ソフトウェア工学 40－1
（１９８５．２．７）

MOTHER SYSTEM によるプログラム製造能力

西村　高志
（日本電気㈱　ソフトウェア生産技術研究所）

１．まえがき

MOTHER SYSTEM は情報処理振興事業協会（ＩＰＡ）技術センターにおいて、プログラム合成が技術的に可能であることを示し、さらに合成技術をベースとしたソフトウェア製造形態がどのようになるかを確かめるために、対象を事務処理業務の限定された分野に絞って開発した実験システムである。プログラム合成系を中核とした、プログラム仕様部品からのソフトウェア製造を特徴としている。

開発した実験システムについては、次の２種類のプログラムを製造実験して、プログラムの外部仕様に位置付けている３宣言から、単体で実行可能なプログラムが合成できることを確認している。

・　合成する処理の典型例である「給与明細データ作成プログラム」と在庫管理をモデル化した「酒類販売会社倉庫受付係プログラム」[8]。

・　既存の業務ソフトウェアである「マスタ・メィンテナンス・システム」。

これまでに、以下のことを報告している。

・　対象分野に撰択した事務処理業務の特性とそこでのプログラムの位置付け[1]。

・　既存プログラムを再利用する上での阻害要因についての考察[2]。

・　概念設計結果をまとめた部品化方式[3]。

・　プログラム合成方式[4,7]。

・　開発した実験システムのシステム構成[5]。

・　プログラム設計方法と製造形態[7]。

・　モデル・プログラムの入力仕様リストと合成されたプログラム・ソース・リスト[4,6]。

ここでは次の３点について報告する。

・　MOTHER SYSTEM がベースとしているプログラム・モデル。

・　評価実験データと評価実験よりわかった既存プログラムとの相違。

・　まとめとして、MOTHER SYSTEM の位置付け。

２．プログラム・モデル

MOTHER SYSTEM では、プログラム合成技術とソフトウェア部品とを組み合わせている。それぞれの基本方針をまず説明し、次にプログラム・モデルの骨格を、そしてモデルの追加部分を説明する。

２．１　基本方針

プログラム合成技術については、単体で実行可能なプログラムを、そのプログラムの外部仕様である入出力データについての表現のみから合成することを目標とした。このために関数型と非手続き型の２つの既存モデルを利用した。

関数型モデルは、基本的には電子回路の組み合わせ回路である。ひとつの回路について、一定のタイミングごとに一群の入力がその回路に入り、その一群の入力に対して一群の出力が出て行く。

このモデルの特徴は、すべての出力値を入力値を要素とする関数式で記述でき、さらにこの関数式が各タイミングごとで不変な点にある。

この組み合わせ回路のモデルでは、タイミングにまたがる記憶ができないので、あるタイミングで演算した値を次のタイミングでの演算の際に参照できるような順序回路にまで、モデルを拡張する。

非手続き型モデルは、事務業務処理を人手でやる際に使われるフォームシートである。給与計算を例に上げれば、各社員ごとに一枚のフォームシートがあり、社員データをまず記入して、次に計算できるものから順次計算していくと、最後には必要な項目値がすべて計算される。

このモデルの特徴は次の点にある。

・　各項目値は、それぞれの計算式で宣言される。

・　各項目値は、はじめは空で、記入あるいは計算によって値が決まったなら、値は変更されない。

・　各計算式は、必要な項目値がすべて決まったならその計算をする。計算実行順序は計算式中での各項目の参照関係により自動的に決まる。

ソフトウェア部品については、機械や自動車などハードウェアの部品からの類推で考えられる場合が多いが、ハードウェアがはっきりとした実体を持っているのに対して、ソフトウェアが、その実体は計算機上のビット列であり、仮想の実体を対象とすることから、部品についての考え方を変えている。

仕様からプログラムを合成できる技術に基づいて、次の点を特徴とする仕様記述の部品化を図る。

・　言語文法に合えば単体で実行するプログラムとなれる仕様言語の存在。

・　組み立ての道具としての合成系の存在。

・　各行間に相互作用も副作用もない仕様記述。

２．２　プログラム・モデルの骨格

モデルの骨格を決めた経緯を説明する。

プログラムの仕様として次の２つは必須である。

・　入出力のデータ宣言。

・　出力データ各項目値の演算式。

一方、COBOL プログラムは、見出し部とデータ宣言部、手続き部よりなり、さらに手続き部には、次の機能を行なう部分が混在している。

・　データ領域の初期化やデータ転送。

・　データの入出力。

・　各項目値の演算。

・　処理ロジックの制御。

各項目値を演算する部分を他から切り離して、残りの部分を入出力データのタイミング情報と入出力データ宣言とから、なんらかの方法で合成する。

事務処理プログラムでは、プログラム処理パターンが、一般には、媒体変換と分類、抽出、集計、分配、併合、照合、更新、報告書作成とに分類されている。このうちで、抽出と集計、分配、併合、照合、報告書データ作成とについては、１レコード入力１レコード出力の形態である。そこで、プログラムの処理ロジックを次のように規定する『一群の入力データ値が揃った時点で一群の出力データ値を演算する。この操作を全入力データ群について繰り返す』。入力データは、同一キーでソートされた主データ群と入力した当該主データ値により入力データが決められる関連データ群とに分ける。出力データは別に分けない。

上記の処理形態に対象を限定することにより、入出力データ構成（プログラム宣言）と入出力データ宣言（データ宣言）とから、演算部分以外のプログラムのすべての部分を合成できる。残った演算部分を合成するために、各項目値の演算式（手続き宣言）を設けた。

手続き宣言は、出力データの各項目値を入力データの項目値からの演算宣言で表現する構成とするが、演算中間値の格納用の一時データと記憶機能のために旧値と新値とを持つ記憶データとの２種類の入出力データに対応しない内部データを追加する。

演算宣言はフォームシート形式とする。

・　各行について、演算に必要な項目値がすべて決まったなら、その演算値を計算する。

・　各項目値は、ひとつの値のみを持ち、一回の操作中に値が変更されることはない。

・　各行の演算順序は、各項目の参照関係により自動的に決める。

各項目の演算式の書式を次のものとする。

項目ａ　＝　［　演算条件１　］　演算式１
　　［　演算条件２　］　演算式２
　　・
　　・

・　演算条件ｉが真ならば、演算式ｉの値を項目ａの値とする。

・　代入式ではなく等式である。

・　演算条件が真なるものが複数個あった場合には、どの演算式の値をその項目値とするかは非決定とする。非決定とした理由を適切に説明できないが、直感的には、厳密かつ形式的に記述することと現実の手続きを無理なく記述することとを調和させるためである。非決定性を入れたことにより、３つ組（項目、演算条件、演算式）間での相互作用と副作用とが禁止され、結果的には、演算部の合成が３つ組の並べ替えで済む。

２．３　プログラム・モデルの追加部分

完全なプログラムとするために次の機能をモデルの骨格に追加する。

① 出力条件というフィルターの追加

各出力データについて、出力条件というフィルタを設けて、この条件が真の場合のみ出力データを実際にファイルに出力する。これにより、各項目の演算をデータ出力条件から分離して宣言できる

② 入力データ項目値の値域検査機能

入力データの各項目値の値域を、利用者作成・登録の値域検査用外部手続き呼び出しの形式でデータ宣言中に宣言できる。値域外の場合は、入力データを読み飛ばすかプログラムを異常終了させる。

③ 領域初期化

真なる演算条件があれば、対応する演算式の値がその項目値となるが、真なる条件がなかった場合のために、すべての項目値を次のように初期化する。

- 入力データ項目：　該当データがなければ空。
- 出力データ項目と一時データ項目：　空。
- 記憶データ項目の新値：　旧値。

④ 条件関数の追加

該当入力データ有無を演算条件と出力条件中で参照できるように、有無を検査する２種類の条件関数を追加する。

⑤ 演算関数の追加

内部関数を作れないので、演算式中で参照できる、利用者作成・登録の外部手続きを追加する。

⑥ 一次元配列

一次元配列の定義機能と配列単位での配列要素操作のために２種類の特殊添字とを追加する。詳細は文献[7] で説明している。

実際の事務処理 COBOLプログラムを製造するために、さらに次のようにする。

- 主データと出力データとは順編成ファイル、関連データは順編成または索引編成ファイルとする。
- 表引きデータの定義・参照機能を追加する。
- 演算条件・出力条件と演算式とについては、合成系の負荷を少なくするために、 COBOLの条件式と COBOLの単項または COMPUTE文で書ける式とする。

３．　評価実験

既存業務ソフトウェアの製造実験結果を説明する。

３．１　評価実験概要

製造実験の対象システムは、製品情報の登録・管理と他システムへの製品情報提供を担当するシステム「マスタ・メィンテナンス・システム」である。

システムは、ソート・ユーティリテイ 5本と COBOLプログラム12本からなる。プログラムの内訳は、分配 1本、チェック 4本、更新 2本、抽出 3本、作表 2本で、プログラム規模は合計 8.5K行。

上記プログラム中で、MOTHER SYSTEM で合成できる分配とチェック、抽出とについて、分配 1本、チェック 3本、抽出 1本、合計 5本のプログラムを実際に作成した。

実験チームの構成は、実験管理を担当するリーダ（経験 9年）と製造を担当する女性プログラマ 3名（経験 3.3，1.3，0.3年）である。MOTHER SYSTEM とUNIXとの簡単な説明の後、既存の仕様書に基づいてプログラムを製造した。

３．２　評価結果

次の順序で評価実験結果を説明する。

- プログラム内部構成上の特徴。
- プログラム・モデル上での不足点。
- 定量評価データ。
- 定性評価結果。

３．２．１　プログラム内部構造上の特徴

MOTHER SYSTEMが合成するプログラムの内部構造を図-1に示す。

手続き部は次の部分からなっている。

- 一連の操作を繰り返す制御部
- データの入出力を行なう入力部と出力部
- 各項目値を演算する演算部
- ファイルのオープンとクローズ、１レコード入力、出力レコード初期化をそれぞれ分担する内部関数群。

```
↑  IDENTIFICATION  DIVISION.
|     見出し部
①
|  INPUT-OUTPUT  SECTION.
↓  FILE-CONTROL.
↑     SELECT
|        ファイル属性宣言
|
|  DATA  DIVISION.
②  FILE  SECTION.
|  FD
|        ファイル領域宣言
↓
↑  WORKING-STORAGE  SECTION.
③        内部変数宣言
↓
④        入力／出力／一時／記憶データ宣言
↓
⑤        値域関数／演算関数パラメータ領域宣言
↓
↑  PROCEDURE  DIVISION.
|        制御部
⑥        入力部
|        出力部
↓        内部関数群
↑
⑦        演算部
↓  END COBOL.
```

図-1:　MOTHER SYSTEM が合成するプログラムの内部構造

```
↑  IDENTIFICATION  DIVISION.
|     見出し部
①
|  INPUT-OUTPUT  SECTION.
↓  FILE-CONTROL.
↑     SELECT
|        ファイル属性宣言
|
②  DATA  DIVISION.
|  FILE  SECTION.
|  FD
|        ファイル領域宣言
↓
↑  WORKING-STORAGE  SECTION.
③④⑤
|        データ宣言
↓
↑  PROCEDURE  DIVISION.
|        メイン手続き
⑥⑦
|        内部手続き群
|
↓  END COBOL.
```

図-2:　既存プログラムの内部構造

演算部は、３つ組（項目、演算条件、演算式）に対応する（条件）代入文が分岐もなく上から下に実行する形態になっている。入出力データについては、ファイル領域宣言中とデータ宣言中とに二重に領域が取られる。プログラム言語は COBOL 77 である。

既存プログラムの内部構造を図-2に示す。

・　見出し部とファイル宣言のすべてとデータ宣言の主なものとはマクロ化されている。

・　手続き部は機能分割されており、メイン手続きと下位内部手続き群からなっている。

・　手続き部については、コメントである各手続きのヘッダを除いてマクロは使われていない。

・　プログラム言語は、COBOL/S(構造化 COBOL)である。

３．２．２　プログラム・モデル上での不足点

プログラム・モデル上での不足点を列挙する。

①　集団項目と再定義（ REDEFINES）属性とについては、データ宣言中で記述はできるが、手続き宣言中では、各演算宣言の３つ組を最下位項目単位での記述とし、さらに再定義属性を考慮していなかった。そのために既存のプログラムと比べて、演算式の量が増え、表現できない場合があり得る。合成時の３つ組並べ替えの際に、集団項目と再定義属性とを考慮したアルゴリズムに拡張する必要がある。

②　３つ組（項目、演算条件、演算式）を互いに独立とし、さらに初期値が空か旧値のみに限定したために、演算条件が複雑になり、現実には記述できない場合がある。初期値として、空や旧値以外に、入力項目値を指定できる機能が必要である。

③　一次元配列については、添字を直接に指定する機能が要りそうである。

④　演算条件中で演算関数が使えない。これは合成系の負荷を少なくするための制限であったが、やはり、演算条件と演算式については、一般の数式にまで拡張する必要がある。

３．２．３　定量評価データ

次の４つの定量評価データ値を表-1に示す。

- MOTHER SYSTEM での仕様記述量。
- MOTHER SYSTEM で合成されたプログラム量。
- 既存プログラムでの記述量。
- 既存プログラムマクロ展開後のプログラム量。

３．２．４　定性評価結果

MOTHER SYSTEM でのプログラムと既存 COBOLプログラムとの相違点を列挙する。

① 既存プログラムが処理単位に機能階層に分割されているのに対して、プログラムが内部に階層構造を持たずに、一層の平坦構造になっている。機能階層構造を再利用上の阻害要因のひとつとする考察を文献[2] で述べたが、平坦構造では、記述が明確であり、情報も変に分散することもなく、変更も影響範囲が明白なのでやり易いように思える。

② 既存のプログラムが順序実行と条件分岐、繰り返しとの３制御構造からなっているのに対して、

- 順序実行　−→　実行順序を意識しない。
- 条件分岐　−→　次のものに制限する。

  if　条件　then　項目：＝　演算式
- 繰り返し　−→　一次元配列の要素すべてに対する集団操作に制限す

演算宣言が展開された部分は、一次元配列集団操作部分に対する、演算宣言一文に対応する展開部分中での繰り返しを除いて、繰り返しも分岐もなく、（条件）代入文が一列に並んでいて、この文列が上から下へ実行される。

③ 演算宣言について、既存のプログラミング言語のような代入式ではなく、等式にしている。

④ 内部処理のアルゴリズムはデータ構成からほぼ決まってしまう。後は、入出力データ値ぐらいを考えながら記述すれば、プログラム仕様が完成する。

⑤ ３つ組（項目、演算条件、演算式）対を互いに独立にしたために、演算条件が重複する部分が増えて、長くなる。

⑥ すべての変数は値が代入されるか、あるいは初期化されてから参照される。

表-1:　定量評価データ値

【MOTHER SYSTEM での仕様記述量】

	分配	チェック１	チェック２	チェック３	抽出
・プログラム宣言	1	1	1	1	1
・データ数					
入力データ	1	1	1	1	2
出力データ	3	2	2	2	2
一時データ	0	1	1	1	1
記憶データ	0	0	0	0	1
表引きデータ	0	1	1	1	0
・各データのデータ宣言行数−− ( ) 内は最下位項目数−−					
入力データ	7(5)	89(38)	62(42)	65(27)	20(13)
					6(3)
出力データ	7(5)	81(33)	58(39)	7(5)	20(13)
	7(5)	79(33)	108(80)	6(6)	1(1)
	4(4)				
一時データ		5(4)	8(7)	15(9)	4(4)
記憶データ					14(8)
表引きデータ		64(62)	64(62)	44(42)	
（合計）	25(19)	318(170)	300(230)	137(89)	65(42)
・各データの手続き宣言中の３つ組数−− ( ) 内は対象項目数−−					
出力データ	5(5)	14(4)	43(34)	6(5)	11(11)
	5(5)	12(11)	71(63)	8(6)	1(1)
	4(4)				
一時データ		39(4)	40(7)	20(8)	5(4)
記憶データ					4(4)
（合計）	14(14)	65(19)	154(104)	34(19)	21(20)

【MOTHER SYSTEM で合成されたプログラム量】−−コメント文はない−−

	分配	チェック１	チェック２	チェック３	抽出
・見出し部①	8	8	8	8	8
・データ宣言部					
ファイル宣言②	43	261	242	85	65
内部変数宣言③	20	53	53	25	43
データ宣言④	29	322	306	144	107
パラメータ領域⑤	10	15	12	12	15
（小計）	102	651	613	266	230
・手続き部					
制御部／入出力部／内部関数群⑥	83	72	72	74	137
演算部⑦	17	323	373	113	129
（小計）	100	395	445	187	266
・　総計	210	1054	1066	461	504

【既存プログラムでの記述量】−−コメント文を除く−−

	分配'	チェック１'	チェック２'	チェック３'	抽出'
・見出し部①	1	1	1	1	1
・データ宣言部					
ファイル宣言②	14	12	15	10	17
データ宣言③④⑤	34	31	93	29	18
（小計）	48	43	108	39	35
・手続き部⑥⑦	87	277	443	211	168
・　総計	136	321	552	251	204
・マクロ数					
見出し部①	1	1	1	1	1
ファイル宣言②	11	8	9	9	8
データ宣言③④⑤	4	5	6	5	1
（合計）	16	14	16	15	10
・階層構造					
手続き総数	6	10	11	9	12

【既存プログラムのマクロ展開後のプログラム量】−−コメント文を除く−−

	分配'	チェック１'	チェック２'	チェック３'	抽出'
・見出し部①	11	11	11	11	11
・データ宣言部					
ファイル宣言②	58	116	149	95	116
データ宣言③④⑤	67	144	264	133	28
（小計）	125	260	413	228	144
・手続き部⑥⑦	87	277	443	211	168
・　総計	223	548	867	450	323

3．3　総合評価

プログラム合成については、既存の仕様書の機能をほぼ満たすプログラムを、プログラム仕様に位置付けた3宣言から合成できた。

プログラム仕様記述量については、データ宣言部は内部変数宣言がない分少ない。手続き部は演算宣言3つ組をCOBOL 2行に換算すると半分程度となる。

合成されたプログラム量については、データ宣言部は、入出力データを二重の領域に取った分大きくなっている。手続き部は、プログラムによって異なるが、1，2割大きくなっている。

実行速度については、現在の最適化を行なっていないものでは、2，3倍になると思われる。

プログラム・モデルについては、不足点を補えば、対象に撰択した分野では、ほぼ十分と思える。

残りの更新と報告書作成についても、技術的には同様の方式で合成系を作れる。

部品化については、時間がなくて、部品の作成・登録・再利用を実験できなかったが、部品ベースを作るに要する労力と得られる効果の比を考えると、データ宣言について参照できる程度の軽い部品ベースを備えた、ソフトウェア自動製造システムが現実的な方向に思える。

4．　MOTHER SYSTEM の位置付け

現在のプログラムの歴史は、プログラミング言語から大きな影響を受けている。そしてプログラミング言語は、機械語から高級言語と発展してきたが、計算機の命令を基本にしているように思える。現状の言語を改良するにしても、異なる土台の上にプログラムを構築し直すとしても、たぶんより論理的土台に基づいたモデルが必要であろう。

MOTHER SYSTEM では、関数型と非手続き型の両面を持つモデルをベースとして、プログラム合成技術を中核とした実験システムを開発し、そこでのプログラム設計方法と製造形態を示し、さらに既存の業務ソフトウェアが製造できることを示した。

MOTHER SYSTEM は、今後のプログラムを考える上でのひとつのモデルに位置付けられよう。

5．　むすび

MOTHER SYSTEM については、これまでにプログラムの設計方法と製造形態と合成技術とについて説明していた。ここでは、関数型と非手続き型との両面を持つプログラム・モデルと既存業務ソフトウェアを製造実験してわかった既存プログラムとの相違点とを説明した。

謝　辞

評価実験チームの構成は日本電気情報サービス㈱の佐藤平和氏と有賀恵美子さん、石井景子さん、細矢由佳理さんとの4名である。

システム開発は、技術センターの廣道博史研究員（現在は日本電気ソフトウェア㈱）と日本電気情報サービス㈱の藤野博之氏，斎藤明久氏，石井景子さんと日本電気技術情報システム開発㈱の斎藤実氏，田口安男氏との6人が行なった．

コンサルタント委員会委員及びワーキング委員会委員からは，多数の御意見をいただいた．

御協力いただいた方々に深謝する．

参考文献

[1] 西村：表現された知識として見たソフトウェア．第24回プログラミング・シンポジウム，報告集pp.39-47（1983.1月）．

[2] 西村，廣道：ソフトウェアの再利用における阻害要因の分析，情報処理学会ソフトウェア工学研究会資料28-4（1983.2月）．

[3] 西村：事務処理業務ソフトウェアの部品分割方式，情報処理学会ソフトウェア工学研究会資料30-3（1983,6月）．

[4] 西村：MOTHER SYSTEM によるソフトウェアの設計と製造，情報処理学会「プログラム設計技法の実用化と発展」シンポジウム，論文集pp.103-112（1984,4月）．

[5] 西村，廣道，藤野，斎藤明久，斎藤実，田口：仕様部品からのプログラム合成システムMOTHER SYSTEM の構成，情報処理学会ソフトウェア工学研究会資料35-3（1984,5月）．

[6] 西村，廣道：MOTHER SYSTEM:事務処理 COBOLプログラム合成システム，情報処理振興事業協会，技術センター第3回発表会，資料集pp.221-280（1984，10月）．

[7] 西村：MOTHER SYSTEM によるソフトウェアの設計と製造，情報処理，Vol.25，No.11，pp.1247-1254（1984,11月）．

[8] 山崎：共通問題によるプログラム設計技法解説，情報処理，Vol.25，No.9，p.934（1984，9月）．