既存ソフトウエアの再利用によるオブジェクト指向開発の支援環境の構築

佃 軍治* 団野博文* 永岡郁代**

* (株) 日立製作所 システム開発研究所
** (株) 日立製作所 ビジネスシステム開発センタ

保守・運用コストを削減するため、ホスト上のソフトウエア資産のＣＳＳ環境へのダウンサイジングが進行中である。ここでは、保守性・再利用性が高いとされるオブジェクト指向による再構築のニーズが高まっている。業務システムの再構築にあたっては、既存システム中の必須機能を網羅する必要がある。しかし、マニュアルや仕様書が既存システムを正確に反映していないため、これらからすべての必須機能を抽出することは困難である。我々は、既存システム中の設計情報を抽出・整理することにより、オブジェクト指向環境で利用するデータ構造（オブジェクトモデル）や各データ項目に対する処理ロジック（業務ルール）を作成する方法を検討し、支援ツールを開発した。また、作成した情報を用いてオブジェクト指向言語のコードを生成するツールも開発した。これらのツールにより、新システムを高品質に構築することが可能になると共に、再構築コストを削減することができる。

Support system for object-oriented development using existing software

Gunji Tsukuda* Hirofumi Danno* Ikuyo Nagaoka*

* Systems Development Laboratory, Hitachi, Ltd.
** Institute of Advanced Business System, Hitachi, Ltd.

To save maintenance cost, many company is re-building existing software in client-server system environment by object oriented development method. Re-built System should keep important data/function property of existing software. But using only manual and specifications of existing system, it is difficult to keep them in new system. Because in many case manual and specifications are not reflected correctly when exsiting source code is changed. So we study new method to extract important property from source code of existing system. We have developed support tools to make object model(data structure) and business rules which are re-used in object oriented environment from design information in existing system. And we have developed code generation tool for object oriented language from object model and business rules. By these tools, it is possible to build new hi-quality system and reduce re-building cost.

1．はじめに

既存ソフトウエアの保守／運用コストを削減するため、メインフレーム上の既存ソフト資産のCSS環境へのダウンサイジングが進行中である。一方で、（1）保守性、再利用性が高いこと、（2）PC／WSのOSやミドルウエアがオブジェクト指向インタフェースを有する傾向があること、（3）下流工程の開発支援環境が充実しつつあることから業務ソフトウエアに対するオブジェクト指向による開発支援、特に既存業務ソフトウエアに対するオブジェクト指向化技術への要求が高まっている。

既存ソフトウエアの再構築を行う場合、既存ソフトウエアのマニュアル／仕様書を要求仕様と見なし、これらからオブジェクトや機能を抽出して新規開発する方法が考えられる。しかし、既存ソフトウエアは、一般に幾度も機能拡張されており、マニュアル／仕様書をその都度修正しているとは限らない。このため、マニュアルなどに未記述のデータ項目や機能が存在し、マニュアル／仕様書のみに基づく再構築では、設計情報の抽出洩れが生じる可能性が高い。また、設計情報の抽出作業やオブジェクト指向環境での設計情報の再利用は、現在主に手作業で行われており、工数削減が望まれている。

そこで、我々は、マニュアル、仕様書に加え、既存ソフトウエアのソースコードを利用することにより、設計情報の抽出／再利用を網羅的かつ効率的に行う方法を検討し、既存ソフトウエアの再利用によるオブジェクト指向開発の支援環境を構築した。本稿では、既存ソフトウエアの再構築を支援する方法の概要（2章）、既存システムを反映した業務オブジェクトモデルの構築方式（3章）、各データ項目に対する処理ロジックの抽出方式（4章）、抽出情報を利用した実装支援方式（5章）、および支援環境の評価（6章）について述べる。

2．オブジェクト指向リエンジニアリングシステムの概要

既存ソフトウエアから設計情報を抽出し、オブジェクト指向開発に利用するアプローチとして、制御系システムを対象とした機能モデルを抽出する試み[1]がある。これは、プログラムから作成したデータフローを抽象化することにより実現しており、処理の流れが複雑な制御系システムには有効な方法である。

一方、事務処理系のソフトウエアの場合、データの設計が最重要であり、処理よりもデータを中心にして設計情報を抽出すべきである。そこで我々は、既存ソフトウエア中のデータ項目の構成関係や、転送関係、計算式などを利用して、クラスの構造やクラス間の関係を示すオブジェクトモデルの構築や、データ項目の値を導出する導出式やデータ項目の値の範囲を制限する値制約式である業務ルールの抽出を支援することを目的とした。また、構築／抽出したモデル／ルールを利用して実装工程のコストを削減することも目的としている。

上記の目的を達成するために構築したオブジェクト指向リエンジニアリングシステムは主に3つの機能から構成される（図1）。

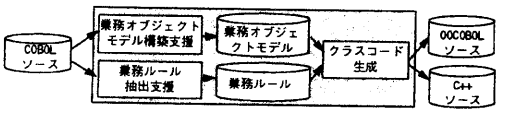

図1　オブジェクト指向リエンジニアリングシステムの機能構成

（1）業務オブジェクトモデル構築支援

既存ソフト中のレコード定義情報やデータ項目間の転送などによる同値関係を利用して、クラスの構造やクラス間の関係を識別することを支援する。

（2）業務ルール抽出支援

既存ソフトのコードからフラグなどの実装上のデータ項目を削除し、データ項目やプロセスの重複を排除することにより、業務に関係するデータ項目のみで構成した業務ルールの抽出を支援する。

（3）クラスコード生成

属性に対する取得／設定／検査などの基本的なメソッドの処理内容の型紙となるテンプレートを用意し、それを各属性に適用することにより基本的なメソッドのコードを生成する。設定／検査メソッドの対象となる属性に関する業務ルールが存在する場合には、それも利用してコード生成を行う。

本システムは事務処理系ソフトウエアのリエンジニアリングを目的としている。従って、既存システムの言語はCOBOLであり、再構築後のシステムの言語は、OOCOBOLまたはC＋＋とした。

3．業務オブジェクトモデル構築支援方式

3.1 目的

オブジェクト指向方法論を用いて業務ソフトウエアを開発する場合、クラス／属性／クラス間の関係を識別する一般的な規則が存在しないため、要求仕様から業務ソフトウエアの静的なモデル、すなわち業務オブジェクトモデルを構築することが困難になっている。

業務オブジェクトモデル構築支援の目的は、既存業務ソフトウエア中のレコード定義などの設計情報を有効に利用することにより、各識別作業を容易にし、業務オブジェクトモデルの構築工数を削減することである。

3.2 モデル構築方式

既存業務ソフトに関する業務オブジェクトモデルに構築において、OMT[2]などのオブジェクト指向方法論を適用した場合、既存業務ソフトのマニュアル、仕様書を要求仕様と見なし、これらからデータ項目を洗い出し、クラス／属性／関連の識別を行うことになる。

事務処理を行う業務ソフトは、データを中心に設計されており、データ構造に関する設計情報であるレコード定義がクラスになると考えることは自然である。ただし、既存のレコード定義は必ずしも正規化されていないので、1レコードが1クラスに相当するとは限らない。しかし、モデル構築者にとって、レコード定義情報などの設計情報は各識別作業における有効な情報となる。設計情報を利用したモデル構築の各工程の作業内容を以下に述べる。

（1）クラスの識別

既存ソフト中のレコードの識別に用いているデータ項目（これをキーデータ項目と呼ぶ）をクラスを識別するデータ項目の候補として示し、その中からモデル構築者がクラスの識別子を選択し（選択したデータ項目を識別子データ項目と呼ぶ）、クラスを作成する。既存ソフトを利用することにより、クラス識別子の候補を、すべてのデータ項目からレコードを識別するデータ項目に絞り込むことが可能になり、クラス識別作業を効率化することができる。

（2）属性の識別

モデル構築者が作成したクラスの識別子データ項目、およびこれと転送関係などにあり、名称が異なるが同じ意味を持つデータ項目（これを異名同義データ項目と呼ぶ）を要素として持つ既存レコード中の他のデータ項目をそのクラスの属性の候補として示し、その中からモデル構築者が必要なデータ項目をそのクラスの属性として登録する。既存ソフトを利用することにより、あるクラスの属性候補の範囲をすべてのデータ項目からそのクラスの識別子デ

ータ項目に関係するデータ項目に絞り込むことが可能になり、属性の識別作業を効率化することができる。

（3）関係の識別

同一の既存レコードに含まれている複数の識別子データ項目の関係をクラス間の関係の候補として示し、その中からモデル構築者がクラス間の関係を登録する。既存ソフトを利用することにより、関係の候補をすべてのクラス間の関係から、同一レコード上の識別子データ項目間の関係に絞る込むことになり、関係識別作業を効率化することができる。

以上のように、各工程の識別において、必要な情報を絞り込むことが可能になり、識別作業を効率化することができる。

再構築の対象となる既存業務ソフトウエアが大規模の場合、モデル構築の各工程で利用する設計情報も大量となり、各工程の識別作業があまり効率化されない可能性が生じる。このような場合、まず対象ソフトウエアの主要データを蓄積しているマスタファイル、DB、およびそれを利用するプログラムをマニュアル、運用手順書、JCL などから識別し、識別したプログラムの設計情報のみを利用して、業務オブジェクトモデルを構築する。

3.3 支援ツールの概要

業務オブジェクトモデル構築支援ツールは、既存ソフト中のレコード定義情報や異名同義情報を用いて、キーデータ項目の候補一覧や特定のデータ項目に関係するレコードのレイアウトなどを表示し、マウスを利用した簡単な操作によりクラスの登録、属性の登録、関連の登録を実現している[3]。また、表示中のレコードの各データ項目の状態（編集中のクラスに登録済み／他のクラスに登録済み／未登録）を色により表現しており、3.2 で述べたクラス・属性・関係の識別を容易にしている。図2に、キー候補一覧から選択された反転項目に関するレコードを表示している様子を示す。

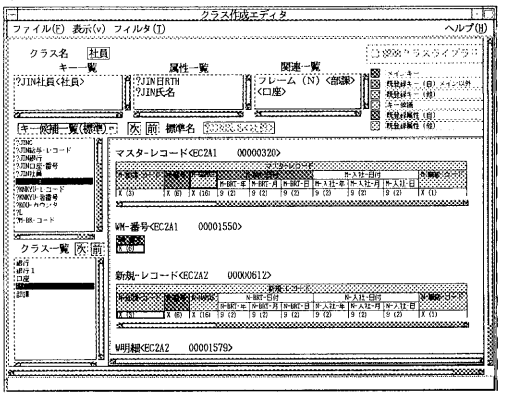

図 2 業務オブジェクト構築支援ツールの画面

3.4 関連ツール

オブジェクトモデル構築支援ツールでは、クラスの継承関係の表示や、あるクラスに対する関連クラスの表示を行うことはできるが、クラス間のすべての関係を一度に表示することができない。そのため、クラス間の関係からクラス全体を表示するための各クラスの位置情報を計算する機能を既開発のオブジェクト図エディタに追加し、全体像の把握をエディタ上で行うことにした。

4. 業務ルール抽出支援方式

4.1 目的

既存ソフトウエアは幾度も保守／拡張されているため、一般に、異名同義データが多数存在し、あるデータ項目の使用目的を網羅的に把握することが困難である。また、あるデータ項目の値を導出する式が複数箇所存在するとき、どの式がどういう条件の時実行されるかを把握することも困難である。そこで我々は、デー

タ項目や計算式を整理し、重複を排除することにより、データ項目とそれに関係する業務ルールを明確にすることを目的として、業務ルール抽出手順と支援ツールを開発した[4]。抽出した業務ルールは、標準的な業務データ名称を持つ項目のみで構成されており、フラグなどの実装上の項目を含んでいないため、既存のプログラムの理解だけでなく、オブジェクト指向環境でも容易に再利用することができる。

4.2 ルール抽出手順と支援ツール

（1）データリバース

まず、ＤＢあるいは画面・帳票に関係する仕様書や既存ソフトウエアのレコード定義から業務上必須であるデータ項目を洗い出し、各データ項目に対して標準名称を付与する業務データ分析作業を手作業で行う。次に、既存ソフトウエア中の異名同義データ項目の抽出・グループ化を行い[5]、グループ毎に標準名称を設定するデータ同値解析作業をバッチツールで行う。標準名の付与には、業務データ分析作業における成果物である既存データ項目と標準名の関係を利用する。

（2）ルールリバース

ルールリバースでは、プログラム内のデータ項目を標準名称に置換し、GO TO 文を排除する。また、プログラムのソースコードを二項関係に分解し[5]、同種類のものを排除する。二項関係は、１つの操作とその操作に関係する２つの要素であり、プログラムコードはすべてこの二項関係で整理することができる。例えば、

Ａ ＝ Ｂ ＋ Ｃ

と記述されたプロセスは、

識別子ＣＰ０	Ａ　ＣＰ１　＝
識別子ＣＰ１	Ｂ　Ｃ　＋

と表現できる。これらの作業はすべてバッチツールで行う。

データ／ルールリバースにおけるすべての解析情報をリポジトリへ登録しており、ブラウザを用いて、必要な情報を検索・出力すること（図３参照）や識別した実装上の項目を排除することができる。また、ルールリバースによる重複排除の結果、プログラム解析情報が削減されており、ブラウジング作業が効率化されている。

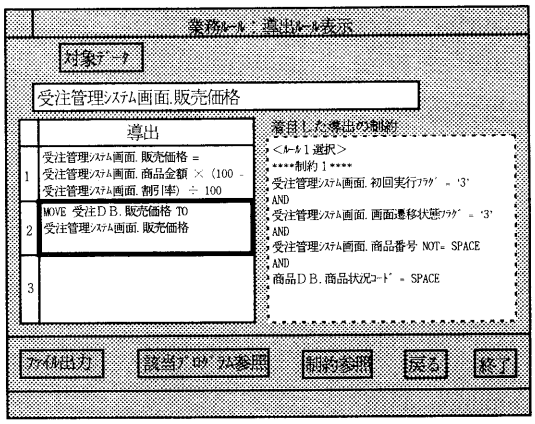

図 3　業務ルール抽出支援ツールの画面

5 クラスコード生成方式

5.1 目的

オブジェクト指向方法論を用いてソフトウェアを開発する場合、クラス、属性、クラス間の関係を識別した後、または同時にクラスや属性を操作するメソッドの識別を行う。これを行うためには、開発者が業務フロー図、業務シナリオを参考にして各オブジェクトの機能を一つ一つ洗い出してメソッドを識別する必要があり、多大な作業を要する。

そこで、メソッド識別を支援する可能性を探るため、既存のオブジェクト指向言語で開発されたシステム中のメソッドを、メソッドの作用対象（属性、属性の集合、１：１関連の関係を持つクラス、１：Ｎ関連の関係を持つクラス、

１：１構成の関係を持つクラス、１：N構成関係を持つクラス）と機能（生成、削除、設定、追加、取得、更新、検索、判定）による分類体系で分類を行った[6]。この結果、操作対象別に必要な機能が限定されていること、約９０％のメソッドがこの分類体系で分類可能であることが判明した。このことから、メソッドの識別にこの分類体系が有効に活用できるといえる。また、対象システムのメソッドの中には、比較的単純で定型的なコードで実装されているものも多数存在し、これらについてコードを自動生成することも可能であった。

そこで、我々はメソッドの識別の支援とメソッドの実装工数の削減を目的とした、クラスコード生成方式を検討した。

5.2 コード生成方式

上記目的を達成するため、以下の方式を検討した（図４）。

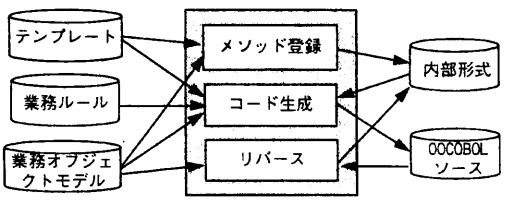

図 4　クラスコード生成ツールの概要

（1）メソッドの容易な登録

利用者がメソッドの作用対象を選択した時、作用対象の種別に応じて、必要である可能性の高い機能の一覧を利用者に提示する。利用者は表示された機能を用いてメソッドの識別を行い、必要な機能をメソッドとして登録する。

（2）テンプレートの使用によるコード生成

調査対象のシステムでは、分類体系の分類毎に代表的な処理のパターンが存在した。そこで、分類毎のメソッドの処理パターンをコードの雛形となるテンプレートとして用意し、属性名やクラス名などの固有値を埋め込んでソースコードを自動生成する方式とした。また、利用者やシステム部門による設計スタイルやコーディングスタイルのカスタマイズや多言語対応を可能にするため、テンプレートをツールの入力情報とした。このため、テンプレートを記述するための言語を設計し、ツール側でテンプレート記述言語を解釈し、コードを生成する方式とした。また、テンプレートには、（1）で用いる作用対象と機能の関係も記述し、この関係もカスタマイズ可能にした。

（3）業務ルールを利用したコード生成

属性に対する「設定」メソッドには、引数として受けとった値を単純に代入するだけのものと、引数として受けとった値と自クラスの他の属性とを使って一連の計算を行い、その結果を代入するものがある。この２種類のメソッドの内容の差異は対象となるデータ項目の値を計算する導出式が存在するどうかによるものである。そこで、導出式が存在するデータ項目のための設定メソッドに対するテンプレートを作成し、導出式を埋め込んでコードを生成することにより生成率を向上させることにした。同様に、データ項目の判定メソッドに対しては値制約式を埋め込むことにした。

図５にコード生成のイメージを示す。

また、クラスコード生成ツールを既存のOOCOBOL ソースコードに対するメソッドの追加／編集にも利用できるように以下の機能を追加した。

（4）ソースコードからの逆生成

既存ソースコードのクラス名、メソッド名をキーにして、既存メソッドのコードを業務オブジェクトモデルに対応付け、クラスコード生成ツールで扱う内部形式に変換する。

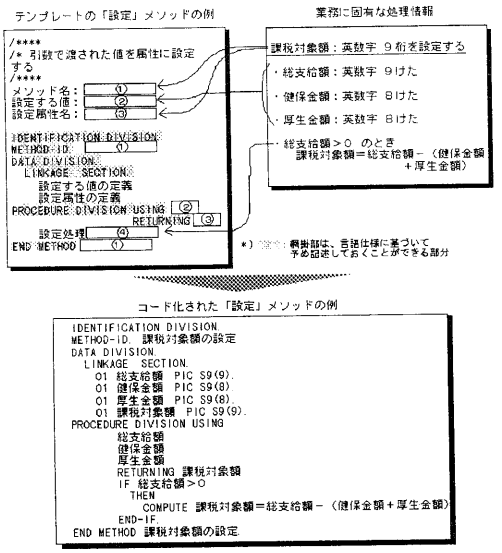

図 5 コード自動生成のイメージ

5.3 支援ツールの概要

クラスコード生成ツールは主に4つの画面から構成されている。これらの画面を用いてツールの機能を説明する。

（1）クラス階層表示画面（図6）

この画面には、作業中のオブジェクトモデルのクラス集合のクラス階層が表示され、ユーザはクラス間の継承関係の理解と各クラスに含まれる属性・メソッドの理解に役立てる。クラス間に継承関係があれば下位クラスは上位クラスよりもインデントされて表示される。

クラス名をクリックして選択状態にするとそのクラスが持っている属性名の一覧またはメソッド名の一覧が表示される。どちらを表示させるかは画面下方のボタンで設定する。

クラス名を選択してクラスコード生成ボタンをクリックすると指定したファイルに選択クラスのコードが出力される。出力されるコードは、クラス定義画面（図7）で選択されたメソッドのコード以外に、属性などのクラス定義のコードも含む。

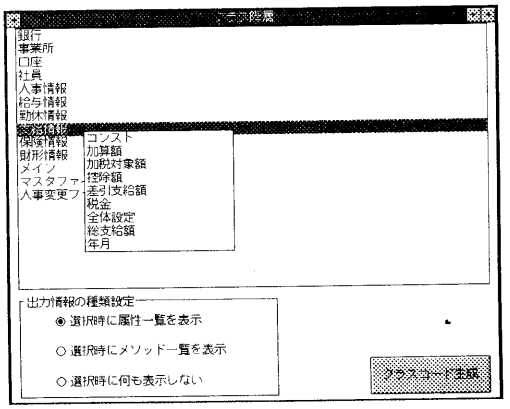

図 6 クラス階層表示画面

（2）クラス定義画面（図7）

この画面では、ユーザは主にメソッドの選択を行う。

属性・関連表示欄では、属性や実装上必要となる他のクラスへのポインタ情報を表示しており、ある要素をクリックして選択状態にすると、テンプレートを用いてその要素（作用対象）に関するメソッド候補一覧がメソッド候補表示欄に表示される。ユーザは候補のひとつをダブルクリックすることによりそのメソッドの登録を指示する。登録を指示されたメソッド候補はメソッド表示欄に表示される。

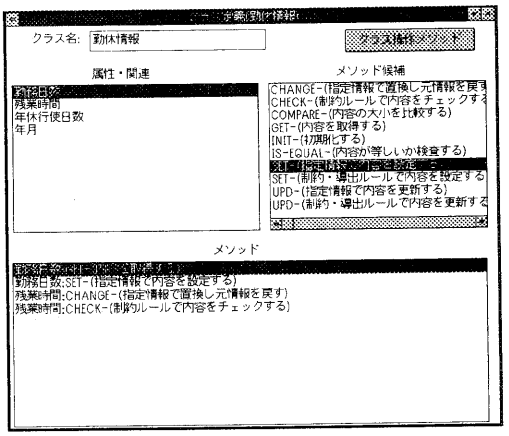

図 7 クラス定義画面

クラス操作メソッドボタンを押すと、メソッド候補表示欄にクラス全体に影響するメソッドの候補一覧が表示される。

また、この画面上で、属性や関連を追加／削除することもできる。

テンプレートでは、メソッド毎にメソッドの自動登録の指示を記述することが可能であり、自動登録を指示されたメソッドは、利用者からの対話指示がなくても、メソッド表示欄に表示される。

（3）業務ルール設定画面

この画面にはユーザが選択したデータ項目に関する業務ルール一覧が表示され、ユーザはコード生成時に使用する業務ルールの選択を行う。

（4）メソッドコード編集画面

この画面にはユーザが選択したメソッドに関してテンプレート、業務ルールから生成されるコードが表示され、ユーザがコードを確認することができる。また、表示されたコードを編集することも可能であり、編集を行ったメソッドに関しては、コード生成時にその編集内容が反映される

6．オブジェクト指向開発の支援環境の評価

ある収益算出システムに対して、（1）要求仕様から作成する従来方式、（2）既存ソフトを利用し、ツールで支援する本稿での提案方式、の2通りで業務オブジェクトモデルを構築した結果、提案方式は従来方式の1／5の工数でモデルを作成できた。また、業務ルール抽出支援ツールでは、仕様書に記述されている業務ルールを網羅し、かつ仕様書未記述の業務ルールまで抽出しており、有効性を確認できた。

7．おわりに

既存システムをオブジェクト指向環境で再構築するための支援環境であるオブジェクト指向リエンジニアリングシステムを開発した。これにより、既存ソフトの設計情報の抽出／再利用が容易になり、新システムを低コストかつ高品質にオブジェクト指向環境で再構築することができる。

業務ルール抽出支援ツールのブラウザ機能を用いて検索する設計情報は、クラス／属性／クラス間の関係／メソッドの各識別において、有効な判断材料になる。しかし、現在各ツールはファイルによる疎結合の状態であり、ツールを切替える必要がある。

今後は、本システムを構成する3ツールの機能統合を実現するとともに、実際の再構築に全面適用し、システムの機能拡張を行っていく予定である。

参考文献

[1]小田，リエンジニアリングにおけるオブジェクト指向分析の役割，オブジェクト指向95シンポジウム論文集，情報処理学会，1995

[2]Rumbaugh, et al., Object-Oriented Modeling and Design, Prentice-Hall, 1990

[3]佃 他，既存業務ソフトのデータ構造に着目した業務オブジェクトモデル構築方式，情報処理学会第51回全国大会論文集4M-4, 1995

[4]津田 他，DORE(1)-二項分析による既存プログラムからの業務ルール抽出技術-，情報処理学会第52回全国大会論文集6S-3, 1996

[5]佃 他，CSS 統合開発環境(7)-リエンジニアリング-，情報処理学会第45回全国大会論文集4U-8, 1992

[6]永岡 他，既存ソフトウエアの再利用とオブジェクト指向によるシステム開発における考察，ソフトウエア工学研究会論文集 Vol.95,No.1, pp.139-144, 1995