オブジェクト指向による再利用部品の記述言語

3G−5

田中立二　川村敏和

(株)東芝　重電技術研究所

1．はじめに

ソフトウェア開発の生産性向上は重要な課題である。生産性向上の手段として、既存ソフトウェアの部品化・再利用が提唱されている。しかし再利用をプログラマが手作業で行うのでは効率化に限界があり、計算機による再利用支援環境の構築が望まれている。現在我々は、プロセス制御システムのソフトウェアをモデルに、以下のテーマでプログラム自動生成システムの研究を進めている[1]。

・記述言語の開発
・仕様からデータ宣言ソースの自動生成
・ソフト部品の再利用によるプログラム合成

本稿では、そのうち記述言語体系を中心に述べる。

2．部品再利用の問題点

一般に部品の再利用は下記のステップで行われる。

1．検索　2．理解　3．修正　4．試験

それぞれのステップでの主な問題点を列挙すると下記のようになるが、これら以外にも多くの問題がある。記述言語はこれらの問題を考慮して検討する必要がある。

（1）検索
・仕様に最も適した部品の検索が難しい
・検索手段（キーワードなど）の整備が必要
・複雑な関係を持った部品の検索支援がない

（2）理解
・ソフトウェアを理解・修正するのは容易ではない
・必要な資料が簡単に得られない
・部品作成者と再利用者で能力や知識の格差が大きい

（3）修正
・修正時に誤りが入る可能性が高い

（4）試験
・修正による品質の低下の度合いが定量的にわからない

3．部品化・再利用支援技術の現状

部品化では、いかに部品に汎用性を持たせるかが重要なポイントになる。プログラミング言語レベルでは、C++やSmallTalkなどのオブジェクト指向言語が注目されている。しかし開発体制の移行コストや大量の既存ソフトの扱いなどの問題がある。

またソフトウェア開発では、単にソースコードのみでなく、要求仕様書、外部仕様書、内部仕様書、試験仕様書、マニュアルなど大量の文書が作成される。これらの文書は多くの情報を含んでいるが、従来の文書媒体は紙中心であるため、情報の検索や再利用が容易でなく、結果的にソース・コード偏重の再利用形態になりがちである。最近ではマルチメディア技術やハイパーテキスト技術の進歩によって、文書を計算機で容易に扱える環境が整ってきた。またオブジェクト指向データベースなどの新しいデータベースにより、ソフトウェアの持つ複雑な構造を管理することが可能になってきた。そのためにはまず、既存部品を、再利用支援システムが扱い易いオブジェクトとして定義する必要がある。

一方再利用部品の変更・修正に関しては、現状のプログラミング言語では、カスタマイズに関する記述が十分できない。例えばパラメータやマクロによる記述は自由度が少ない。

4．本システムの概要

ハイパーテキストとマルチメディア技術を使用すれば、相互に関連する部品の検索や、必要なソースや資料の検索を効率化することができる。HyperCardを使ったプロトタイプによって検索や理解を支援できることを実験的に確認した。また部品オブジェクトの管理にはオブジェクト指向データベースを利用する。既存部品をオブジェクト化するために、我々は今まで述べたような問題点を考慮して、記述言語を下記の3階層とした。

・プログラミング言語
・部品記述言語
・仕様記述言語

5．記述言語体系

5．1　プログラミング言語

プログラミング言語は、CやFORTRAN、C++などの言語であり、処理シーケンスやデータを記述する。このレベルではコンパイラやOSなどの計算機環境に強く依存する。

Object Oriented Language for Software Package Description
Tanaka Tatsuzi, Toshikazu Kawamura
TOSHIBA CORPORATION

5．2　部品記述言語

部品記述言語はプログラミング言語と仕様記述言語の中間に位置する。部品記述言語の導入は

・部品ソースコードの隠ぺい
・部品（ソースや文書）と再利用手続きのカプセル化
・継承による部品の振舞いの共有と再利用
・プログラミング言語と仕様記述言語間の記述レベルの落差緩和と、独立性の確保

を目的としている。つまり既存部品を部品オブジェクト（クラス）として定義し、統一的に扱えるようにする。

例えば部品が持つ様々な部品属性として下記がある。

・計算機環境に関するもの
　（計算機の種類、ＯＳ、コンパイラなど）
・検索に関するもの（検索キー、関連情報のリンクなど）
・再利用に関するもの（使用上の制限、情報形態など）

再利用手続きでは、インスタンス生成や文書管理、試験などに関する記述を行う。再利用では部品のカスタマイズが必要になるが、システムは図１に示したカスタマイズ形態をサポートする。即ち、単純な複製、パラメータ変更による固定的な修正、オプション機能選択による修正、編集ルールに従った部分的な修正、最後はエディタによる自由な編集である。部品作成者は（最後の方法を除いて）これらを組み合わせて部品の生成手続き(NEW)を記述する。再利用者はインスタンス生成時に、必要な指定を行ってカスタマイズを行う。

またインスタンス生成した部品は、複数の手続きで共有するか、専用のローカル手続きとして再利用するかを選択する。

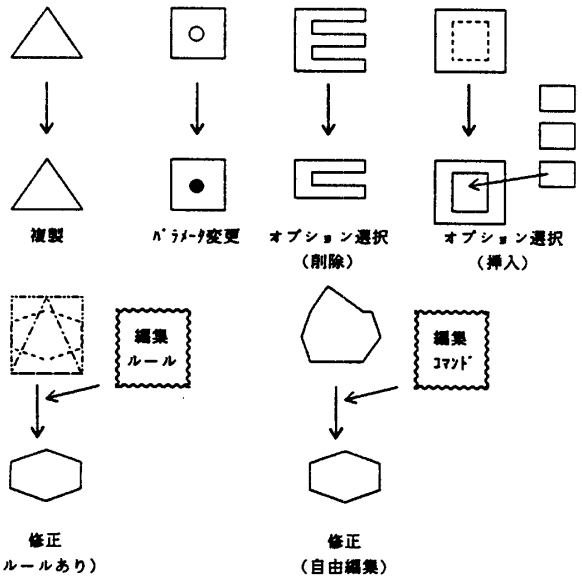

図１　部品カスタマイズの形態

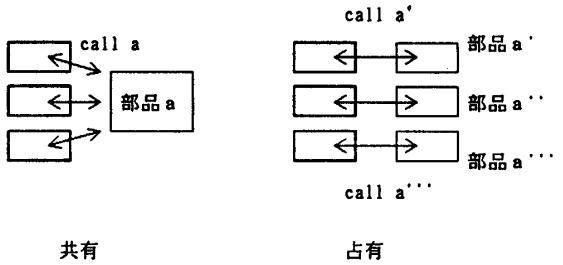

図２　部品インスタンス利用形態

5．3　仕様記述言語

仕様記述言語は一種のＰＯＬであり、対象とする分野に強く依存する。従って今回は詳細な検討をしなかったが、基本的にはオブジェクト指向言語の形式を取り、部品クラスのインスタンス生成、部品オブジェクトへのメッセージ送信形式で処理の記述を行う。必要な部品クラスがない場合は、直接ソース記述言語で処理を記述するか、新しく部品クラスを登録して使用する。

図３は仕様記述言語により、部品を組み合わせて処理を記述する場合の例である。点線内が仕様記述言語である。仕様記述言語で書かれたプログラムは、最終的にはＣ言語などのソース・コードに変換する。

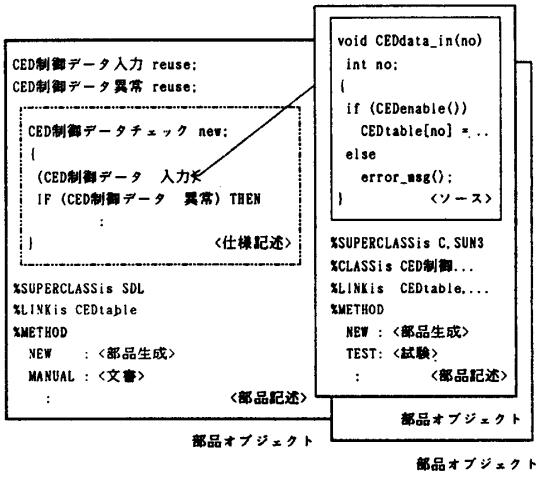

図３　仕様記述言語の記述例

５．まとめ

部品記述言語の検討を行った。記述言語を階層化することにより、部品のオブジェクト化、仕様記述言語の設計がより実際的になる。また仕様記述言語で書かれたソースも部品クラスとして定義することができるため、部品の集合で構成されるシステム全体も部品として扱うことができる。

参考文献

[1]川村，光武，飯村，"オブジェクト指向によるプログラム自動生成システムの構想"，情報処理学会第４０回全国大会講演論文集，１９９０．