4. 基礎ソフトウェア・システム

ICOT 研究所第2研究室長　古川康一
ICOT 研究所第3研究室長　横井俊夫

アブストラクト　基礎ソフトウェア・システムは第5世代コンピュータ・システムの中核ソフトウェアである．知識情報処理用の高度な並列型スーパ・コンピュータを作るために，超並列コンピュータ・アーキテクチャと知識情報処理との間の橋渡しをするものが強く求められている．このプロジェクトでは，概念的な橋渡しとして「論理型プロミング」が選択された．そして，基礎ソフトウェアは実際の橋渡しの役割を果すと考えられる．このプロジェクトの前期において，システム全体を設計するため，およびそのサブシステムを実現するのに必要な技術を開発するために，論理プログラミングが選択された．このプロジェクトは開始後まだ2年半しか経っておらず，研究計画全体のほんの一部を達成したに過ぎないが，我々は論理型プログラミングに基づいたアプローチが極めて将来性に富むものであると確信している．この論文は基礎ソフトウェア・システムについての我々の研究開発の現状を記述している．

1. はじめに

第5世代コンピュータ・システム（FGCS：Fifth Generation Computer Systems）プロジェクトの目標は，知識情報処理のための超並列コンピュータを作ることである．このプロジェクトでは，超並列コンピュータと知識情報処理との間の橋渡しとして，図1に示すように「論理型プログラミング」が選択された．

実際の“橋”を実現するためには，ここで基礎ソフトウェア・システムと呼ぶ高度なソフトウェア・システムが必要である．それは図2に示すように，次の5つのモジュールから構成されると考えられる．

（1）　核言語/知識プログラミング言語
（2）　問題解決推論ソフトウェア・モジュール
（3）　知識ベース管理ソフトウェア・モジュール
（4）　知的インタフェース・ソフトウェア・モジュール
（5）　知的プログラミング・ソフトウェア・モジュール

このプロジェクトの前期は上記各モジュールを実現するのに必要な基礎技術の開発に当てられ，各モジュールは独立に開発された後，統合化され，全体として基礎ソフトウェア・システムを形成する．

統合化は第5世代コンピュータ・システムの開発における極めて重要な問題であり，各モジュールの開発に際して共通のプログラミング言語を使うことによって達成される．このプロジェクトでは，そのために論理型プログラミングが選択された．一連の核言語，すなわちKL0，KL1およびKL2が計画されている．これらはハードウェアとソフトウェアの間の抽象的インタフェースを定義するための論理型プログラミング言語である．1982年の最初のステップとして逐次型推論マシンおよびそのユーザ言語であるESP（Extended Self-contained Prolog）のための機械語KL0が，共通言語として設計

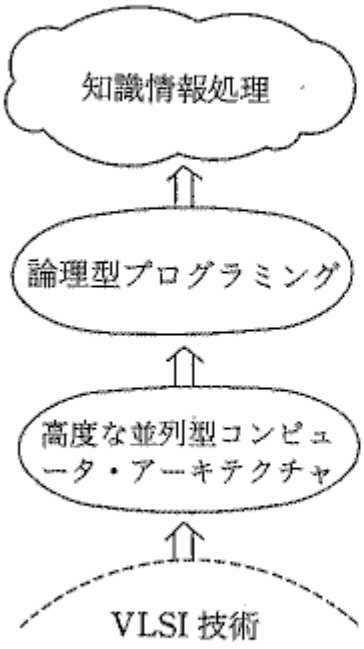

図1.　第5世代プロジェクト

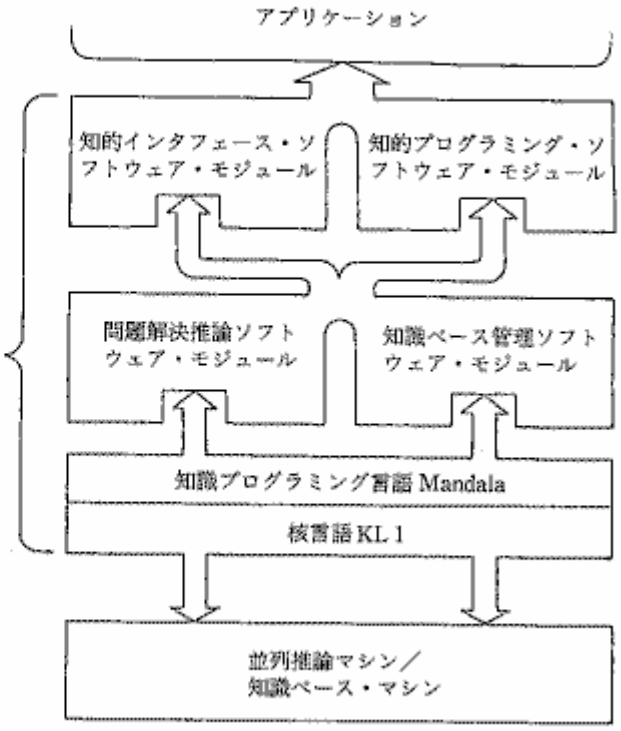

図2.　基礎ソフトウェア・システム

された．

この一対の言語 KL0 および ESP だけでは図1に示す目標を満足する最終の基礎ソフトウェア・システムを開発するには不十分である．そこで，目標を実現するための確固たる基盤を開発する目的で，KL 1 およびそのユーザ言語 Mandala が現在設計されつつある．KL 1 が Prolog に比べて本質的に拡張されている点は「ストリーム AND 並列」の機能で，これは並列に動作する複数のオブジェクトの行動を記述するために使われ，並列推論マシン（PIM）の上で並列に実行される．KL 1 に対するユーザ言語 Mandala（古川，他 1983 c）も，知識情報処理に関連した各種のアプリケーション・システムを作るための知識プログラミング言語となることを意図している．この言語は大量の並列性を抽出できるものとなる．したがって，Mandala の中の各機能はできる限り直截的な方法で実現されることが重要である．

問題解決推論モジュールおよび知識ベースソフトウェア・モジュールを1つの枠組みに統合化することが計画されており，この2つのモジュールは Mandala を介して結合されて強力な知識表現システムを確立することになる．

問題解決推論機能の中期における最大の課題は並列性を折り込むことである．この目標を達成するためのキー・ポイントは協調型問題解決である．我々は Mandala の中にいくつかの並列推論エンジンを実現するための検討を行った．CP で書かれた Concurrent Prolog（CP）インタプリタはその一例である．もう1つの例は CP で書かれた純粋 Prolog インタプリタである．一階の定理証明機および項書替えシステムの KL 1 による開発が計画されている．

知識ベース管理機能を取り扱う際，特に注目すべき分野は，（1）非常に大規模な知識ベースを取り扱う方法，（2）知識獲得の問題の研究，および（3）知識表現システムの設計である．

（1）に関しては，新しく開発された逐次型推論マシン PSI（Personal Sequential Inference machine）および関係データベース・マシン “Delta” が，大規模な知識ベースの実現に向っての第1ステップとして，ローカル・エリア・ネットワーク INI（Internal Network in ICOT）を通じて接続されようとしている．

(2)に関しては，知識獲得の問題に論理的な観点からのアプローチが行われている．無予盾性のチェック，事実の集合からの規則の帰納および真理値の維持問題の定式化が，メタ・プログラミング機能を使って Prolog の上に実現された．

(3)に関しては，前2者が KL 1 上に作り直され，Mandala を介して結合されて，究極の知識表現システムが中期に実現される予定である．

知的マン・マシン・インタフェース機能および知的プログラミング機能も，基礎ソフトウェア・システムの枠組の中で研究されている．この研究は FGCS プロジェクトの中で，基礎ソフトウェア・システムを作るための技法の確立，および論理型プログラミングのアプローチの評価およびその有用性の増大という2重の役割を演じている．

知的マン・マシン・インタフェース機能については，我々は自然言語の理解の研究に焦点を絞り，次の3つの課題に取り組んだ．

(1) 構文解析システム
(2) 文脈理解システム
(3) 電子化辞書

知的プログラミング機能については，この機能の実現に必要な主要問題点を求めるために，多数の独立の調査研究活動が行われた．これらの中にはソフトウェア仕様，プログラム理解，プログラム検証，プログラム変換，および自動プログラミングなどが含まれる．

次に，基礎ソフトウェア・システムの上記5種類のモジュールについて詳しく説明する．

2. 核 言 語

この論文では主として KL 1 について説明する．KL 1 には図3に示すような4つの主要な機能がある．それらは AND 並列機能，OR 並列機能，モジュール化機能(古川，他 1983 b) およびメタ推論サポート機能である．以下，各機能について簡単に説明する．

2.1 AND 並列機能

AND 並列機能は問題領域でのオブジェクトの行動を記述し，したがってオブジェクト指向プログラミング (Shapiro 1983 a) をサポートするために導入されている．AND 並列性を持つ論理型プログラミング言語としては，Relational Language (Clark 他 1981)，PARLOG

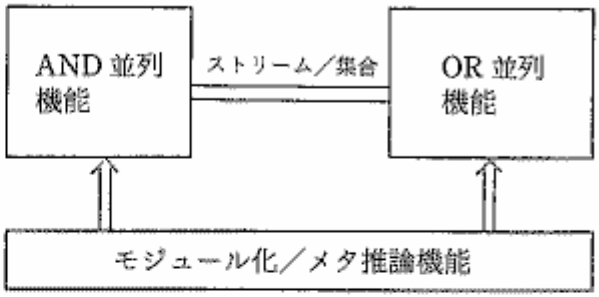

図3. KL 1 の概念的構成

(Clark 他 1984) および Concurrent Prolog (Shapiro 1982 b, 1983 b, 1983 c, 竹内 1982, 1983, 1984) などがある．KL 1 の詳細設計はまだ終っていないが，これらの言語がその設計に大きく影響することになろう．

2.2 OR 並列機能

KL 1 の非常に重要な要素の1つが OR 並列機能である．知識情報処理の多くのアプリケーションの中で，人工知能的なアプリケーションでは，多数の可能性の中から解を探索することによってのみ解かれる問題を扱うことが多い．OR 並列機能はそのような探索の問題を取り扱う．このプロジェクトの前期に開発される関係データベース・マシンは一種の全解探索マシンと見なされる．標準の Prolog から逐次制御とカット・オペレータをなくすことによって得られる純粋 Prolog は，全解探索機能を有する代表的な言語である．

KL 1 のこの2つの構成要素 (AND 並列機能および OR 並列機能) はストリーム/集合インタフェースによって連結される．言い換えれば，ある条件を満足するすべての解は OR 並列サブシステムの中では1つの集合として得られる．その後，この集合はデータのストリームとして AND 並列サブシステムの中で扱われる (平川，他 1983 b, 1984 a, 横森 1984)．

2.3 モジュール化サポート機能

KL 1 の主要条件の一つは，KL 1 をベースとするオブジェクト指向の知識プログラミング言語である Mandala を効率良く実現することである．この条件を満たすために，モジュール化をサポートする機能およびメタ推論機能が必要である．この二つの機能は密接に関連している．モジュール化の機能には二つの目的がある．一つはシステム・プログラム・レベルでのソフトウェアの開

発（プログラムの作成，管理など）を容易にすること，もう一つは知識を階層および複数の世界の中に構造化するためのサポートである．これらの目標を達成するために必要となる基本機能は KL 1 に組み込まれることになるであろう．基本機能としては，モジュール内のみで使われる局所化された述語，モジュールをパラメータ化するために必要な外部参照機能，モジュールの階層的構造化などがある．

2.4 メタ推論サポート機能

メタ推論機能の効率的な実現は極めて難しいが，それを達成することは上述したように Mandala の開発にとって重要であるだけでなく，知的エディタおよび知的デバッガの開発，並列に動作しているマシン間のインタフェースの開発，および協調的問題解決のアプリケーションにも大きく影響する．メタ推論機能はモジュール化サポート機能をベースにしている．モジュール化サポート機能はコンパイルされたプログラムを指定することによって与えられた目標を解決するための操作を含む（國藤，他 1984 a，1984 b）．この機能を核として使う場合，制御をより柔軟性のあるものにするには，制御方法をパラメータ化することが必要である．

KL 1 の暫定仕様は 1983 年に公表され（古川，他 1984 b），言語の詳細は現在設計中である．前の仕様は言語プロセッサのプロトタイプを実際に開発することによって，効率の面からの見直しが行われた（宮崎 1984，上田，他 1983，1984）．KL 1 の表現能力も，KL 1 のユーザ言語である Mandala で小規模な電子回路のシミュレーション・プログラムを書くことによって検討された．その結果，実際のアプリケーションで使われるそのような複雑なプログラムにおいて高い効率を得るためには，モジュール化機能およびメタ推論機能の効率の良い実現法の開発が重要であることが分かった．

3. 問題解決および推論の問題

問題解決推論ソフトウェア・モジュールおよび知識ベース管理ソフトウェア・モジュールは知識情報処理システムの 2 つの中心部分である．このプロジェクトの中期に計画されている最重点研究項目の 1 つは，これらのモジュールに並列性を組み込むことである．

並列実行環境に適合した問題解決推論ソフトウェアモジュールを設計する場合，2 つの重要な要素がある．その 1 つは問題解決機能そのものの並列化を実現することである．もう 1 つは問題解決プログラムの並列動作を制御することである．この他，サブモジュールの設計に際して，より本質的な問題を解くための高度な推論機能についての検討もなされた．

前期の目標は次の通りであった．

（1） 中期で問題解決推論ソフトウェア・モジュールを開発するのに必要な主要機能/構成要素の抽出．

（2） プロトタイプによる各機能/構成要素の設計．

抽出された機能/構成要素は次の通りである．

［1］ 並列推論機能

［2］ メタ推論機能

［3］ 高機能推論エンジン

以下，各機能/構成要素について簡単に説明する．

3.1 並列推論機能

並列推論機能の導入の目的は，推論のプロセスを高速化することだけではなく，分散型の問題解決を実現するために推論の能力を向上させることにもある．この機能は論理型プログラミング言語の並列実行の問題に深く関係している．POPS と呼ばれる OR 並列純粋 Prolog インタプリタ（平川，他 1983 b）が CP で開発された．このインタプリタは OR 並列計算を CP でストリーム AND 並列性に変換することによって実行する．POPS では，特別な計算機構を導入せずに，eager computation と lazy computation の両計算モードを実現している（平川，他 1984 a）．

分散型の問題解決に関しては将棋が例題領域の 1 つとして選ばれ，将棋の終盤における人間の問題解決についての研究が行われた．その結果，分散型の問題解決はその定式化の大局的枠組みを与えることが分かり，知識アーキテクチャと呼ばれる新しいモデルが提案された（近藤 1984）．このモデルはフィールド（通信メディアとしてのフィールド機能），認識型の知識，記憶型の知識，制御型の知識およびオブジェクト・モデル（オブジェクト・モデルは将棋盤に対応する）から構成される．

3.2 メタ推論機能

推論のプロセスは各規則の後に適用する規則を直接指示する，あるいは規則の使用をより一般的な（たとえば，正規文法などの）規則で制限することによって制御できる．メタ推論機能はそのような制御情報を使って推論を制御する（國藤，他 1984 a，1984 b，中島，他 1984）．

demo 述語は逐次型実行環境の下でメタ推論機能を実現するためのツールとして良く知られている．demo 述語には，与えられたゴール文が，与えられた制御情報を使って与えられた公理の集合から証明され得ることを示す機能がある．我々は，並列実行環境において上記と同じ機能を実行する simulate と呼ばれるシステム述語を開発した．simulate 述語も Mandala インタプリタを実現するために使われている（國藤，他 1984 a，1984 b）．

3.3 高機能推論エンジン

KL 1 に組み込まれている推論エンジンは純粋 Prolog および Concurrent Prolog（または同様なストリーム並列言語）のためのホーン節演繹型エンジンであるので，自然言語の理解，数式の操作，ゲーム，プログラムの合成など多くの目的のために，もっと強力な推論エンジンが必要である．

この目的で，2 つの推論エンジンが検討された．1 つは一階定理証明機，もう 1 つは項書換えシステムである．2 種類の一階定理証明機が，linear resolution（向井，他 1984）および lock resolution に基づいて設計された．上述した目的の他に，一階定理証明機の研究を通して PIM（並列推論マシン）を作るために適用できそうな他の推論エンジンの可能性を探ることも行われた．その結果，lock resolution および connection graph method が定理の証明に並列性を組み込むための方策として適していることが分かった．

実世界のアプリケーションから実用に供し得るための機能および性能に関する条件を抽出するために，電子回路設計における自動配置問題が検討された（古関 1984，光本，他 1984 b，森，他 1984 a，1984 b）．この研究によって，自動配線の高速アルゴリズムおよび，得られたレイアウトをグラフィック・ターミナル上に表示する機能などの低レベルの処理を扱うために，特殊化された推論要素が必要であることが分かった．論理型プログラミングの他にそのような特別の要素を必要とするのは過渡的な状態と考えられる．しかし，少なくとも現在では，Prolog と Fortran を結合することは実世界のアプリケーション・システムにおいてかなり有用であることが分かった（後藤 1984，光本，他 1984 a）．さらに，空間の問題を取り扱うための基本的技法を準備する目的で，2 次元のプログラミングも検討された（古川，他 1983 a）．

4. 知識ベース管理の問題

知識ベース管理ソフトウェア・モジュールは問題解決推論ソフトウェア・モジュールと同様に，基礎ソフトウェアの中心部を構成している．前期ではそれを開発するための前提条件として解決すべき問題点を抽出するために，つぎの 4 つのテーマについて研究が行われた．

（1） 大規模な知識ベースの実現
（2） 論理による知識獲得の定式化
（3） 知識表現システムの設計
（4） エキスパート・システムの開発

これらの研究の結果は中期以後で開発される知識ベース管理ソフトウェア・モジュールに反映される予定である．次に各テーマについて説明する．

4.1 大規模な知識ベース

大規模な知識ベースの構築への第 1 ステップとして，ホーン節の演繹システムを関係データベースとハードウェア・レベルおよびソフトウェア・レベルの両方で結合することが試みられた（北上，他 1984 e）．

ハードウェア・レベルの結合に関しては，新しく開発された Prolog マシン PSI を大規模データベース・マシン Delta とローカル・エリア・ネットワーク INI 経由で結合する方法が検討されている．同時に，この 2 つのマシンを密結合するための機構も検討されている．

ソフトウェア・レベルの結合（北上，他 1984 c）に関しては，ホーン節の演繹を関係代数とリンクするための新しい方法が提案された（國藤，他 1982，横田，他 1983 a，1983 b）．これはいわゆるコンパイル型のアプローチに基づいており，再帰的に定義される関係に基づく質問を取り扱う場合には関係代数式のシーケンスを発生する．

PSI と Delta の実際の組合せをサポートする実験的なデータベース管理システムが設計され，PSI 上で開発されつつある．それは KAISER（Knowledge Acquisition-oriented Information Suppli ER；知識獲得指向の情報提供システム）と呼ばれ，ユーザの親しみ易いインタフェースおよび知識獲得支援などの機能を含んでいる．

知識獲得の問題については後でもっと詳しく説明する．KAISER はまた，分散型知識ベース・システムのプロトタイプともなる．というのは Delta における外部データベースと同様に PSI の内部データベースも管理するからである．PSI のファイル・システム上の関係デー

タベース・システムも設計され，内部データベースを実現するために現在開発が進められている．

使い易いユーザ・インタフェースでは，談話における話題の変化の認識および省略形入力の意味推定などの会話の制御に重点が置かれた．その目標は多くの異なる応用領域で使用できる会話制御システムの構築である（宮地 1984 c)．

4.2 知識獲得

知識獲得の問題には論理的な観点からのアプローチが行われた．認識論的解析（國藤，他 1984 c，1984 d）の結果として，知識獲得のプロセスは知識同化プロセス（宮地，他 1983 a，1983 b，國藤，他 1983 a)，知識調節プロセスおよび知識均衡化プロセスから構成されることが分かった．したがって，KAISER には上記のプロセスをサポートする知識獲得機能（北上，他 1983 c，1983 e，國藤，他 1984 c)が備えられている．メタ推論機能を使って，これらのすべてが組織的に Prolog によって開発された．これらの機能の概念図を図 4 に示す（北上，他 1983 e，1984 a)．この図はメタ推論機能，知識同化機能，知識調節機能，および知識均衡化機能など，いくつかの機能から構成されている．

メタ推論機能はオブジェクトの知識の使い方に関する情報を管理（モニタ）する．メタ推論機構（國藤，他 1983 b，北上，他 1983 e）は知識獲得機能の実現にとって本質的なものである．"demo"述語（Bowen, 他 1982）がこの機能のプリミティブとして使われ，それは Prolog で書かれた Prolog インタプリタを拡張することによって実現されている．

知識同化機能（宮地，他 1984 a，1984 b，北上，他 1983 a，1983 b）は知識ベースの中で規定されている一貫性の制約条件に矛盾しない外部世界からの知識を組織的に同化する．同時に，この機能は冗長性を取り除く機構を備えている（宮地，他 1984 b)．

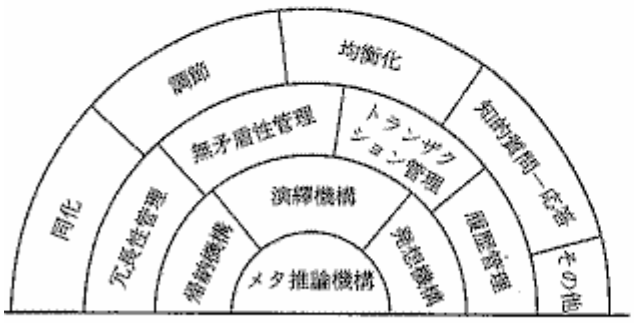

図 4． 知識獲得モジュール

知識調節プロセスにおいては，新しい事実が真であると仮定して現在の知識ベースの中の誤った知識の部分を訂正して矛盾をなくすようにする．1 つの実験的な知識調節システム（北上，他 1983 d，1983 e，1984 a）が Shapiro のモデル推論システム（Shapiro 1982 a）の機能を強化することによって開発された．これはあるクラスに属するモデルが満足すべき一般的条件を与える一貫性制約条件の考えを使っている．この機能強化によって，システムに与えられるバグ発見用の反例の量を減らすことに成功した．

知識均衡化機能（北上，他 1984 c，1984 d，1984 e）は前提型の知識と仮定型の知識（信念）との間に矛盾がないように調整する機能を有する（北上，他 1984 b)．これは真理値維持システム（Doyle 1979，Goodwin 1982，Martins 1983)において信念を翻意する機能を提供する．信念の翻意はメタ推論プロセスを通じて証明木を構築すること，および証明木の中の最も不確かな知識片を（もしあれば）ユーザから与えられた代替情報で置き換えるか，あるいは元の知識片の否定で置き換えることによって実現される．

KAISER においては，前に引用した"demo"述語（國藤，他 1984 b)の機能を強化することによって，これらの機能が実現された．これは論理型プログラミングを選択したことの利点を示している．本システムは，将来 KL 1 による統合化された知識獲得システムを開発し，並列実行により，また Mandala と組み合せることにより，効率を向上させることをねらっている．

4.3 知識表現

知識表現は人工知能の研究における主要テーマの一つである．すべての種類の知識を表現する単独の枠組を作ることは極めて難しいので，特殊化された知識表現システムを作るのに使える知識プログラミング言語が設計され，その処理系が試作された．ワーキング・グループ WG 4（主査：溝口助教授，コンサルテーション・システム研究委員会）において，そのような言語を設計するための予備調査が行われた（溝口，他 1984)．この言語は Mandala（古川，他 1983 c，1983 d，1984 a）と呼ばれ，

その詳細は別稿で報告されている（古川，他 1984 c）．Mandala は図 2 に示すように，第 5 世代コンピュータ・システムにおいて重要な役割を果すものと期待されている．その特徴を次に示す．

1. Mandala は単なる知識プログラミング言語ではなく，知識ベース管理システムの基礎となるものでもある．この二面性はホーン節の二重解釈（手続き的解釈および宣言的解釈）の機能に直接由来する．

2. オブジェクト指向プログラミング機能による問題記述，および親言語 KL 1 に由来する実行形態の両面において並列性が組み込まれている．

3. 各種のプログラミング・スタイルが用意されており，したがって知識情報システムを記述するための強力なツールとなりつつある．特に，静的な性質の記述以外に動的な行動の記述も可能である．

Mandala が知識プログラミング・システムとして利用可能となるためには，強力なプログラミング・サポートが必要である．そのようなサポートの役割を演じるのが，知識ベース・エディタである．Mandala の二面性によって示されているように，この知識ベース・エディタは通常のプログラミングにおけるトレーサ，デバッガ，エディタなどの機能のためのサポート環境に類似している．基本プログラミング言語が KL0 から KL1 へと進化するにつれて，並行性が更に重要になってきた．これらの機能を実現する場合，この知識プログラミング・システムを利用するのが有利であると考えられる．Mandala においては，知識は述語論理に形式化され，無矛盾性チェックなどの知識ベース管理が容易になっている．

知識ベース・エディタは Mandala で書かれた知識を表現するプログラムを編集する．ここで，編集するということは静的なプログラムの編集だけでなく，プログラムによって扱われる世界の編集をこれらの世界が変化した後で行うことをも意味している．したがって，知識ベース・マネージャおよびユーザ・インタフェース（これらは面方共プログラムの実行時に必要である）も知識ベース・エディタの構成要素と見なされる．単位世界エディタおよび実体エディタは，Mandala の基本構成要素である単位世界（KL1 プログラムの断片，モジュール）および実体（KL1 の実行プロセス）をそれぞれ編集するものである．自然言語理解の研究のための知識表現システムのプロトタイプが Prolog で開発された（杉山，他 1983）．これは ESP 用のものと同様に Mandala のための知識ベース・エディタを設計するために利用された．

4.4 エキスパート・システム

この節では，知識利用実験システムについて記述する．その目的は実用システムを開発する際にどのような機能が必要か，したがって，知識利用コンピュータ・システムのためにどんな機能が要求されるかを明らかにすることである．

1982 年度における準備検討から出発し，2 つの実験システム，すなわち，日本語校正支援システムおよび論理設計支援システムが選択された．両実験システムは現在開発中である．

4.4.1 日本語校正支援システム

この日本語校正支援システムは日本語の文章の中の誤りを見付け，可能な場合はそれを訂正する知識利用システムである．このシステムはコンピュータ利用の日本語文書作成システムの一部を構成すると考えられる．

まず，約 4,000 行の新聞記事に対して行われた校正を使ってどの様な校正が行われているか，更にコンピュータ化にはどの様な機能が必要かを調査した．その結果，56%の訂正のコンピュータ化には文章や記事の深い理解が必要であることが分かり，これは将来の問題である．

一方，訂正の 44%は，辞書や用語集の中の知識を使うことおよび人名録類の参照によって，現状でも比較的簡単にコンピュータ化できることが分かった（石井 1983 b，1984）．したがって，このシステムはそのような知識を効果的に扱うように設計された．知識の構造は重要な問題であり，慎重な調査が行われた（石井 1983 a）．

4.4.2 論理設計支援システム

論理設計支援システムはコンピュータのハードウェアの論理設計を支援する知識利用システムである．このシステムはハードウェアの動作アルゴリズムとして与えられる設計仕様を既存の回路素子の接続関係に変換するものである．

機械的な変換プロセスを使うだけでは，冗長度の大きい設計となり，そのシステムは使いものにならない．一方，人間は経験によって蓄積された各種の知識を使って，良好な設計を得ることができる．

この論理設計支援システムはそのようなノウハウを Prolog の中に取り入れ，コンピュータによる論理設計の高度化を図ろうとするものである（丸山，他 1984）．そ

のようなノウハウは Prolog で十分に記述できることが分かった．構造化された知識の取り扱いの可能な知識表現システムの利用が今後の課題である．

5. 知的マン・マシン・インタフェース

5.1 構文解析（パージング）システム

この研究の目的は高機能文法記述言語および効率の高い構文解析機能を提供するパージング・システムを開発することである．

我々のアプローチは論理プログラミング言語に基づいている．既に指摘したように，論理プログラミング言語と文脈自由文法（CFG）は密接に関連している．すなわち，ホーン節は CFG の規則の手続き的解釈を与えると見ることができる．CFG を構文解析の基礎として採用することは自然である．我々の主目的は，論理プログラミング言語での CFG に対する効率の高い解析方法の開発，および柔軟性の高い強力な文法記述システムの設計である．この研究には 2 つの主要部分，すなわち，構文解析法および文法記述システムがある．次にこれらの現状について簡単に説明する．

5.1.1 構文解析法

我々は ETL（電総研）および TIT（東工大）の協力を得て，BUP システムと呼ばれる拡張 CFG パージング・システムを開発した（松本，他 1983，田中，他 1984 a，横井，他 1982 a，1982 b）．このシステムはボトム・アップ型の深さ優先のアルゴリズムを採用し，当然 Prolog に埋め込まれている．この機能により効率の高いものになっている．さらに，BUP システムはパージングの過程の中間結果を記録することによる最適化手法を採用している．また，このシステムは BUP トレーサ，BUP トランスレータ，ε レデューサなど，文法開発用のツールをいくつか備えている．

BUP システムの開発に加えて，我々は構文解析に並列実行の機構を適用する方法を開発した．並列実行のパージング・システムはチャート・パージングのアルゴリズムに基づいており，Concurrent Prolog で開発されている（平川 1983 a，平川，他 1983 b，1984 b）．このパージング・モデルは複数のプロセスおよびそれらの間のメッセージ転送から構成されている．

パージング・システムに関連して，形態素解析システムが日本語の文章の解析用に開発された．日本語は膠着言語なので，日本語の形態素処理は英語の場合より複雑である．この形態素規則は語形変化の情報および語の連結の状態を含む．形態素規則は DCG 記述によって表現される（三吉，他 1983）．

5.1.2 文法記述システム

主な関心事は文法的関係を表現するための形式的システムに関する研究である．このシステムは文章が文法的に正しいかどうかのチェックおよび大規模な文法の維持管理のために不可欠である

実際には，現在の言語学での新しい文法理論である，LFG および GPSG が我々のシステムのための候補として採用されている．これらは言語学的現象および強力な文法記述システムを非常に理解し易い形式で一般的に説明する．この理由で，これらの理論は構文解析に有用である．LFG システムは DEC-10 Prolog で開発されている（安川 1983，1984）．LFG の計算機構は F-構造に関する等価性を Prolog に導入することによって実現できる．LFG の文法規則は DCG の規則に変換できる．これらに対してトップ・ダウンおよびボトム・アップの両方のパージング戦略が適用できる．GPSG システムを実現する方法は現在ワーキング・グループ WG 3（主査：田中助教授）で研究されている．この研究テーマは GPSG の枠組みのための効率の高いパージング・アルゴリズムおよび GPSG の中での基礎日本語文法の開発である．

DCG の整理および拡張の観点から，GDL0（Grammer Description Language 0 ： 文法記述言語 0）が設計され，DEC-10 Prolog で開発されている（森下，他 1984）．GDL0 は文法のカテゴリ定義のためのデータ構造および，文法の読み易さおよび変更容易性を改良するためのマクロ機能を導入している．また，オブジェクト指向のプログラミング言語の概念に基づく文法記述システムが開発されている（三吉 1984）．このシステムは Head Feature Convention および Control Agreement Principle などの文法上の関係の定義にクラスの概念を導入している．

今後，この文法記述システムに関する研究および開発を継続する予定である．並列実行機構の導入を含め，このシステムの高度化および拡張を行う必要がある．

5.2 文脈理解システム

文脈理解の計算モデルを確立することは，自然言語理解研究の中でも，重要で多くの難しさを含む問題である．モデルは，第 5 世代コンピュータに柔軟で使い易い自由

なインタフェースを作り出すことにも結びつく．本節では，文脈理解のアプローチ方法と研究の現状を概観する．

5.2.1 基本的アイデア

会話のモデルには少なくとも4つの重要な要素がある．それらは発話行為，一貫性(coherency)，前提(presupposition)，および相互信念(mutual belief)である．話し手と聞き手は，これらの要素をいわゆる文脈の行動原理，すなわち，通信のための一貫性および最適性の原理を使うことによって計算している．このことはシステムが強力な推論能力を備えていなければならないことを意味している．

我々は文脈理解に関して次の立場を採用した．

（1） 会話の理解は意志疎通のための一貫性および最適性が計算の原理となっている動的なプロセスである．

（2） このプロセスの目標は会話参加者間の発話（記述された）と相互信念のための一つの世界モデルを構築することである．

（3） このプロセスはその世界の部分モデルに関する仮説を立てることおよびその世界または会話の相手に関する知識を使って，その仮説をテストするサイクルを繰り返す．

（4） 実際の世界は事象および対象物から構成され，それらは互いに種々の関係にある．

（5） 構築中の世界モデルはシステムの知識構成要素にフィード・バックされる．

（6） メンタル・モデルおよび行動の理論は文脈の理解において重要な役割を果す．

我々の基礎ツールは Barwise & J. Perry によって自然言語のモデル理論的意味論のための新しい枠組として，スタンフォード大学で開発された状況意味論（Barwise，他 1983）である．この理論は上記の観点に良く合っているように見える．前期においては，日本語での会話の分析に努力が向けられる．

5.2.2 現在の活動

現在，我々は状況意味論の影響を受けた文脈理解システムのプロトタイプを設計試作中である．このシステムは日本語で書かれた物語を読み，その中で記述されている状況を構築し，それらに関する各種の質問に答える．

ケース・スタディとして，日本語の物語の本の一部を選んだ．この物語は大勢の客を乗せ，エンジンが故障した状態で飛んでいる飛行機の中の危険な状態を記述している．この物語の中では，困難な状況において最善をつくそうと努めている機長と1人のスチュワーデスの間での言葉のやり取りを含む，いくつかのアクションが発生している．

我々は1年近くの間，この文章の意味を分析しようと努力してきた．このテキストの中のすべての文章に対して完全な意味を与えるには多くの困難があることを知った．

困難は次の領域にある．

（1） 定量化された名詞句，副詞句，助動詞，および発話行為の動詞

（2） 文章または語句の省略

（3） 文章の解釈を制御するための各種の知識の表現および使用

（4） フォーカシング・プロセスのダイナミックス

我々は状況意味論が会話のコンピュータ的モデルにとっての前途有望な理論であると確信している．これらの困難を解決するために，状況意味論に照らして言語理解の枠組みを捉え直そうとしている．これは，一つは自然言語の文章の意味を定めるため，もう一つはその意味を理解するために必要である．

次に二，三の関連技術メモ(向井 1984 d，1984，鈴木 1984）を紹介する．会話テキストの実例解析がワーキング・グループ WG 3 の現在のテーマである．

5.3 電子化辞書

第5世代コンピュータ・プロジェクトにとって，十分な量の言語データを整備すること，とりわけ，自由に利用できる電子化辞書の開発が極めて重要である．このような言語データは自然言語処理，知識ベースなどの研究における基礎的なデータベースである．これらの電子化辞書は，将来，たとえば機械翻訳用の変換辞書，文章の理解のための意味論的辞書など，多くのアプリケーションに変換されて使われることになる．

電子化辞書に関するリサーチ・グループ（主査：石綿教授）は辞書の開発計画を立てた．この開発プロセスは2つの段階に分かれている．第1期(1984年から3年間)では，次の4種類の汎用マスタ辞書を開発する（石綿，他 1984）．

・日本語辞書
・英和辞書

・和英辞書
・英語辞書

各マスタ辞書には，出版されている標準的な辞書に記述されている文法上の情報および，コンピュータ処理用の文法情報も含まれる．

第2期（1987年から2年間）では，意味論的情報および語用論的情報が追加され，機械翻訳など各種のアプリケーション用の辞書に変換される．

電子化辞書を開発する目的は，深い意味解析のためのツールを提供することである．WG 3では，高品質の機械翻訳を実現するためのアプローチとして，専門の翻訳家による実際の翻訳の特徴が調査された（田中　1984 b）．意味解析用の予備実験的辞書システムとして，5000項目の小規模な辞書の開発が進められている．この中にはシソーラスおよび語用論的情報が含まれる．この辞書は文章理解システムに使われる．特定の単語が使われている文脈を解析するために，オンラインKWIC（Key Word In Context）システムが開発された．これは大量の言語学的情報を提供する便利な辞書である．

6. 知的プログラミングシステム

6.1 論理型プログラミングのためのプログラミング環境

SIMに対する高度なプログラミング・システムが現在開発中である．その詳細は別の論文（高木，他　1984）に報告されている．この活動は知的プログラミング・システムの研究用の実際の土台および経験を得ることを目的としている．

次にSIMプログラミング・システムについて簡単に説明する．

このシステムはエキスパート・プロセスの集合として設計されている．特別のエキスパート，コーディネータがこれらのエキスパート・プロセスを管理する．代表的なエキスパートはデバッガ/インタプリタ，エディタ，およびライブラリである．デバッガ/インタプリタはプログラムを解釈し，「手続きおよび文節ボックス制御フロー・モデル」に基づいてプログラムの制御の流れに関するデバッグ情報を提供する．エディタ（Edipsと名付けられている）はESPプログラムの編集に特に便利なように設計されている構造エディタである．ライブラリはクラスの登録，プログラム・ファイルのローディング，およびインヘリタンス解析によるクラス・オブジェクトの作成を管理する．

現在，これらのエキスパート・プロセスはあまり知的ではない．しかし，これらの知的能力はその内部機構に問題解決および知識ベースの機能を追加することによって徐々に向上する予定である．

6.2 プログラムの仕様，検証および変換

知的プログラミングの研究分野には，次の3つの主要研究要素がある．

・プログラムの仕様
・プログラムの検証
・プログラムの変換

6.2.1 プログラムの仕様

ソフトウェア工学の研究における最も難しい部分の1つはプログラム仕様記述システムである（Staunstrup 1981）．一般に，我々は形式的でない（理解し易い）仕様および形式的な（論理的に確立された）仕様の両方に関心がある．

・自然言語仕様：形式的でない仕様としては，我々は自然言語，特に日本語に関心がある．このため我々は榎本教授の率いる東京工業大学のTELLプロジェクトに協力している．TELLプロジェクトでは制限された構文形式の中で英語と日本語の両方が考慮されている（榎本，他　1984 a）．

形式的な仕様については次の2つのアプローチがある．

・一階述語論理：これは我々の研究のベースの中に論理型プログラミングが採用されているので，妥当なアプローチである．検証システムはこのアプローチの中で開発されている．
・時間論理（榎本，他　1984 b）：述語論理のアプローチには，システムの動的な行動が記述できないという問題点がある．このギャップを埋めるために，時間論理の枠組みが現在検討されている．

その他のアプローチとしては，たとえば次のものがある．

・プログラミング言語そのもの：Prolog自身が1つの有用な仕様と考えられる．

一方，仕様のためのサポート・システムが開発されるべきである．この方向で，いくつかの試み（たとえば，榎本1983，杉本1984 a，1984 b）が行われた．

6.2.2 プログラムの検証

Prologのプログラム用の検証システムが現在開発中である．これは“知的プログラミング・システム”の研究の基礎的な仕事であると考えられている．我々の検証システムの設計においては，論理型プログラミングの枠組みによって与えられるものの可能性を明らかにし，Prologの特性をできるだけ利用することを試みた．Prologの実行が1種の推論であり，Prologの意味構造が一階述語論理で非常に容易かつ簡潔に表現されることは特に便利である．

我々の検証システムでの一階述語の推論は実行スタイルの拡張形となっている．推論は肯定的目標の実行または，否定的な目標に対しては“失敗としての否定”のように実行される．これらの規則にケース分割を加えたものおよび簡単化の規則はPrologプログラムの多くの特性を証明するのに十分強力である．

Prologに対する計算的な帰納法の適用は関数的プログラムに対する場合よりずっと簡単である．というのはPrologの基礎となっている単純な一階の意味構造の内部だけで適用できるからである．作り出されるインダクション公理も単純である．これによって検証が効率的に行われ，一階述語に関する推論のオーバヘッドが大きいという欠点をカバーしている（金森 1984 b）．

多くのBMTP（Boyer-Moor Theorem Prover）スタイルの推論および論証の適用を制御するヒューリスティクスがこのシステムに統合化されている．特に，Prologのために新しく開発されたタイプ推論は効果的に使われている（金森 1984 c）．

6.2.3 プログラム変換

プログラム変換は与えられた（有効な）仕様から正しいプログラムを作り出すための方法として有望である．

理論的な観点からは，変換のステップの“正しさ”を検証する必要がある．玉木はこの等価性維持変換に対する枠組みを与えている（玉木 1983，1984 a，1984 b）．渕は基本的なPrologインタプリタを作り出すための変換技法を実証している（渕 1984）．

6.3 特定のサブプロジェクト，CAPおよびACT^{-1}

ワーキング・グループWG 5（主査：広瀬教授，基礎理論研究委員会）はコンピュータ科学の理論的側面の現状と将来動向についての検討を続けてきた．その検討から得た結果に基づいて，WG 5は2つの特定研究プロジェクトを設定した．それらはCAP（Computer Aided Proof）およびACT^{-1}（Theoretical Computer Architecture）と名付けられた．

6.3.1 CAP

CAPプロジェクトは数学的問題を解くための人工知能およびそのような作業における理想的なマン・マシン・インタフェースを検討するために，いくつかの具体的な定理の証明チェッカを作ろうとしている．我々は次の3つのターゲット理論をCAPプロジェクトのために選択した．

・線型代数
・記号演算
・綜合微分幾何

このうち，最初の線型代数は最も良く知られているので，説明を要しないであろう．証明チェッカの最初の目標は，新人用コースのための教科書である．

2番目の記号演算は，佐藤による一連の仕事（佐藤，他 1983 a，1983 b，1984 a，1984 b，1984 c）に端を発している．プログラミング言語Qute（これ自身がこの理論に基づいている）が，そのシステムの開発に使われる予定である．この理論の特徴は，ペアノ演算の場合のように，自己参照が可能であるということである．したがって，最終目標はこの理論の不完全性定理の証明であろう．

最後の綜合微分幾何はKock（Kock 1981）に始まる数学の非常に新しい分野である．この理論は無限小のオブジェクトを導入することによって綜合的な手段で微分幾何を研究することを目的としている．このプロジェクトはまだ基礎的理論調査の段階であり，実際の研究は来年から開始される．

6.3.2 ACT^{-1}

論理型プログラミング，関数型プログラミングおよびオブジェクト指向プログラミングなどの新しいプログラミングの枠組みは，ACT^{-1}の研究にとっての起動力となっている．現在，上記3種類の枠組みは，アルゴリズムについて考える場合の概念的枠組みを別々に提供している．さらに，それぞれがその底流に1つの計算モデルを用意している．たとえば，論理型プログラミングでは一階述語論理，関数型プログラミングではラムダ計算，そしてオブジェクト指向プログラミングではアクタ・モデ

ルがそれである．

したがって，我々の研究の本質は次のことにある．

- 上記3種類の枠組みの相違点および類似点を計算モデルのレベルで識別すること．
- できれば，上記3種類の枠組みを計算モデルのレベルで，あるいはもっと下の実現レベルで統合する方法を見出すこと．
- 上記枠組みの中に並列性（暗黙的または明示的な）を組み入れる方法を探索すること．
- 各プログラミング枠組みの中で開発されたプログラミング技法を評価すること．たとえば，d-リストを使った遅延評価およびストリーム通信．
- 計算モデルの実現のいくつかに関する計算量を解析すること．
- 計算モデルの実現のための現在の技法（ハードウェアおよびソフトウェアの両方）を見直すこと．
- 計算モデル（上記1および2の意味での）をデータ・フロー・モデルやパケット・リダクションモデルなどの更に詳細な実現指向の計算モデルに関連付けること．

現在の研究段階では，実現の側面よりはむしろコンピュータ・アーキテクチャの理論的側面に注目している．

6.4 既存技術への転移

知的プログラミングに関する研究においては，既存のソフトウェア技術との密接なコンタクトを保つことが非常に重要である．そこで，特別のワーキング・グループWG 6（主査：斉藤助教授）が組織され，FGCSプロジェクトに対するソフトウェア業界からの意見，期待および要求を調査することになった．このグループでは，新しい技術をソフトウェア開発環境の改善のために適用する方法などの問題を検討している．

この目的の別の活動として，このプロジェクトの成果を実際のソフトウェア開発の問題に適用するための具体的なシステムの開発が試みられている．COBOLプログラムの再利用のためのシステムの研究および開発がこの活動の一例である．

7. まとめおよび将来動向

このプロジェクトは開始後まだ2年半しか経っておらず，研究計画のごく小部分を遂成したに過ぎないが，我々は論理型プログラミングに基礎を置くアプローチが非常に有望であると確信している．

現在までに得られた成果は論理型プログラミングに勢力を集中したことによって達成されたものである．この研究の基本戦略は，困難な問題を論理型プログラムによって再挑戦することである．このアプローチは個々の問題に対して最良の解を与えるとは限らないが，個々の成果を基本となる共通プログラミングの枠組みによって1つのシステムに統合化するための機構を提供することになる．

プロジェクトの中期を成す次の4年間の目標は，これまでに開発された概念および構成要素を組み入れることによって各サブシステムを構築することである．次のサブシステム作成段階に進む前に行うべき作業はかなり残されているが，各サブシステムについてのイメージはほぼ固まっており，したがってプロジェクト全体としては，次の段階である中期に進むことが可能である．

次の段階での主な研究課題は並列性の導入である．これは非常に難しい問題であり，我々が本質的に並列性を有する論理型プログラミング言語KL 1の設計に力を注いで来たのはそのためである．

システムの統合化も中期に解決すべき，困難ではあるがやりがいのある問題である．

謝辞

この研究は8社のメーカの非常に密接な協力を得てICOT研究所の第2研究室および第3研究室によって遂行された．ワーキング・グループWG 2，WG 3，WG 4，WG 5およびWG 6のミーティングにおいて多くの有益なディスカッションが行われた．