ICOT Technical Memorandum: TM-0477

TM-0477

並列プログラムのプロトタイピング

松本一教，内平直志，本位田真一

March, 1988

©1988, ICOT

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

並列プログラムのプロトタイピング

Prototyping System for Concurrent Program

松本一教，内平直志，本位田真一

Kazunori MATSUMOTO, Naoshi UCHIHIRA,and Shinichi HONIDEN

（株）東芝　　システム・ソフトウェア技術研究所

Systems & Software Engineering Lab., Toshiba Corporation

In this paper, prototyping system for concurrent programs based on software components reusing is described. In the system, we must give following two specifications as program specifications,
(1) components specification and
(2) scheduling specification
The first one is used for retreival and interconnection of components stored in a library. And the second one is used for generating a scheduling program which synchronize components communications. Moreover, some methods which increase the efficiency of retrieval and interconnection are described.

はじめに

近年，ソフトウェアの規模が増大してくるに伴い，その品質を保証する技術が重要視されるようになってきている[1]．さらに，ウォータフォール形のソフトウェアライフサイクルモデルに対する批判がなされるようになり，プロトタイピングの手法が注目を集めるようになってきた．このような状況をふまえ，筆者らは部品の再利用とスケジューラプログラムの自動生成を行う並列プログラムのプロトタイピングシステムの提案を行ってきた[2,3]．それは，オブジェクト指向言語MENDEL[4]により作成されたプログラムを，部品として蓄積しておき，新たなプログラムの作成は，蓄積されている部品の中から適当なものを幾つか検索し，それらを結合することによって行う手法である．また，各部品間の同期を取るためのスケジューラを，本質的にはWolper[5]の手法に基づき，時制命題論理で与えられた仕様から，定理証証明器を利用して，自動的に生成する．本論文では，これまでの提案に基づいて試作中のプロトタイピングシステムについて報告する．

1．　並列プログラムにおける部品化および再利用

部品の再利用によるプログラムの合成を行う場合，部品が再利用しやすい形になっていることが望まれる．そのためには，オブジェクト指向に基づく言語を用いることが有効であることが指摘されている[6]．また，各部品間の同期をとる機構をいかに実現するかについても，部品再利用との関係が深い．本研究で用いたMENDEL（例1）では，並列に動きうる単位はオブジェクト（部品）であり，同期の機構は各オブジェクト間のメッセージ伝達を制御することにより実現される．図1は，本システムにおける部品と同期部の関係を模式的に示したものである．

2．　システムの概要

図2に，本システムによるプロトタイピングの流れと，システムの構成について示す．本章では，この図に基づ

いて各段階での処理について説明する．

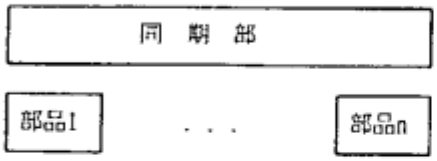

図1． 部品と同期部

2．1 MENDELによる部品の作成と蓄積

プログラマが作成したプログラムは，プログラム全体だけでなく，その各オブジェクトも部品として部品ライブラリに蓄積される．と同時に，部品仕様ファイルと呼ぶファイルに部品の入力属性名と出力属性名を記録し，さらに，プログラマが部品の仕様を自然言語（英語）で与えれば，それを1つの文字列と見做して，キーワードに分解したものもこのファイルに記録される．キーワードの抽出法については，いくつかの不用語を設定しておき，その除去を行った後に単数形や原形に変形するという手法を用いている[7]．

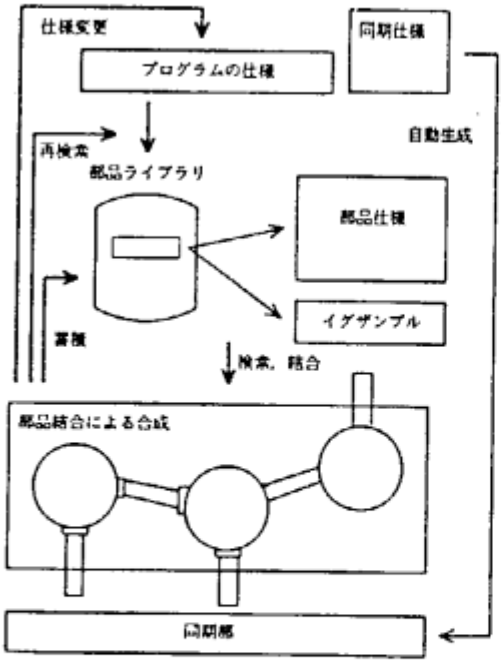

図2． システムの概要と構成

2．2 プログラムの仕様

作成すべきプログラムの仕様記述方法については，述語論理式[8]によるもの，自然言語[9]によるもの，例示によるもの[10]などが研究されている．本システムでは，プログラムに与える入力の属性名と最終的な出力の属性名を仕様として与えることにした．さらに，次節で述べる属性マッチングによる部品検索の効率を改善するためやユーザへの質問回数を減少させるために，例示及び自然言語による仕様記述を併用することもできるようにした．例2に，キーワードカウントプログラムに対する仕様記述の例を示す．

例2． キーワードカウントプログラムの仕様

```
input := keyword.stream ;
output := summary_report ;

spec := Keyword count program
        needs 3 components.
(1) read data from a input
    stream and divide them
    into words.
(2) check a word whether it
    is a keywordor not.
(3) output a summary_report
```

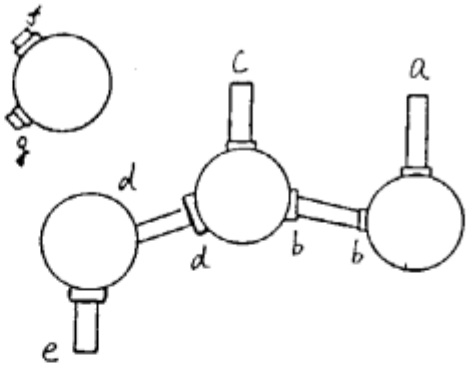

図3． 属性マッチングによる検索と結合

2．3 部品の検索と結合

仕様が与えられると，部品ライブラリからいくつかの部品を検索し，それらを組み合わせることで求めるプログラムを実現する．これは，属性マッチングとよぶ手法によって検索と結合を同時に行う．即ち，与えられた入力属性名から出発して，与えられた出力属性名にたどり着くまで，同じ属性名を持つ部品を検索し，結合する（図3）．このとき，ライブラリの全部品に対して属性マッチングを試みるより，ある程度部品の個数を減らしておき，それらに対してマッチングを試みることにすれ

ば検索の効率は向上する．そこで，前節で述べた自然言語による仕様が与えられていれば，部品ライブラリの一部をまず検索しておき，その結果に対して，属性マッチングを行う．このとき，情報検索の手法としてよく用いられる，シソーラスによる検索[7] を行うことで，より柔軟な部品の選択が可能である．

結合に成功すれば，実際にプログラムを動作させてみて，もとめるプログラムであるかどうかを評価する．もし，要求と異なったものであればPrologのバックトラックと同様に，属性マッチングによる他の解を求め，再度評価を要求する．このとき，ユーザへの質問回数をできるかぎり減少させることが望ましい．そこで，プログラムの検証に使用した，テストデータとそのときの出力を全て，<(入力属性名1，データ1),....,(入力属性名n，データn),(出力属性名1，出力1),....,(出力属性名n，出力n)，評価値>なる形で，イグザンプルファイルに蓄積しておく．ここに，評価値はプログラムのテスト結果を，要求を満たす場合をT，そうでない場合をFとして表わしたものである．部品の再検索，再結合を行う場合には，イグザンプルファイルに格納されている各イグザンプルに対し，評価値がTのデータに対しては同一の出力を生成し，Fのデータに対しては異なる出力を生成するような結合のみ，判断をユーザに問う．

2．4　同期部の自動生成

各部品（オブジェクト）間の同期は，メッセージ伝達を制御することで実現する．即ち，オブジェクト間は，途中にゲートのあるパイプで結合され，いったん途中に設けられたキューに格納される．同期部によりゲートが開かれるとキューから1つメッセージが取り出され，相手のオブジェクトへ送信される．

例3にキーワードカウトプログラムで用いる同期部の仕様を示す．ここでは，命題g が真なることを，ゲートg を開くという意味に対応付け，[](常に)，<>(いつかは)，○(次に)，P#Q(Q が成立するまではP が成立)という時制オペレータを用いて，ゲート開閉の時間的制約条件を記述している．論理式として仕様が与えられると，次にタブロー法[5] を用いた定理証明器によりその証明が試みられる．そして，もし充足可能であれば，空でない状態遷移図が生成される．これは，論理式を成立させる命題記号への全ての割り当てをグラフとして表現したものである[3,5]．同期部は，この状態遷移図から，可能な遷移を選択する機構として実現されている．このとき，フェアーネスの条件[11]を満たすために複数の可能な遷移がある場合，過去に選ばれてから最も時間の経過している遷移を行うことにする．

例3．　キーワードカウントプログラムの同期仕様

```
FINITE(g1), FINITE(g3)
¯g2 # eog3
OPEN(g4)
<>g5
[](g5 => @halt)
HALT
```

2．5　プログラムの蓄積

部品の再利用により，求めるプログラムが作成できればそれを新たな部品としてライブラリに蓄積し（図4），他のプログラムの部品として再利用することができる．このとき，入出力属性名や自然言語や例示として与えられていたプログラムの仕様も，部品仕様ファイルに格納される（図4）．

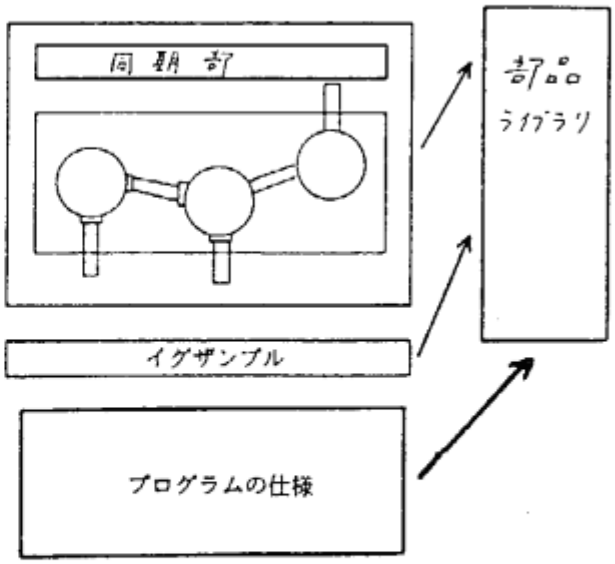

図4．　プログラムの蓄積

おわりに

MENDELのオブジェクトを部品として再利用するソフトウェアプロトタイピングシステムについて述べた．本システムでは，プログラムの作成と修正，さらに仕様の変更に伴うプログラムの再作成が全て同一の環境でなされることを特長とする．

現在，一部についての試作を終了し，文字列処理の分野に限定して，本システムの試用と評価を行っている．今後，さらに研究および開発を続ける予定であるが，当面の問題点として，

（1） 時制命題論理式は，記述力が十分でない．また，読解性に欠ける．

（2） 部品の再検索，再結合を行う場合，実行データをさらに有効に利用する必要がある．

などが明らかになった．

また，ＭＥＮＤＥＬについてもプロトタイピングという目的により適したものになるように改良を行っていく予定である．最後に，ＭＥＮＤＥＬオブジェクトの例を示しておく．

謝辞

本研究は，第５世代コンピュータプロジェクトの一環として行われた．研究の機会を与えて下さった新世代コンピュータ技術開発機構の方々に感謝する．

例1． ＭＥＮＤＥＬオブジェクト

```
WordCut
(
dec :(
        input(stream).
        output(word).
        state(buf![]).
    )
meth :(
    method(stream?' '.buf?[]).
    method(stream?' '.buf?W.buf![].word!W).
    method(stream?C.buf?W.buf![C|W]).
    )
junk:()
)
```

参考文献

[1] 本位田，松本　ほか：リアルタイムシステムにおけるプロトタイピングの一手法，情報処理学会論文誌，Vol. 26, No. 5.

[2] 内平、本位田　：　時制命題論理を用いた部品からの並列プログラム合成、日本ソフトウェア科学会第３回大会論文集

[3] Honiden, Uchihira : MENDEL Prolog Based Concurrent Object Oriented Language. Proc. of the COMPCON '86.

[4] 本位田，内平　ほか　：推論型システム記述言語MENDEL，情報処理学会論文誌，Vol. 27, No.2.

[5] Wolper : Synthesis of Communication Process from Temporal Logic Specification, Ph.D thesis Stanford Univ.

[6] 内平，関　ほか：MENDEL における並列プログラムの部品結合．　ソフトウェア工学研究会資料，　86.

[7] 有川　ほか：　ＭＩＲ－ＲＦ情報検索システム，情報処理学会１６回大会予稿集

[8] 吉田　ほか：　一階述語論理を用いたソフトウェアモジュールの機能検索．Proc. of the LPC '85.

[9] 市川　ほか：　自然言語に基づく静的システムの仕様のプロトタイププログラムへの変換手法，情報処理学会論文誌，　Vol. 27, No. 11.

[10] E. Shapiro : Algorithmic Program Debugging, MIT Press.

[11] L. Lamport : Sometimes is Sometimes Not Never Proc. of the 7th ACM Sympo. Programming Language.