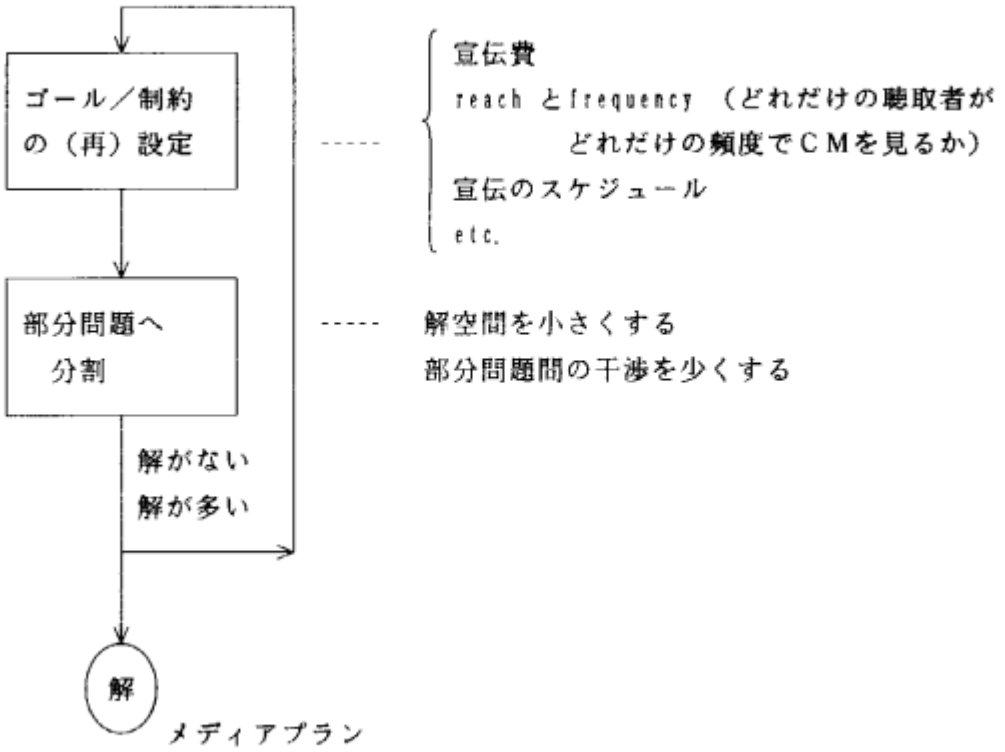
メディアプランナー
（広告代理店）

商品の宣伝計画
・予算のメディア間の配分
・ＣＭの時間帯，期間
・期間での予算の配分etc.

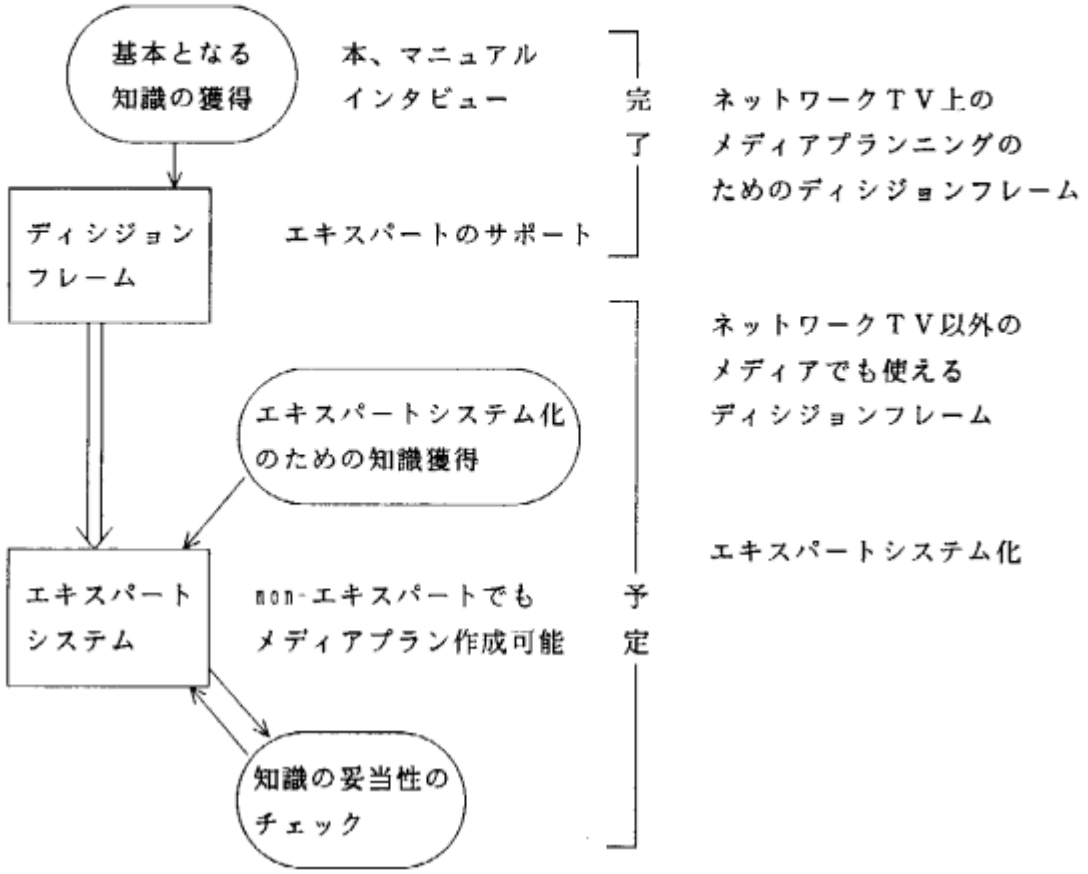
＜メディアプランニングの問題の特徴＞

- ill-structuredである。
- 考慮しなければならないデータが多い。
- 解空間が膨大である。
- 解の質の評価が難しい。

（3）メディアプランナーの作業



（4）　システムの成長過程と知識獲得



（5）知識獲得法

現在のシステムを成長させていくに当って、次の４つの方法で知識の獲得を行っている。

① Elicitation procedure

エキスパートにそのドメインで重要な概念をいくつか提示し、それから想起されることに関しての説明を求めることによって、宣言的な知識を得る。この方法はspreading activation modelに基づいている。

この方法により、メディアプランの与件とメディアプランナーの思考がどう結びついているのか，特定のメディアプラン作成問題に関して特別の知識が関与しているのかを明らかにする。

② Problem sorting

エキスパートに与件の異なる問題を幾つも与え，それから作られるメディアプランの類似性にもとずいて，問題をグループ分けしてもらう。 この方法によって，メディアプランナーがまず問題をカテゴライズしてから問題に対処していることを確かめると共に，与件と中間の目標と作成されるメディアプランの間の関連を理解する。

Problem-sorting による知識獲得においては，heuristic classification procedure を使うこともできる。

③ Protocol analysis

メディアプラン作成の間，声に出して考えてもらう。 これにより，以下４つのタイプの問題解決がメディアプラン作成の過程で行われている事を理解する。

（1）ゴールや制約をどう設定するか。

（2）ゴールをメディアプラン作成にどう使っていくか。

（3）ゴールや制約が満たされないとき，メディアプランをどう調整するか。

（4）メディアプランをどう評価するか。

④ ディシジョンフレームの使用

ディシジョンフレームを使ったメディアプランの作成を行ってもらう。 ディシジョンフレームはエキスパートが下した決断と，決断が下されたシーケンスを記録しておけるようになっているので，それと，protocol analysis の結果とを比較する。

[参考文献]

Mitchell:The use of alternative knowledge-acquisition in the development of a knowledge-based media planning system.
International Journal of Man-Machine Studies, vol 26, Number 4, Academic Press, April 1987.

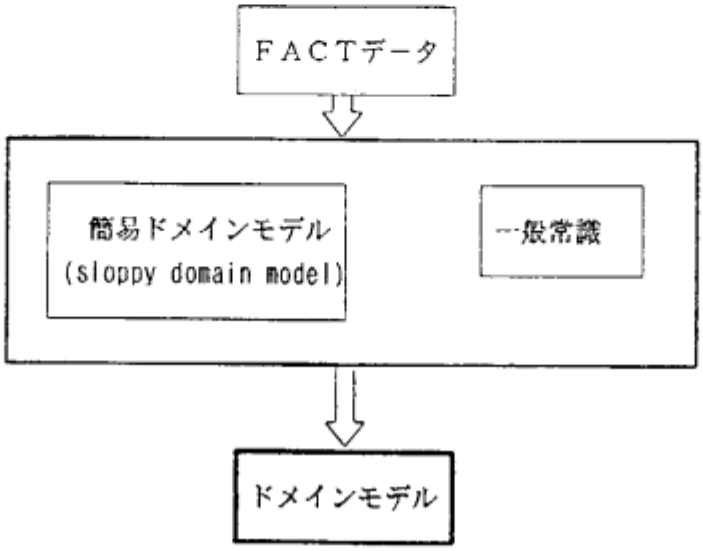
6．7．ＢＬＩＰ

ＢＬＩＰの位置付け：

ＢＬＩＰは、ＬＥＲＮＥＲと呼ばれるシステムの知識獲得を担当する部分である。獲得された知識は、ＴＷＡＩＣＥと呼ばれるシェル上で動くフォーマットに変換される。

ＢＬＩＰの機能概要：

ＢＬＩＰは、予めシステムに組み込まれた「一般常識」とユーザーがインプットした「簡易ドメインモデル」に基づいて、ＦＡＣＴデータから『ドメインモデル』の構築を行う。



一般常識（どのような知識についても影響されない）：

『三段論的推論法』などの論理的推論方法

簡易ドメインモデル，ＦＡＣＴ（対象知識によって影響される）

（簡易ドメインモデル例）

```
_CURE_1( ( drug ), ( illness ) ):この薬はこの病気に効く.
_CURE_2( ( substance ), ( symptom ):この物質はこの兆候のときに効く.
_CURE_3( ( drug ), ( symptom ) ):この薬はこの兆候のときに効く.
_CURE_4( ( substance ), ( illness ) ):この物質はこの病気のときに効く.
_CONTAIN( ( drug ), ( substance ) ):この薬はこの物質を含む.
_STRENGTHERN_1( ( drug ), ( illness ) ):薬が引き起こす病気.
_STRENGTHERN_2( ( substance ), ( symptom ) ):物質が引き起こす病気.
_STRENGTHERN_3( ( drug ), ( symptom ) ):薬が引き起こす兆候.
_mono-substance( ( drug ) ):この薬は単体である.
_multi-substance( ( drug ) ):この薬は複合体である.
_ADVANTAGE( ( patient ), ( drug ) ):この患者には、この薬の投与は利益有.
_IS_GOOD_FOR( ( patient ), ( drug ) ):この患者には、この薬の投与がよい.
_DISADVANTAGE( ( patient ), ( drug ) ):この患者には、この薬の投与は不利益.
```

（ＦＡＣＴ例）

```
- CONTAIN (aetz, paracetamol)
- CONTAIN (aetz, ass)
- CONTAIN (aspirin, ass)
- STRENGTHEN_2 (ass, stomach-ache)
- STRENGTHEN_2 (paracetamol, kidney-trouble)
- STRENGTHEN_3 (aetz, kidney-trouble)
- STRENGTHEN_3 (aspirin, stomach-ache)
```

（副作用のない薬物を選定する薬剤師の知識モデル構築の場合）

ドメインモデルの構築

1．仮説をつくる。

meta-meta-factとmeta-factからから別のmeta-factを導出するルールを作る。

(大前提をmeta-meta-factとし、小前提をmeta-factとし、

別のmeta-factを結論と考えれば良い。)

(i)meta-meta-fact: OPPOSITE_2(opposite_1, inclusive_1)

(ii)meta-fact : OPPOSITE_1(mono-substance, multi-substance)

(iii)meta-fact : INCLUSIVE_1(mono-substance, multi-substance)

に対して、

meta-rule:

OPPOSITE_1(mono-substance, multi-substance)

⇒ not INCLUSIVE_1(mono-substance, multi-substance)

2．仮説を検証し、評価する（0 ~ 1000の数値を付与する）。

仮説の検証の手順:

①推論エンジン中のfactを検索

②仮説のもつ*CHACTERISTIC SITUATION とマッチングが取れるか否かを調べる

③マッチングが取れたfact数、取れなかったfact数を調べる

総てマッチングが取れた場合及びマッチングが取れた数がthresholdを超した場合:

仮説はverify され、

meta-fact ⇒ 推論エンジンのfact 部分

rule ⇒ 推論エンジンのrule 部分

に格納される。

総てマッチングが取れなかった場合及びマッチングしなかった数がthresholdを超した場合:

仮説は棄却される。

マッチングしなかった数、した数ともにthresholdを超さなかった場合:

推論エンジンには何も格納されないがactivenessの得点がふえる。

*

CHARACTERISTIC SITUATION:

factに関する知識を検索するパターン。あるpredicateに対してあるmeta-predicateがあてはまるかどうかの有効性の確認の際に用いられる。あるmeta-predicateに対して、positeive situation か negative situationかがRULE SCHEMAから自動的に生成され格納されている。

6.8 DESIGN FOR ACQUISITION

(1) 知識獲得容易化設計の原理

知識獲得容易化設計(DESIGN FOR ACQUISITION)は，文字通り知識を獲得しやすい知識システムを設計する上での考え方であり，米国マサチュセッツ大学のGruber，Cohenによって提案された。その根本となるのは，次の三つの設計の原理である。

原理1：エキスパートが定義した重要な領域概念を把えるため，タスクレベルの表現プリミティブを設計せよ

ナレッジエンジニアは，タスクレベルの表現手段をエキスパートに提供すべきである。これにより，エキスパートの知識の整理の仕方とインプリメンテーションでの知識の表わし方の間の「表現のミスマッチ」を回避することができる。タスクレベルのプリミティブの例として，「トリガー」がある。これは，データが見つかったときに仮説が即活性化されるというようなデータと仮説の関係を表わすプリミティブである。

原理2：陽的で宣言的な表現プリミティブを設計せよ

知識獲得の点からは，宣言的表現は手続き的表現に勝る。例えば，制御の知識のように手続きで書くことが自然な場合であっても，宣言的に定式化すれば，知識の獲得，説明，保守がしやすい。また，プリミティブの目的や定義を陽の形に書いておくことは，複数のエキスパートやプログラマと共同作業をする上で重要である。

原理3：エキスパートの知識と同じ一般化のレベルで表現を設計せよ

必要がある場合以外は，エキスパートに一般化をさせようとしてはならない。例えば，エキスパートが，血圧を心臓収縮／弛緩の比について広いレンジの値で考えるのに，ナレッジエンジニアが，「高い」「低い」「正常」というように少数のカテゴリにまとめることを強いるケースがある。

エキスパートに知っている以上のことを言うよう求めてはならない。例えば，知識で正しいと判断できる範囲よりもずっと高い精度で，ナレッジエンジニアが確信の度合をきくようなケースがある。

(2) 知識獲得容易化設計のケーススタディ

医療診断システムMUMにおいて，獲得が難しい二種類の知識についてケーススタディを試みた。MUMは，精密検査を通じて患者の胸部・腹部の痛みが今後どうなるかを診断するために，根拠を最良の順序で集めて不確かさを管理するタスクを行なう。このとき，根拠と病名の間のヒューリスティックな関連についての知識の他に，根拠の組合せの知識や推論の制御の知識を用いる。

(a) 組合せの知識の獲得容易化設計

組合せの知識とは，一つの結論を支持するために，いくつかの根拠の断片をどう組み合わせるかを表わした知識である。不確かさを表わすのに，ナレッジエンジニアはエキスパートの言葉ではなく，数値関数を組み込んでしまうことが多い。エキスパートにとってわかりにくくなるので，知識の獲得が難しくなる。このような点を考慮して，MUMでは，次のような方式を用いている。

(i)不確かさを数値ではなく，「どこまで確かか」について領域の言葉として意味がある記号で表現している。それらは，confirmed，disconfirmed，supported，detracted，strongly-supported，strongly-detracted，unknownの7種類である。

(ii)根拠のクラスターをフレームで陽の形に表現している。
(iii)根拠の結合は，グローバルな数値関数ではなく，クラスター毎にエキスパートが与えたローカルな結合関数を用いる。

この方式は，新規ではないが，設計原理１～３を反映したものである。知識ベースの保守や改良に際しても，表現の組合せを疎らにすることかできるので組合せ的問題とならない。したがって，知識獲得のプロセスを効率化できる。

(b) 制御の知識の獲得容易化設計

ＭＵＭが扱う「将来の予測を含む診断」では，次に挙げるようなステップで制御の知識が重要となる。

(イ)適切な順序でデータを収集
(ロ)不要なテストを省略
(ハ)関連する仮説に適したものだけ質問
(ニ)完全に病名がわかる前に予備診断的処置を処法

このため，推論の制御の知識は専門知識の大半を占めることになる。

エキスパートに制御のことで煩わせることなく領域個有の知識を獲得できるよう，ＭＵＭでは，まず，制御の決定に影響を及ぼす領域のパラメータをきいて，次に，そのパラメータで制御の知識を定式化してもらう方式を用いている。「ある病気は別の病気よりも危ない」，「ある臨床試験は費用がかかる」のように，タスクレベルの言葉を使った方が「アジェンダでのタスクの優先度」や「ルールでの節の並べ方」のようなインプリメンテーションのレベルの言葉を使った場合より，知識を獲得しやすくなる。この方式も，設計原理１～３を反映したものである。ローカルに制御を決定することになるため，獲得が容易となるだけでなく，入力の不足もわかりやすくなる。

(3) 知識獲得支援のためのＭＵアーキテクチャ

エキスパートシステムＭＵＭを一般化し，ＭＵアーキテクチャにまとめている。ＭＵの知識ベースの構造を図１に示した。ここで，推論のネット中にあるオブジェクト(ブロックのノードで図示)は，記号の値を伝幡させる推論関係(アークで図示)によって結合されている。例えば，potential-evidence関係は，supportedやconfirmedのような「どこまで確かか」という確信の状態を伝えている。各ノードで，ローカルな結合関数は，potential-evidenceに寄与を持つノードの確信の状態の関数として，そのノードの確信の状態を決定する。制御の知識は，ネットのオブジェクトの状態や，検査・処置の操作の性質が与えられたときに，どのクラスターに注目すべきかというフォーカスを決めるため，あるいは検査を処法するというような可能な操作を選択するために用いられる。

ＭＵアーキテクチャをサポートするツールの階層を，表１に整理してある。

[参考文献]

[1]T.R.Gruber, P.R.Cohen : Design for Acquisition:Principles of Knowledge-System Design to Facilitate Knowledge Acquisition, Int. J. of Man-Machine Studies, Vol.26, No.2, pp.231-243 (1987).

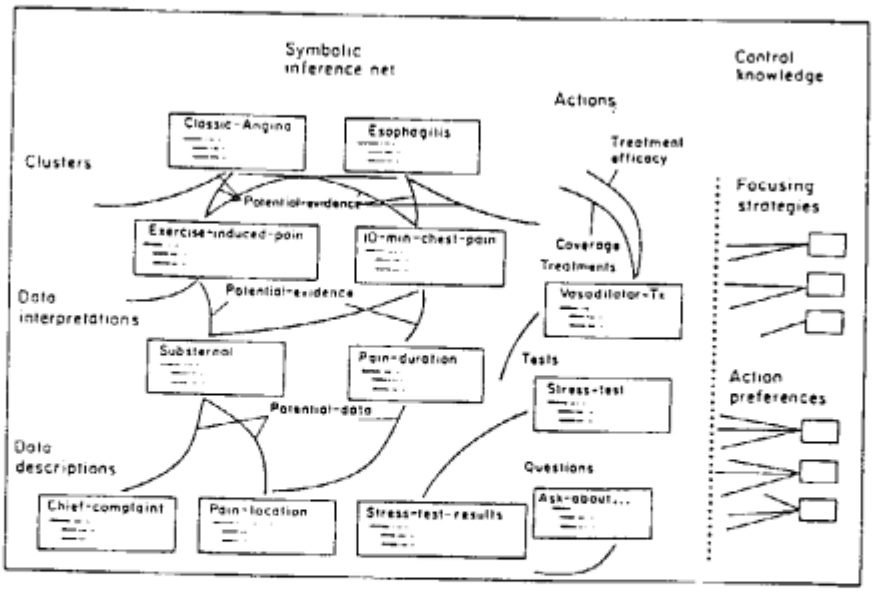


図1　MUの知識ベースの構造

表1　MUアーキテクチャをサポートするツールの階層

Tool	Objects in user's view	Software support
Knowledge acquisition interface	*Application-specific terms* Diseases, tests, treatments, questions, intermediate diagnoses, criticality of diseases, cost of tests, efficacy of treatment	*(Meta-)knowledge-based utilities* Language-specific editors and form-filling interfaces, inferential consistency analyser, graphical display for objects and relations
Virtual machine (shell)	*Task-level constructs* Clusters and combining functions Control parameters Control rules Preference rankings	*Task-specific reasoning mechanisms* Value propagation functions Interface to the inference net Rule-based planner Decision-making support
AI toolbox (KEE)	*Implementation primitives* Rules, pattern matching language frames and slots Lisp objects and functions Windows and graphic objects	*AI programming techniques* Rule interpreter Knowledge base bookkeeping, Inheritance mechanisms, Assumption maintenance Demon and message passing support Window system

7．知識獲得支援システムのイメージ

これまでの検討結果を踏まえて，各委員よりそれぞれの立場から知識獲得システムのイメージ，及び重要と思われる観点や視点の提案があった．要点を以下にまとめる．

①事前に与える情報の検討

②獲得した知識の次のインタビュへの利用

③ドメイン理論の獲得

④動詞を利用したインタビュ

⑤エキスパートシステム構築方法論との融合

⑥動詞を中心とした問題の構造分析

⑦generic taskの抽出とそれへの分解

これらの提案を踏まえて，次のようなシステムイメージを得た．

①知識獲得には生成，学習，インタビュによる獲得，修正，管理の５つの形態があること．従って，ＫＡＳとしてはそれら全てを包含すべきこと．

②エキスパートシステムの構築にはソフトウェアの部品合成的観点が重要であること．

③Generic taskの実現には，深い知識の有無と一般的な探索空間の設定の可否という２つの観点が重要であること．

④知識獲得はGeneric task毎に行うべきこと．

⑤Generic taskを抽象部品として捉え，問題をそれらの組み合わせで表現するためのインタビュシステムと，Generic task毎の知識獲得との２種類のインタビュシステムを中心とすること．

⑥上述のようなインタビュ，知識コンパイラによる深い知識からの浅い知識の生成，そしてＥＢＬによる学習という３つの要素を中心とすること．

本章ではこの考察結果に基づき，知識獲得支援システムのイメージの詳細について述べる[7-1]．

7.1　エキスパートシステム構築方法論における視点

知識獲得をエキスパートシステムの方法論の一環として考察するために，まず本節では，方法論を考えるための重要な視点を整理する．

7.1.1　エキスパートシステムと問題解決

エキスパートシステムには診断型や設計型などの多くのタイプのシステムがあり，それぞれ固有の特性を持っているにもかかわらず，現状におけるエキスパートシステムの構築はプロダクションルールという単純な構造を持つ知識表現方式に頼っている．プロダクションシステムは汎用性の高い推論エンジンではあるが，個々の問題の特性を反映するには概念レベルが低過ぎる．推論エンジンは対象とする問題の構造を反映しているべきであり，ツールはその機能を提供しなければならないと考えられる．

推論エンジンという概念はもはや適当ではない．推論の形態は各汎化タスクによって異なり，一様なプロダクションシステムではその特殊性を表現することは困難である．このようなことから，汎化タスクに対応するエンジンを，汎化タスクに固有の問題解決を行うという意味で，**問題解決エンジン**と呼ぶことにする．

経験則を重視し，それをルールベース化してエキスパートシステムを構築するという現状の方法論では，経験則さえルール化すればよいという考えが支配的であるように思われる．既に述べたように，エキスパートシステムは高度な問題解決システムである．専門家は問題解決を行っているということを忘れてはならない．問題解決には対象とするタスクに固有の方法論があり，専門家の問題解決過程を記述するのに適した抽象レベルでタスクを解析し，ドメインに依存せず，専門家の思考過程の本質を的確に表現する汎化タスクをエキスパートシステムの基本部品として捉えることは重要である．

GPS は汎用性を追求しすぎて失敗した．一方，ルールに基づくエキスパートシステムもやはり推論エンジンとしては汎用であり，知識を重視し過ぎている．両者の中間，即ち，汎用の問題解決ではなく，特定の問題解決に注目して，専門知識を分析するのと同様に問題解決過程を分析し，適切な範囲で汎用性を持つ問題解決システムを考察するところにエキスパートシステムのあるべき姿があるように思われる．

7.1.2 ビルディングブロック

汎化タスクを実現する際には対象とする問題の定式化を行わなければならない．問題の定式化としては，従来は経験則をルールで表現し，インタプリタがそれを解釈するという形態が殆どであった．これは問題解決の立場から言うと，ルールを設定することにより探索空間を規定していることになる．したがって，探索空間を規定するところに重要な知識が用いられていることになる．一方，別の定式化の方法もある．探索空間は一般的なものにし，その中での基本的なオペレータを定義して探索を行い，適用可能な複数のオペレータの選択，或は枝刈りに専門知識を用いるというものである．後者の方法は，R1-SOARで採用された方法であるが，従来のＧＰＳとの整合性が良く，学習が導入しやすいという利点がある．更に，従来の定式化であれば，知識が不足していれば探索空間が未定義となり処理ができなくなるが，後者の定式化では，そのような場合でも効率が低下するだけであり，問題を解くことはできる．この性質はある意味で本質的であり，エキスパートシステムの構築において重要な観点を提供するものと思われる．

汎化タスクをインプリメントしたものをビルディングブロックと呼ぶことにする．汎化タスクは概念的なものであり，エキスパートシステムの概念設計に有効な部品である．一方，ビルディングブロックは実際にエキスパートシステムを構築する際の具体的な部品と考えることができる．その意味でビルディングブロックは対応する汎化タスクの範囲で完全な問題解決器であることが望ましい．問題によっては，一方の定式化が困難な場合があるが，両者の得失を考慮して対処することは重要である．

7.1.3 エキスパートシステムの分類

上述の考えを更に進めれば，エキスパートシステム（厳密には汎化タスク）のアーキテクチャに関する次の二つの見方があることに気付く．

(1)深い知識の有無

(2)一般的な探索空間の設定の可能性

最初の観点は深い知識の有無である．対象とするドメインにおいて，基本原理となる知識が整備されているかどうかは知識コンパイラの利用可能性を決定づけることから重

要である．

２番目の視点は上述の，GPS的なインプリメントの可能性に関するものである．これらの２つの視点に基づけば，エキスパートシステムは図4に示す４通りに分類される．括弧内はその型のシステムの構築に関する対処法を示している．例えば，人工物の故障診断は深い知識があり，探索空間も設定可能である．その意味で最も素直な問題と言うことができる．対処法としては知識コンパイラによる浅い知識の生成が有効である．深い知識はあるが探索空間が事実上設定できないものとして機械設計がある．深い知識が完璧には整備されていない部分もあるかとも思われるが，機械の設計における深い知識は豊富であり，十分利用できる．しかし，部品の組合せ方には無限の多様性があり，探索空間を設定することはできない場合が多い．このような場合には設計例から，深い知識をDomain theory としたＥＢＬを用いた学習が有効に働く．設計とは逆の場合の例がXCONである．XCONのような配置に関する問題（スケジューリングも含む）は，一般に探索空間が容易に設定されるが，深い知識は明確でないことが多い．既に明らかなように，この場合はSOARのような探索に基づく問題解決とChunkingによる学習が有効である．最後の，深い知識もなく探索空間の設定も困難な問題，いわゆる経験的知識だけが頼りの問題としては，例えばＭＵＤ等がある．対処法としてはインタビュによる知識獲得以外にはないが，汎化タスクとそこにおける知識の利用法に基づくトップダウンなインタビュ戦略が重要な役割を果たす．

このように，エキスパートシステム，特に汎化タスクを考えるときにはこの二つの観点による分類は重要である．

7.1.4　知的ＣＡＩシステムにおける汎化タスクの例

汎化タスクの重要性はある程度認識されてはいるが，具体的にどの様なものがあるか現時点ではそれほど明確でないように思われる．ここでは，知的ＣＡＩシステムにおける汎化タスクとビルディングブロックの例を紹介する[7-2]．

教育における汎化タスクの同定は比較的容易であり，表１に示す１０個のタスクが代表例と考えられる．これらのタスクは教材に依存しないという意味で汎用である．診断型で言えば，「故障仮説の階層分類」が自動車，コピー機，病気等多くの診断対象に共通に使われるタスクであることに対応している．

しかしながら，表１に示したタスクは汎化タスクとしては不十分である．各々のシステムの中での役割は明らかにされているが，その実行方式については言及されていない．そこで，次ぎに問題となるのがビルディングブロックの設計，即ち汎化タスクの実現である．次に実現指針を示す．

①汎化タスクの問題空間の形式的定義
②問題空間上での問題解決器の設計
③教材に固有の知識の導入

Prologを表現言語として学生モデルを構築する例をとれば，

(1) 空間として，述語で表現されるPrologのプログラム
(2) 問題解決器として，Prologプログラムの合成システム（ＭＩＳ）
(3) 教材に固有の知識の導入法として，学生が誤り易い述語の優先的使用等

となる．又，学生の誤り分析の例では，

(1) 学生モデルと教材モデルの２つのPrologプログラムの中で同一のヘッドを持つ節同志の可能な組み合わせ．

(2) 節間の対応付けの結果，いくつかの問題に対して最も良く真理値が一致する対応づけの選択．

(3) 教材における学生の誤りパターン

このようにしてビルディングブロックを構築すれば，設定された空間の中での汎用問題解決器と教材に依存したモデル構成の効率化のための知識とが融合され，知的ＣＡＩのための強力な部品が実現される．

7.1.5 事前知識

エキスパートシステム構築支援を考える上でもう一つ別の観点としては，事前に与えられる知識である．エキスパートシステムに関する約１０年の経験を活かして，構築に関するKnowhow を整理して知識ベース化することが可能である．この観点から見ると，現状におけるエキスパートシステムの構築は汎用の推論エンジンであるプロダクションシステムを提供しているに過ぎない．ここで，何を事前に用意しておけるかを考察することは重要である．インタビュにおいては，タスクにおける知識の役割が既知であれば，前述のごとくトップダウンの効率の良いインタビュが可能となる．深い知識はドメインにおける一般原理であり，客観的な知識であることから，事前に用意することができる．又，汎化タスクは問題解決過程の抽象レベルで記述する際に用いられる部品であり，専門家の問題解決過程を分析することで整理しておくことができる．更に，エキスパートシステム構築の様々なKnowhow も徐々に蓄積することができる．このように，事前に与えておける知識とその利用法を検討することはエキスパートシステムの構築方法論を考える上で有益である．

7.2 知識ベース構築支援システムのイメージ

本節では，これまでに述べてきたことをまとめて，広義の知識獲得支援としての知識ベース構築支援システムのイメージについて考察する．

7.2.1 基本思想

本稿で述べる知識ベース構築方法論の本質は

エキスパートシステムの部品合成

という考えにある．事前に用意しておけるものをできるだけ多くし，要求に適合した知識やエンジンを生成し，必要な知識を学習やインタビュによって獲得するというのが基本方針である．

エキスパートシステムの構築において，抽象部品としての汎化タスクの重要性は既に指摘した通りであるが，エキスパートシステムの構築をソフトウェア工学における部品合成とのアナロジーのもとで考察することは意義深い．汎化タスクは，専門家が解決している現実の問題の構造を反映しており，ドメインに依存しない，問題解決の方法論を抽象化したものである．現実のエキスパートシステムの概念的な枠組みは汎化タスクを

組み合わせることによって構成することができ，具体的な枠組み，即ち問題解決エンジンはビルディングブロックの組み合わせとして構成される．

ビルディングブロックは更に基本的な推論技法，例えば仮説推論や定性的推論などの組み合わせとして構成される．あくまでも，ビルディングブロックはドメインに依存しない，問題解決の枠組みであることから，更にドメイン知識を導入して初めて問題解決エンジンの部品となる．

この考えを実現するためには次の課題が重要となる．

・知識コンパイラと問題解決器コンパイラという２種類のコンパイラ

・知識獲得インタビュとタスク分析インタビュという２種類のインタビュ

以下に各々について述べる．

7.2.2 二つのコンパイラ（知識コンパイラと問題解決器コンパイラ）

エキスパートシステムの部品合成において問題となるのは，与えられた問題を既知の汎化タスクに分解し，必要な汎化タスクを正確に同定することと，具体的にビルディングブロックを組み合わせて問題解決エンジンを構成することである．

我々の立場は，基本的で客観的な知識を事前に用意しておき，経験則に対応する知識を生成することによって，効率的な問題解決を行おうとするものである．予め用意しておけるものとして，深い知識と汎化タスクがある．前者は知識のコンパイル用のものであるが，後者は既に述べたようにエキスパートシステムの抽象部品である．従って，目的とするエキスパートシステムをこれらの部品を組み合わせて構築することになる．このために必要な機能には次の二つがある．

(1)与えられた問題の表現に必要な汎化タスクの同定

(2)与えられた問題のShallow Engine （問題解決エンジン）の生成

後者の仕事をするモジュールを問題解決器コンパイラ(ＰＳＣ:Problem Solver Compiler)と呼ぶことにする．

従来のエキスパートシステムで言えば，各汎化タスクが一つのエキスパートシステムに対応し，汎化タスクの実現であるビルディングブロック一つ一つが固有の推論エンジンを持っている．厳密には，ビルディングブロックの中には，推論ではなく，generate &test のgenerator の役目を果たすようなものもあるので，既に述べたように推論エンジンというよりは，問題解決エンジンと呼ぶ方が適切であると思われる．

なんらかの方法で合成に必要な汎化タスクが同定されたあと，専門家の推論プロセスに忠実にビルディングブロックを接続しなければならない．このとき，ビルディングブロック間の入出力属性に関する情報は有効であろう．

従って，合成されたエキスパートシステムは複数の問題解決エンジンが組み合わされてできており，それらがコンパイラによって生成されるわけである．

7.2.3 二つのインタビュ

知識獲得の基本方針として，汎化タスク毎に必要な知識の獲得を行うこととする．従って，獲得すべき知識としては，問題解決過程に関する知識と各タスクの中で用いられる知識の二つがある．又，インタビュもそれぞれに対応して知識獲得インタビュとタス

ク分析インタビュの２種類のものが必要となる．通常の知識獲得インタビュに関しては既に述べたので，ここでは後者について述べる．タスク分析インタビュは，与えられた問題を既知の汎化タスクに分解し，必要な汎化タスクを正確に同定することを目的としたインタビュである．汎化タスクは原理的には予め用意しておくことができる．汎化タスクの同定では，やはり専門家へのインタビュが主な作業となる．通常のインタビュが知識を獲得する為のものであるのに対して，ここでのインタビュは，専門家が用いている問題解決過程の枠組みを獲得するためのものである．専門家が自分自身の問題解決過程について語るときには，問題の解決に用いている知識それ自身と解決の方法論或は推論を進める方針とが混在して現れるので，それらを切り分ける必要が生じる．一般に，専門知識の理解には事前知識が必要であるが，問題解決法は，汎化タスクの定義より明らかなように，ドメインに依存しない表現が用いられると考えられるので，インタビュすることが可能である．このために必要な知識としては，問題解決過程の表現に現れる動詞の意味表現と，汎化タスクに固有の動詞の収集・整理である．このインタビュシステムの処理は，汎化タスクが持つ構造を動詞で表現したものを文法とみなせば，専門家が述べる問題解決過程のパージング（構文解析）と考えることができる．

7.2.4 概念設計

第５図 にエキスパートシステム構築支援システムの概念図を示す．知識獲得の４つの形態（知識ベースの修正を除く）がサポートされている．マン・マシンインタフェイスの部分にインタビュシステムが位置しており，上述の２種類のインタビュを行う．知識ベースは，深い知識ベース等の基盤知識ベース，汎化タスク知識ベース，そして浅い知識ベースの３種類に分かれている．基盤知識ベースは，いわゆる深い知識ベース，ＥＢＬのDomain theory,そして従来から用いられてきた問題解決プリミテイブからなる．浅い知識ベースは直接タスクの実行に用いられる知識からなり，インタビュによる獲得，ＥＢＬによる学習，例題からの決定木の生成，深い知識からのコンパイル，人手による入力の５種類の方法によって，汎化タスク毎に構築される．

知識コンパイラは深い知識からの浅い知識の生成を担当するが，汎化タスクの性質によってコンパイラが異なるのではなく，統一された方式が望ましい[7-3]．

問題解決器コンパイラは，必要なビルディングブロックを集めて，与えられた問題に適合した問題解決エンジン（Shallow engine）を生成する．これによって，エキスパートシステムの枠組みが作られることになる．タスク分析インタビュにより，対象とする問題解決に必要な汎化タスクを同定する．これは中流の知識のインタビュに相当する．ビルディングブロックの汎用の問題解決空間をタスクのドメインに適合させるための知識，即ち探索空間の状態を定義する知識をインタビュにより獲得する．これは一種のドメインモデルの獲得であり，上流のインタビュに相当する．そして，各ビルディングブロックの実行を高速化するための浅い知識（経験的知識）やビルディングブロック毎の浅い知識（下流）の獲得も行われる．

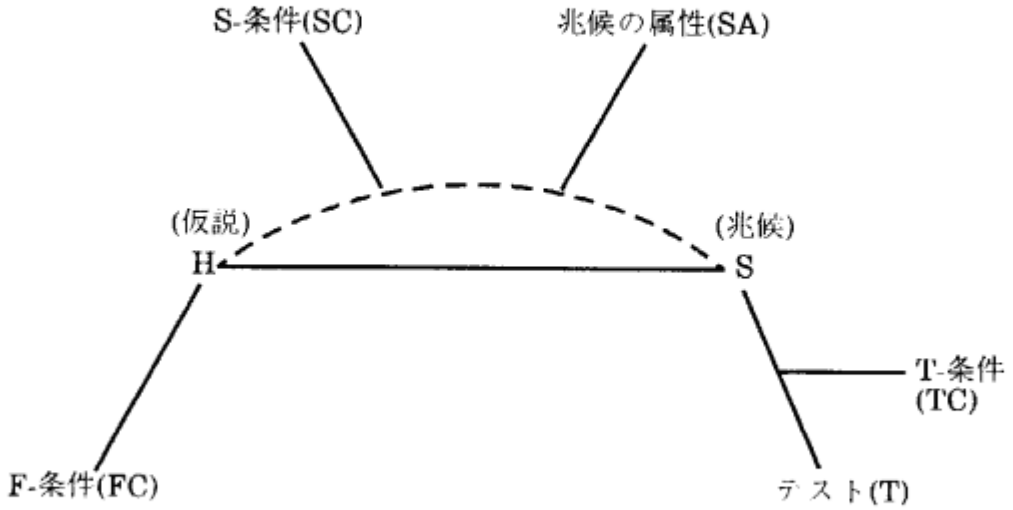


図1. MOREにおけるドメインモデル表現のプリミティブ

表1. 知的ＣＡＩにおける汎化タスクのクラス

（Ｓ）　学生モデルモジュール （Ｓ１）学生モデルの解釈・実行 （Ｓ２）学生モデルの構築
（Ｔ）　個人指導モジュール （Ｔ１）誤り分析 （Ｔ２）教育戦略 （Ｔ２１）宣言的知識の教育 （Ｔ２２）手続き的知識の教育 （Ｔ２３）双方向対話
（Ｅ）　教材知識モジュール （Ｅ１）教材知識の解釈・実行， （Ｅ２）オーサリング支援， （Ｅ３）教材シミュレータ， （Ｅ４）問題の自動生成

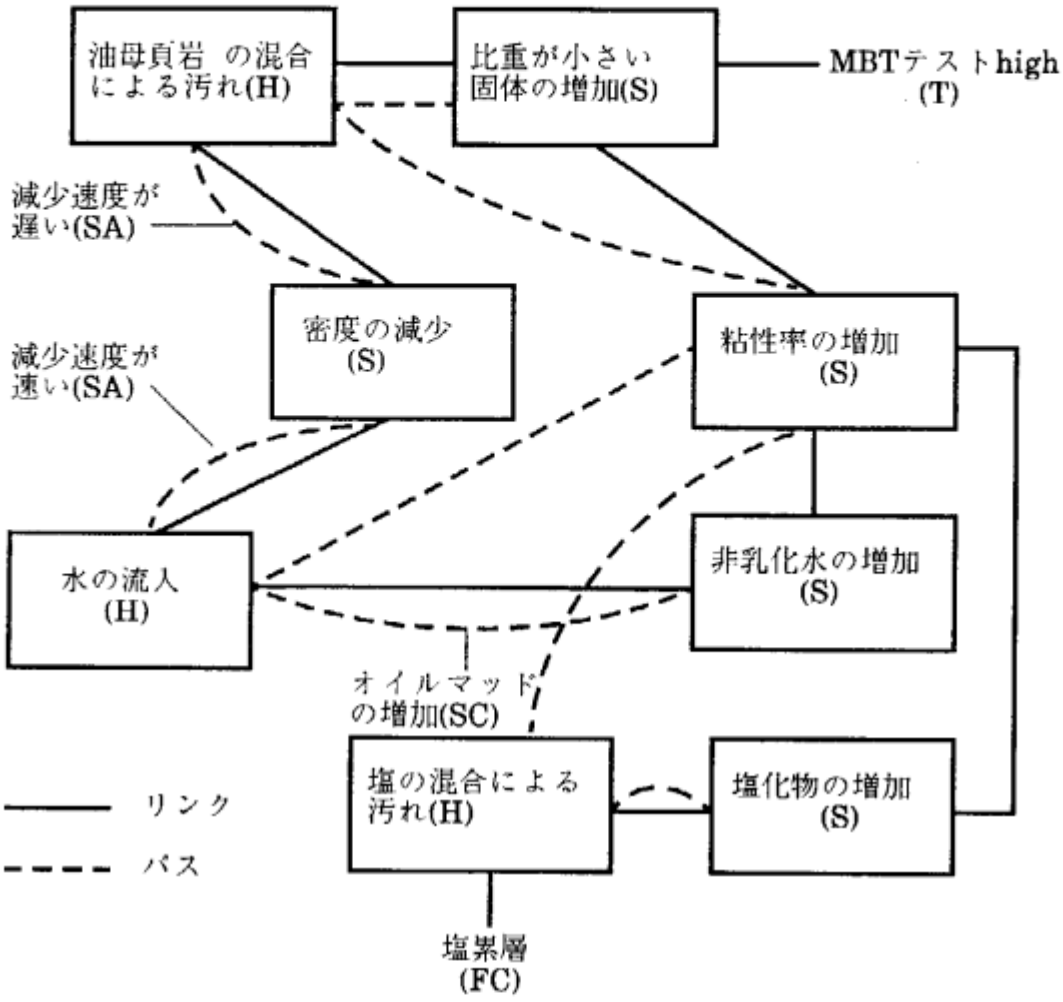


図2 MOREにおけるドメインモデルの例

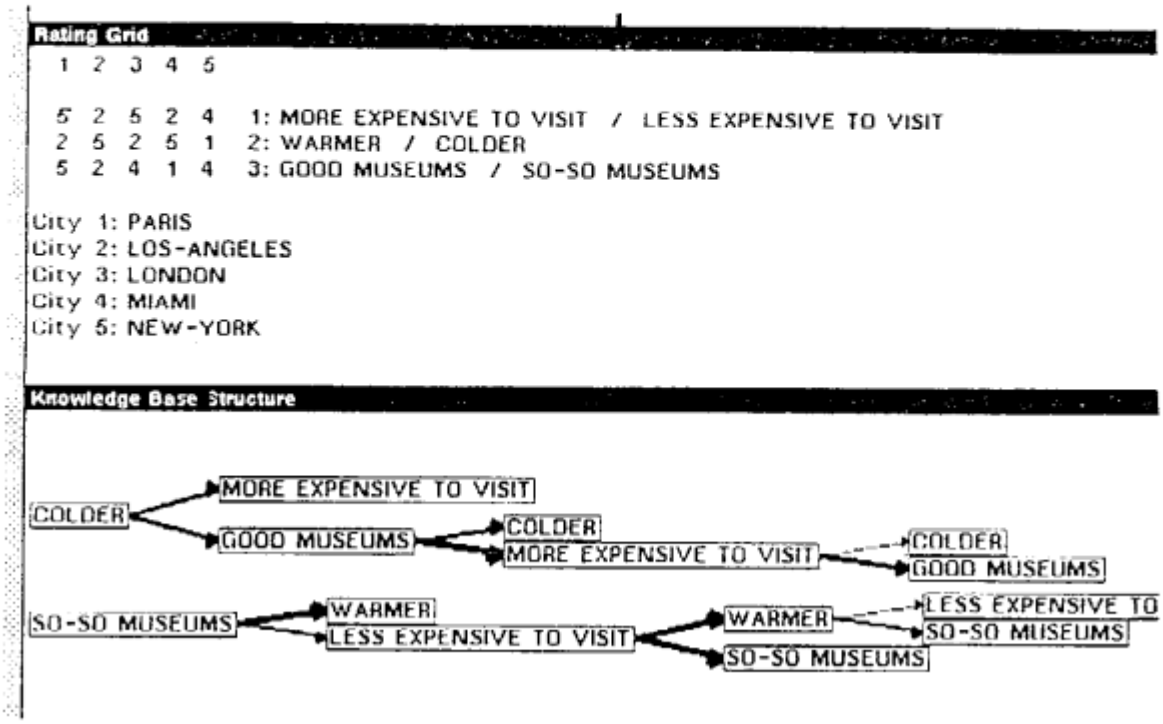


図3 ＥＴＳにおけるRating grid及び
属性の依存関係表現

<参考文献>

[7-1]溝口他：エキスパートシステム構築方法論について，人工知能学会研究会資料，SIG-KBS-880-2，1988.
[7-2]池田他：知的ＣＡＩのためのフレームワークの検討－学生モデル，帰納推論，教育戦略，情報処理学会，教育におけるコンピュータ利用の新しい方法シンポジウム，pp.49-58，987.
[7-3]山口他：エキスパートシステムにおける深い知識の統一的枠組み，人工知能学会研究会資料，SIG-KBS-880-3，1988.

		探索空間	
		可	不可
深い知識	有	故障診断（ＫＣ）	機械設計（ＥＢＬ）
	無	ＸＣＯＮ（ＳＯＡＲ）	ＭＵＤ（ＥＴＳ）

図4　エキスパートシステムの分類

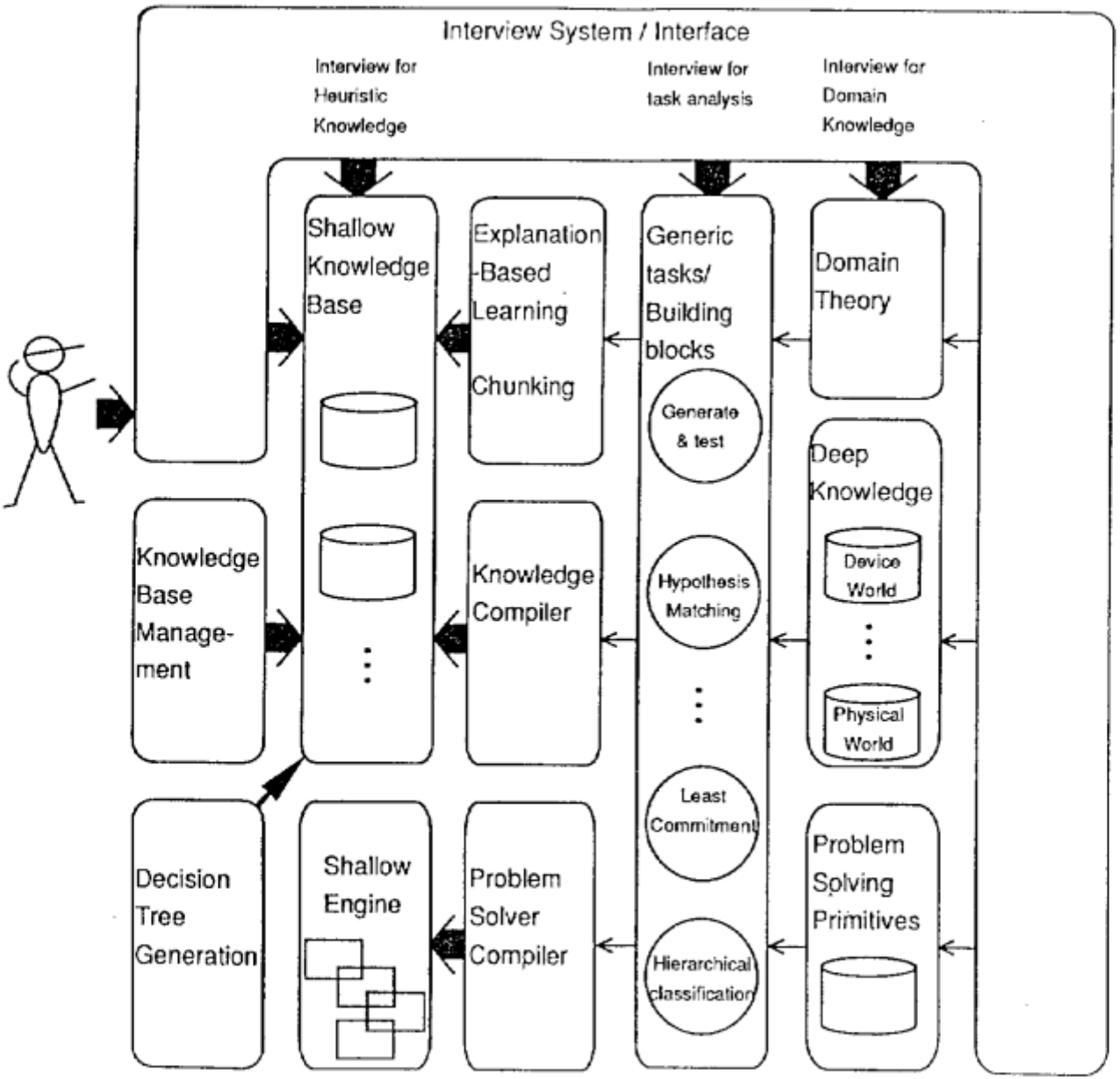


図5 エキスパートシステム構築支援システムのイメージ

8. むすび

知識ベース構築のための知識獲得支援システムのイメージについて述べた．各構成要素の同定とその役割は明らかにすることができたが，各要素相互の関係とそれらの詳細は不透明な部分が多い．本稿を終えるにあたって，今後の研究課題をまとめておく．

(1)汎化タスクの収集とビルディングブロック の実現

エキスパートシステムの部品合成を行うためには，良い部品が豊富に揃っていなければならなことは言うまでもない．これは一部の研究者が単独で行うのではなく，組織的な研究が要請される．

(2)タスクインタビュ

汎化タスクの抽出を行うと同時に，専門家の問題解決過程の認知科学的な検討を行って，タスク分析インタビュ技術を開発すること．

(3)知識コンパイラ

深い知識から浅い知識を生成する知識コンパイラの統一的枠組み，そして複雑なシステムの階層的な取り扱いを助ける階層的知識コンパイラの開発は，知識の生成という本方法論の中核技術として極めて重要である．

(4)問題解決器コンパイラ

問題解決器コンパイラは，部品合成の要であるにも拘らず未知の部分が多い．各ビルディングブロック間の制御の流れを知った上で問題解決エンジンを構成しなければならない．

(5)学習

学習は，決定木の帰納推論，EBL やChunkingを中心として更に研究を進めると共に，相互の方式の関連，及びEBL と知識コンパイラとの関係等を明確にして行かなければならない．

(6)インタビュ工学

タスク分析のためのインタビュを含む知識獲得のためのインタビュを工学的に論じる，インタビュ工学の推進も重要であろう．

又，本支援システムは，エキスパートシステム構築のための設計型エキスパートシステムの一種と見ることができるが，この観点からの考察も重要であろう．