データベース・システム 57－12
（1987. 1. 19）

知識情報に基づく駆動型メソッドベース

持田あけの，千葉正喜，前田 陰
（北大大型計算機センター），（北大工学部）

メイン・プログラムの集合がひとつの処理系をなしており，目的に応じてプログラムを組み合せ、実行順序を決めて問題解決を行う系がある。このような系の実行制御を目的として、プログラム実体，およびプログラムモジュールの機能，操作，動作環境等に関する情報を蓄積して，目的を与えるとプログラムを合成しながら自動的に実行するプログラム・ベースを提案する。プログラムの合成・実行制御に関して，個々のプログラムの実行上の情報・知識，プログラム間の関係，およびプログラムの実行に関する手続き的知識を利用することから，メソッド・ベースと呼んでいる。

A METHOD BASE USING KNOWLEDGE INFORMATION

AKENO MOCHIDA , MASAKI CHIBA , TAKASHI MAEDA
(HOKKAIDO UNIVERSITY COMPUTING CENTER) (A DEPARTMENT OF TECHNOLOGY)
(英称住所) KITA 11-JO, NISHI 5-CHOME, SAPPORO, JAPAN

1. はじめに

大規模なソフトウェアを開発・維持する支援ツールとして、プログラム・モジュールをソフトウェア部品としてデータベースで管理する試みがなされている。

一方、プログラムの巨大化や複雑化、冗長化を避けるために、処理の手順を分割してそれぞれを独立した処理機能を持つメイン・プログラムとして作成しておき、目的に応じてそれらを組み合わせて実行する方法もとられている。この場合も、目的を遂行するためのプログラムの組合わせを、「プログラム」、個々のメインプログラムを「部品」と見做すことができる。

いわゆる部品データベースでは、部品がソースプログラムの場合は利用者が作成するプログラムに部品を埋め込み、必要があればそこで修正して利用し、部品がオブジェクト・モジュールの場合は利用者プログラムと結合（リンクエディット）して利用するのが一般的である。このような利用形態は部品をデータベースから抽出してプログラムを作成する過程と、プログラムの実行とが異るフェーズで行われるという意味で"静的"であると言える。

我々は、上述したようなメイン・プログラム群を部品として持つ部品データベースに、実行制御と結果検証の機能を持たせ、"動的"な利用形態をサポートするプログラム・ベースを提案する。　このデータベースには、部品であるプログラム群と、プログラムを検索するための情報（プログラム・モジュールに関する情報なので、以下モジュール情報と呼ぶ）の他に、プログラムモジュールの実行に関する情報（以下モジュール制御情報と呼ぶ）、およびプログラム一般の実行制御、検証のための手続き的知識が重要な要素として含まれることから、これをメソッド・ベースと呼ぶことにする。

メソッド・ベースの特長は、検索した目的のプログラムを実行を通して更に検索を進めることができる点にある。

本稿では、メソッド・ベースの背景とねらい、および基本構想について述べる。

2. 背景とねらい

プログラムを内容に持つデータベースでは、検索時に即プログラムの実行が許されるべきである。北大大型計算機センターの運用面においても幾つかの面から必要性が指摘されている。

(1) ライブラリ管理システムにおいて

当センターではライブラリプログラムの開発を公募し、利用者が作成したプログラムを他の利用者に公開する制度が十数年にわたって推められている。こうして開発されたプログラムは数も増加し、応用分野も多岐にわたるため、内容、機能、応用分野などをキーに目的とするプログラムを探し出し、試行するための良い手段が要求されている。

(2) CALGO において

CALGO（アルゴリズム情報データベース）には、計算アルゴリズムの言語情報と、ソースプログラムが収集されている。ここでも検索して得られたアルゴリズムをその場で試行する機能が求められている。

また、例えばデータベース構築・維持のためのユティリティ群のように、データベースによる管理・運用が有効と思われるプログラム群がある。

これには、処理系が複数のプログラムで構成される場合、とくにフレームメーカ以外から提供されている場合な

どがある。

ユーザレベルでも全く同じ状況がある。専門分野において，理論がプログラム集合で表現されていることがある。例えば、制御工学において，制御系の設計論を表現するプログラム群(CADE)がある。1つのプログラムは1つの処理機能に対応している。問題解決のためには、これらの処理機能(プログラム)を組み合わせ、一定の順序で実行することにより解を得る。ここで1つ1つのプログラムは「レベル0関数」，その順序つき組合せは「レベル1関数」と呼ばれる。レベル1関数は，レベル0～レベル(n-1)までの関数の順序つき組合せである。

計算機を利用した問題解決の過程は次のように図式化できる（図1.）。

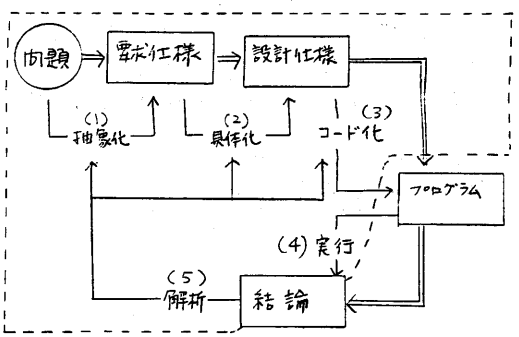

図1. 問題解決過程

(1) 問題を抽象化して要求仕様を明確にするステップである。ここでは，問題領域における専門知識と、問題の正確な把握が必要である。

(2) 要求仕様を具体化して計算機処理のための設計仕様にするステップである。ここでは計算機の側の知識（実現可能性，能率，……）が必要である。

(3) 設計仕様に基づきプログラムを作成するステップである。メソッド・ベースでは，レベル1関数の合成に当る。

(4) プログラムを実行し、結論を出す。

(5) 結論を解析し、不満足であれば(1)～(3)のステップに戻り，再実行する。

ここで，設計仕様が充分に明確かつ具体的であれば，ステップ(3)は機械化できる。

制御器の設計では，目的に応じた制御が得られるまで、関数の構成と解析を試行錯誤的に行う。このような場合に，ステップ(2)と(5)を支援し、ステップ(3)と(4)を自動的に行うことが，メソッド・ベースのねらいである。

制御系の設計のためのプログラム群を対象としてプロトタイプを開発する。

3. システムの概要

システム概念図を図2.に示す。基本機能を問題解決過程にそって述べる。

(1) 要求仕様の作成に関して

知識工学，エキスパートシステムの研究対象として興味深いが、ここでは全く触れない。ただ要求仕様記述は設計仕様記述、レベル1関数と関連づけて保存し、参考データとする。従って要求仕様は、目的を簡潔に自然言語で表現して設計仕様，レベル1関数の「標題」として位置づける。

(2) 設計仕様の作成に関して

要求仕様をもとに，利用者がモジュール情報，過去に実行され保存されているレベル1関数等を検索・参照しながら作成するのを支援する。

要求仕様実現のために必要な、主なモジュール名とその実行順序を決める。記法は、モジュール名，およびレベル1関数の並びと，簡単なIF～Then規則による。

(3) プログラムの合成に関して

設計仕様を解釈し、モジュール情報、モジュール制御情報とから実際に実行するモジュールの並びをできる限り自動的に生成する。もちろん、利用者との会話による修正は受け付ける。生成したモジュールの並び（レベルの関数）を関数ファイルに格納する。結果が検証されるまで、これを繰返し修正する。

ここでは，モジュール情報，モジュール制御情報のデータ構造と表現方法が重要な意味を持つ。

(4) プログラムの実行について

関数ファイルに格納された関数を，プログラムの実行制御に関する手続き的知識を参照しながら実行していく。

ここでは，モジュール情報，モジュール制御情報のデータ構造を，応用分野や使用言語に依存しないで一般的に決めることができ，それでモジュールの性質，実行上の条件等を必要十分に表現することができれば、内容に依存しないで属性のとり扱いに関する手続きの規則が一般的に記述できるであろうとの前提に立っている。

関数に従って次々とモジュールを実行することが不可能になった場合や、関数内にブレーク・ポイントが設定されている場合は，端末を介して利用者と会話し，以後の動作を決める。このとき、その時点までのモジュールの実行状況の参照や，関数の修正が可能である。

(5) 解析.

利用者が結果を解析し、ステップ(1)、(2)，(3)のいずれかに立ち戻り、修正して再試行するのを支援する。

(6) モジュール登録機能

同じ処理機能のプログラムも、新たな手法や計算法により作成される可能性がある。プログラムの登録は、

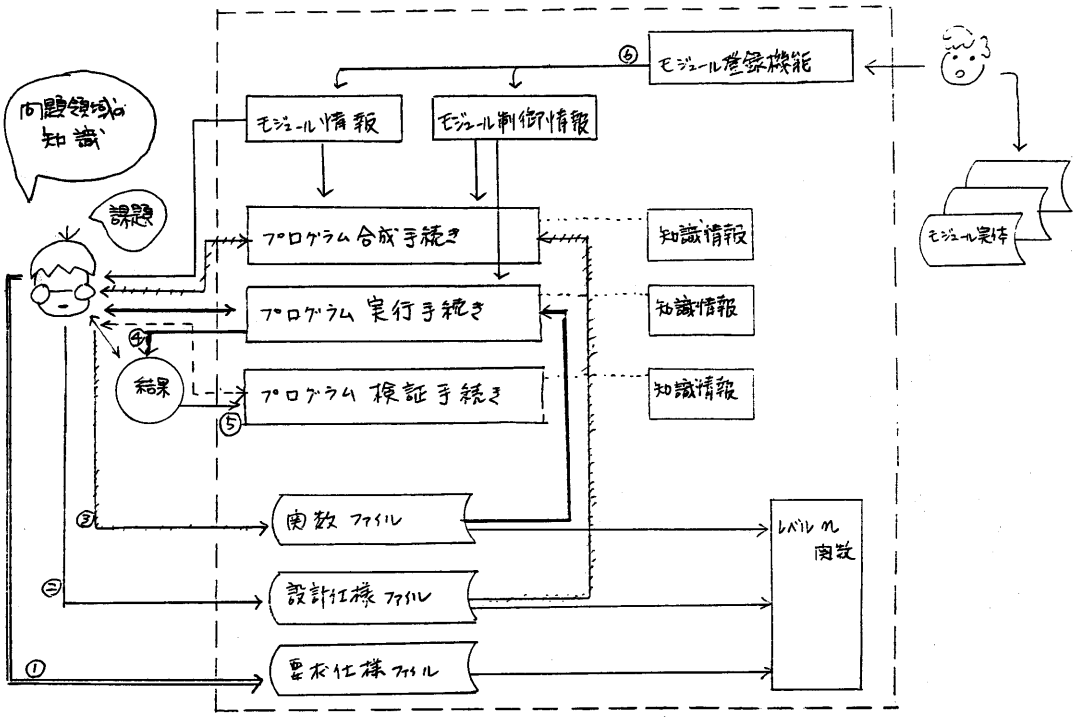

図2. メソッド・ベース概念図

プログラムのモジュール情報，モジュール制御情報を端末から受取る。モジュール実体，ソースプログラム，コンパイル&リンクの制御文は，順編成データセット上に置く。

モジュールの削除，モジュール実体の修正，モジュール情報・モジュール制御情報の追加・修正も行う。しかしこれらの機能は，(1)～(5)の問題解決過程では動作しない。問題解決は，登録されている情報，モジュールの範囲内で行う。

(7) 情報獲得機能

メソッドベースが，専門家によって設計・研究過程における試行錯誤，プログラムによる実験の手段として使用されたとき，結果として得られる{要求仕様，設計仕様，関数}の組は，専門分野における問題解決に関する知識である。また，モジュールの組合せが新たな処理機能を表わすことがある。これらは，実行結果の一部として一時的に保持されている中から，とりおしてメソッドベースにとりこむ。

4. データおよび知識表現

メソッドベースに蓄積され，利用される情報を次のように分類してきた。

(1) プログラム・モジュールに関する情報
(2) 〃 の実行に関する情報
(3) プログラムの実行に関する一般的手続き

(この他に，問題領域における専門的知識が利用に伴って蓄積される。)

(1)，(2)の情報は，ほとんどが，属性と値の組で表現できる。その一部を図3.に示す。(次頁)

(3)は，(2)の情報の属性間の関係をとり扱う一種の規則の形で表現することができる。たとえば，完了コード(cc_m)と対応する処理方法(p_m)があれば終了時の完了コード CC と一致する cc_m に対応する p_m を実行してやればよい。CC が実際にどんな値であったかは意識する必要がない。CC=0以外で，CC=cc_m なる cc_m が定義されていない場合は，関数の実行を中断し，端末利用者に指示を求める。

このような形で実行に関する手続きがすべて記述できるように(1)，(2)を定めることが重要である。

5. おわりに

北大大型計算機センターで稼動している汎用DBMS ADABAS上に，このメソッドベースを構築することを予定している。ユーザ言語NATURALから，他言語(FORTRAN等)のプログラムを起動し，終了後NATURALに制御が戻る機能を利用する。

当メソッド・ベースの研究は，まだ緒についたばかりである。まず，データ構造とそれを扱う手続きの表現形式を明らかにしていく予定である。

<参考文献>

1, 喜多；「制御系に関する高精度・大容量CADシステムの研究開発

2, 千葉，伊藤；「プログラムライブラリ管理データベースとユーザインタフェイス」，北大大型計算機センターテクニカルレポートNO.8，

	属性	値	注
プログラム・モジュールに関する情報	プログラム名	プログラム名称	
	作成者氏名	作成者名	
	キー・ワード	キー・ワード のリスト	
	利用説明書 標題		
	〃 著者		
	運用バージョン番号	運用中のバージョン番号	
	バージョン番号	001 ～	
	日付	YY. MM. DD	
	リースプログラム所在	データセット名（メンバ名）	
	オブジェクト 〃	データセット名（メンバ名）	
	コンパイラ制御文 〃	データセット名（メンバ名）	
	コメント種類	D：制限事項，U：修正個所，…	
	コメント・テキスト		
	プログラム機能	機能説明	
	手法・解法 アルゴリズム	手法，解法，アルゴリズムの名称	
	参考文献 標題		
	〃 著者		
	〃 出典		
	応用分野	応用分野名	
	精度	倍精度，…	
	記述言語	FORTRAN，C，…	
	使用条件	「論文に明記せよ」云々…	
	使用制限	～場合に限り有効である，云々	
	関連プログラム名		
プログラム・モジュールの実行に関する情報	呼出し方法		
	引数名	引数名 または 順序番号	
	タイプと長さ	実号，数号，バイナリ，…	
	標準値		
	最大値		
	最小値		
	入力データ	データセット名 または 端末	
	割付け方法	ジョブ制御文 または 不用	
	出力データ	データセット名 または 端末	
	割付け方法	ジョブ制御文 または 不用	
	完了コード	0 ～ 255	
	処理	NEXT / 実行モジュール名 / ABNORMAL END	
	前処理条件		
	処理	モジュール名	
	後処理条件		
	処理	モジュール名	
	呼出し元プログラム	モジュール名 のリスト	
	呼出し先プログラム	〃	

図3. データ構造（一部）