ソフトウェア工学 76－2
（1990. 12. 4）

ソフトウェア設計手順定式化の一提案

山口　和幸*　　　篠田　晃**

*日本電信電話株式会社　ソフトウェア開発センタ　**日本電信電話株式会社　ソフトウェア研究所

ソフトウェア設計作業の自動化率向上には，適切な設計手法の採用とそれに基づく設計作業の定式化が重要な課題である．これまで，各種の設計手法が提案されているが，いずれも概念的な枠組みを与えるに留まり，ソフトウェア設計の詳細な手順を規定したものは少ない．このため，同一の設計法を用いても①個人毎に設計手順がまちまちとなる，②設計担当者のスキルに頼る割合が大きく，設計作業が共通化し難い，③設計ノウハウが属人的になり，均質な設計結果を得難い，などの問題がある．これらの問題点を解決するには，設計作業を分析することにより設計作業モデルを作成し，設計作業を定式化する必要がある．このため，設計手順の具体化度が高いオブジェクト指向設計法をベースとし，これの規定していない作業手順の具体化，詳細化を図った「改良設計手順」を提案する．ついで，この改良設計手順に基づく設計手順と設計情報とを，それぞれ設計シナリオおよび設計フォームと呼ぶ表現形式で定式化した設計作業モデルを示す．本設計作業モデルに基づく設計実験を実施したところ，モジュール構造設計作業の約７０％を定式化することができた．

AN APPROACH TO ALGORITHMIC DESCRIPTION OF SOFTWARTE DESIGN PROCESS

Kazuyuki Yamaguchi*　　Akira Shinoda**

* NTT Software Engineering Center　　** NTT Software Laboratories

* 2-1-1 Nishigotanda,Shinagawa-Ku,Tokyo 141,Japan

** 223-1 Yamashita,Naka-Ku,Yokohama 231,Japan

An improved version of Booch's Object-Oriented Design methodology is proposed. The improvements include refinement and clarification of a few design steps which the original methodology does not specify. Next a design process model is proposed which represents the design process and design information in scenarios and design forms ,respectively.

A preliminary experiment has shown that the design process model can formalize 70% of module structure design.

１． まえがき

ソフトウェアの生産性・信頼性の向上を図るには，ソフトウェア開発の初期段階である設計の自動化が重要な課題である．ソフトウェア設計作業の自動化率向上には，適切な設計手法の採用とそれに基づく設計作業の定式化が重要である．これまで，各種の設計手法が提案されているが，いずれも概念的な枠組みを与えるに留まり，ソフトウェア設計の詳細な手順を規定したものは少ない．このため，同一の設計法を用いても①個人毎に設計手順がまちまちとなる，②設計担当者のスキルに頼る割合が大きく，設計作業が共通化し難い，③設計ノウハウが属人的になり，均質な設計結果を得難い，などの問題がある．これらの問題点を解決するには，設計作業の分析に基づき設計作業モデルを作成し，設計作業を定式化する必要がある．設計作業分析については，ＬＥＯＮＡＲＤＯプロジェクト[8]，ＴＲＩＡＤプロジェクト[9]などでも進められている．ＬＥＯＮＡＲＤＯプロジェクトでは，ソフト開発チーム内のコミュニケーションに基づく「設計行動モデル」を追求している．一方，ＴＲＩＡＤプロジェクトでは，定型的な設計フォームによりプログラマの思考過程を支援する人間工学的方法論を提案している．しかし，いずれのプロジェクトでも設計手順レベルまで詳細化したモデルの提案は今後の研究課題である．

本稿では，まず設計手順の具体化度が高いオブジェクト指向設計法[1]~[5]をベースとし，これの規定していない作業手順の具体化，詳細化を図った「改良設計手順」を提案する．ついで，この改良設計手順に基づく設計手順と設計情報とをそれぞれ設計シナリオおよび設計フォームと呼ぶ表現形式で定式化した設計作業モデルを示す．最後に，本設計作業モデルに基づく設計実験を実施し，モジュール構造設計作業の約７０％を定式化することができた結果を示す．

２． Boochのオブジェクト指向設計法

Booch[1]~[3]のオブジェクト指向設計法は，本来Ａｄａ[12]に適合させる設計法として考案されたものである．しかし，設計過程ではＡｄａをＰＤＬ（Programming Design Language）として用い，プログラムの実現段階で目的言語に変換することにすれば，オブジェクト指向設計法を他のプログラミング言語と組み合わせることが可能である．

本章では，Boochのオブジェクト指向設計法の設計手順について概説する．オブジェクト指向設計法は，大きく以下の３ステップから構成される（表１）．最初の２ステップは，オブジェクト指向設計法の環境を規定するステップであり，第３ステップの「戦略の形式化」がオブジェクト指向設計法の中核部分である．

表１　Ｂｏｏｃｈのオブジェクト指向設計法の概要

設計ステップ		作業内容
問題の定義		・問題の解析
非形式的戦略の記述		・非形式的戦略の記述
戦略の形式化	オブジェクトの識別	・オブジェクトの識別 ・オブジェクトと属性の対応付け ・オブジェクトのグループ化とタイプの識別
	オペレーションの識別	・オペレーションの識別 ・オペレーションと属性の対応付け ・オペレーションのグループ化
	インタフェースの定義	・可視インタフェースの形式的記述 ・設計上の決定の分析と解明 ・可視インタフェースの図形による記述
	プログラムの実現	・個々のオペレーションをプログラミング言語で実現 ・ただし，個々のオペレーションが複雑すぎる場合には，本設計法を再帰的に適用

（１）問題の定義（Defining the Problem）ステップ

設計対象システムに関するあらゆる情報を収集・解析し，設計対象システムの設計条件をまとめる．

（２）非形式的戦略の記述（Developing an Informal Strategy）ステップ

上記ステップで明らかになったシステムの処理動作を自然言語を用いて文法的に正しい文章で記述する．

（３）戦略の形式化（Formalizing the Strategy）ステップ

以下の手順により非形式的戦略を具体化する．

（ａ）オブジェクトの識別（Identifying the Objects of Interest）

非形式的戦略から名詞，代名詞，名詞句を抽出し，プログラムの実現に必要なオブジェクトか否かを判

断する．ついで，オブジェクトはその属性（形容詞，形容詞句）により特徴付けられることから，属性を調べて同一オブジェクトの識別を行う．同一オブジェクトには同一のタイプ（オブジェクトのクラス）名を付与する．これにより，オブジェクトがグループ化される．

（ｂ）オペレーションの識別（Identifying the Operations of Interest）

非形式的戦略から動詞，動詞句，述語を抽出し，プログラムの実現に必要なオペレーションか否かを判断する．ついで，オペレーションはその属性（副詞，副詞句）により特徴付けられることから，属性を調べて同一オペレーションの識別を行う．更に，オペレーションの操作対象となるオブジェクトのタイプ毎にオペレーションをグループ化する．タイプとオペレーションとで抽象データタイプを構成する．

（ｃ）インタフェースの定義（Defining the Interfaces）

抽象データタイプ間のインタフェースおよび依存関係を決定し，Booch図（Boochが提唱するモジュールのインタフェースを記述する図）で記述する．モジュールとしてパッケージ，タスク，サブプログラムおよびジェネリックのいずれを採用するかは，設計者に委ねられる．

（ｄ）プログラムの実現（Implementing the Program）

Booch図に記述したインタフェースに基づいて，各モジュールを段階的にプログラムコードまで詳細化する．設計者は，あるモジュールが大きすぎ，そのままではプログラムコードまで詳細化することが困難と判断した場合，そのモジュールの機能の実現を新たな問題とみなし，再帰的に本設計法を適用する．

３．オブジェクト指向設計法の定式化

設計作業手順の定式化には，何等かの設計法に基づく設計作業の分析が必要である．他の設計法と比較し，より具体性の高い設計手順を与えることから，筆者らは，オブジェクト指向設計法をベース設計法として採用した．本章では，まず，ソフトウェア設計作業にオブジェクト指向設計法を適用した際の問題点およびその改良法について述べる．ついで，改良したオブジェクト指向設計法の定式化について提案する．

３．１　オブジェクト指向設計法の具体化

Boochのオブジェクト指向設計法に従った設計を実験的に行った結果，オブジェクト指向設計法は以下の点を具体化する必要があることが判明した．

（１）問題解析手法の特定

「問題の定義」ステップにＳＡＤＴ[10]，ＤＦＤ[6][11]などの手法を適用することを前提としているが，具体的な適用方法については述べられていない．しかし，完結した設計法とするためには問題分析手法を特定する必要がある．そこで，筆者らはデータフロー図に基づく問題分析手法を適用し，「問題の定義」ステップを次の２ステップに具体化した．

（ａ）入出力情報解析ステップ

ここでは，設計対象システムが必要とする入出力情報の包含関係を調べ，この関係を情報階層木として記述する．この記述により，①システムの入出力処理対象データおよびシステムの処理を明確にすることや，②オブジェクトのグループ化において，データが所属する木を基にオブジェクトタイプを判断することができる．

（ｂ）シーケンスフロー記述ステップ

Boochのオブジェクト指向設計法は，設計仕様（非形式的戦略）からオブジェクトやオペレーションを選択する具体的な手順を与えている．ところが，設計初期段階では設計仕様そのものが曖昧であることが多いため，オブジェクトとオペレーションの抽出および対応関係の認識を誤り易い．そこで，設計仕様の単文をバブルに，単文の処理の流れをアークに対応付けたチャート（シーケンスフローチャートと呼ぶ）を記述するシーケンスフロー記述ステップを追加した．この結果，①文章表現のみでは把握し難い処理の関連を明らかにすることや，②システムの動作を表わす文章の記述誤り（不足，過剰，不適など）を検出することができる．

（２）問題解析作業と記述作業の不可分性

Boochのオブジェクト指向設計法では，「問題の定

義」結果を受けて「非形式的戦略の記述」を行うように規定されている．しかし，実際にはシステムの処理内容を記述し，充分明確になるまで解析作業と記述作業を繰り返す必要がある．この点を明確化するため，「非形式的戦略の記述」ステップは「問題の定義」ステップに吸収した．

（3）作業実施基準の曖昧性

Boochのオブジェクト指向設計法は，①同一オブジェクトの識別基準や②オブジェクトのグループ化基準などが曖昧であり，オブジェクトの識別作業を機械的に実施できない．そこで，以下の基準を設け，オブジェクトの識別およびオペレーションのグループ化処理を具体化した．

（a）オブジェクト候補に対して次の基準を適用し不必要な候補を削除する（オブジェクト削除基準）．

①設計対象システムの中で直接使用されないもの．

②「プログラムを起動する」などのシステムの抽象的な操作を表わすもの．

③「処理」，「実行」などの一般的な動作を表わすもの．

（b）オブジェクトを修飾しているすべての形容詞類を抽出する（形容詞類抽出基準）．ここで，形容詞類とは，動詞，形容詞，形容動詞の連体形および連体修飾詞の総称である．

（c）「名詞，代名詞，名詞句」が同じ表現で，かつ「形容詞類」が同じ場合には同一オブジェクトとする（同一オブジェクト識別基準）．

（d）次の基準によりオブジェクトをグループ化し，タイプ名を付与する（タイプ分類基準）．

①オブジェクトが所属する情報階層木が同じ．

②オブジェクトに対応する形容詞類が同じ．

③実世界で実体が同じ．

④オブジェクトが意味的に同じ．

（e）オペレーション候補に対して次の基準を適用し不必要な候補を削除する（オペレーション削除基準）．

①設計対象システムを実現するのに直接使用されないもの．

②「プログラムを起動する」などのシステムの抽象的な操作を表わしているもの．

③対応するオブジェクトが存在しないもの．

（f）次の基準を適用しオペレーションの同一性を判断する（同一オペレーション識別基準）．

①対応するオブジェクトが同じ．

②副詞類が同じ．

③「動詞，動詞句，述語」が同じ表現である．

④オペレーションが意味的に同じ．

ここで，副詞類とは，オペレーションを修飾している部分の総称である．

3.2　オブジェクト指向設計法の定式化

3.2.1　改良したオブジェクト指向設計法の概要

3.1で指摘した問題点の改善を図った「改良設計手順」について述べる．設計作業をアクティビティ，タスクおよびユニットの3階層で構成することとした．ここで，設計作業ユニットは，設計作業の最小単位である．設計作業の全体の流れを図1に示す．以下に，改良したオブジェクト指向設計法の設計手

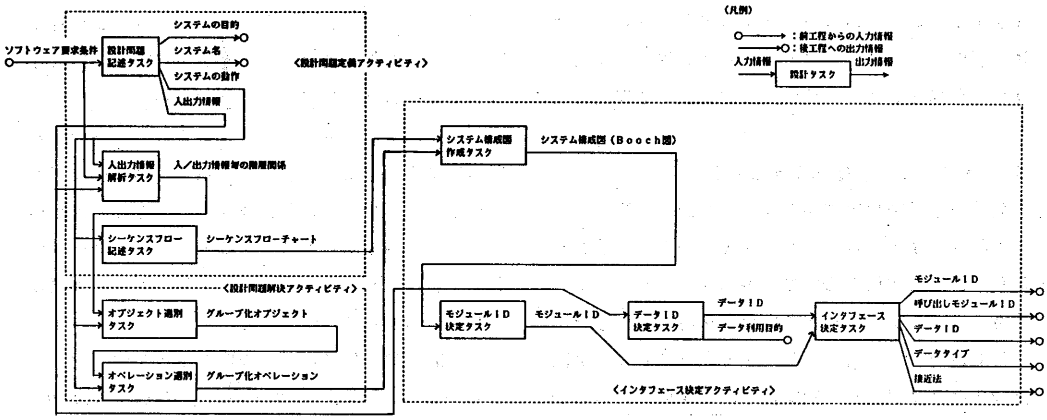

図1　改良オブジェクト指向設計法の作業フロー

順を構成する設計アクティビティおよび設計タスクの処理を概説する．

（a）設計問題定義アクティビティ

設計問題定義アクティビティは，以下の設計タスクから構成される．

（i）設計問題記述タスク

本タスクでは，前工程の生産物である基本仕様書で与えられる設計対象システムの目的，位置付けなどを吟味し，設計対象システムの名称，目的，動作および入出力情報などを記述し，設計条件を明らかにする．

（ii）入出力情報解析タスク

設計問題記述タスクで記述した内容から設計対象システムが必要とする入出力情報の構成およびその用途を抽出し，記述する．この記述は，システムの処理流れの把握，オブジェクト選別タスクのオブジェクトタイプの識別およびデータＩＤ決定タスクのデータ識別などに利用する．

（iii）シーケンスフロー記述タスク

設計問題定義アクティビティで作成したシステム動作の記述を基に，処理順序を表わすシーケンスフローチャートを作成する．本タスクで設計条件の誤りを検出した場合には，設計問題記述タスクに戻り，手直しする．

（b）設計問題解決アクティビティ

設計問題解決アクティビティは，以下の設計タスクから構成される．

（i）オブジェクト選別タスク

設計問題記述アクティビティで作成したシステム動作の記述からオブジェクトを抽出する．ついで，その中から設計対象システムの実現に必要なオブジェクトを選別する．選別したオブジェクトには，オブジェクト名を付与する．また，各オブジェクトにタイプ分類基準を適用し，オブジェクトをグループ化すると共にオブジェクトの持つ意味を特徴付けるオブジェクトタイプ名を付与する．

（ii）オペレーション選別タスク

設計問題記述アクティビティで作成したシステム動作記述からオペレーションを抽出する．ついで，その中から設計対象システムの実現に必要なオペレーションを選別する．選別したオペレーションには，オペレーション名を付与する．また，操作対象となるオブジェクトのタイプ毎に，選別したオペレーションを分類する．同一のタイプに属するオペレーションは，同一グループに含めることにより，データ抽象化を図ることができる（データ抽象化タイプの実現）．

（c）インタフェース決定アクティビティ

データ抽象化タイプ（モジュール）の構成，モジュール名，モジュールで取り扱うデータ名およびモジュール間インタフェースなどを明らかにする．本アクティビティは，以下の４設計タスクからなる．

（i）システム構成図作成タスク

モジュールの構成およびモジュール間の依存関係をBoochチャート[1]で表現する．

（ii）モジュールＩＤ決定タスク

設計対象システムを構成するモジュールをプログラミング言語の具体的なモジュールに割り当てる．従って，モジュールを識別するモジュール名として，プログラミング言語で使用できる名称を付与する．

（iii）データＩＤ決定タスク

入出力情報をプログラミング言語のデータに割り当てる．従って，データを識別するデータ名として，プログラミング言語で使用できる名称を付与する．

（iv）インタフェース決定タスク

モジュールＩＤ決定タスクで規定したモジュール名を用いて，モジュール間の呼び出し関係，モジュールで取り扱うデータ名（アーギュメント／パラメータ名，共通データ名など）および接近法（参照，更新，設定・参照）などのモジュール・インタフェースを記述する．

（d）プログラムの実現

Booch図に記述したインタフェースに基づいて，各モジュールを段階的にプログラムコードまで詳細化する（２（３）（d）を参照）．

３.２.２　設計シナリオ・設計フォームによる定式化

前項の作業内容を「設計シナリオ」と「設計フォーム」で定式化する．設計シナリオは各タスクの作業順序および作業内容を表現するものであり，設計

フォームは各タスクの設計情報を表現するものである．設計者は設計シナリオに従って設計作業項目対応に用意されている設計フォーム内にデータやチャートを埋め込むことで設計を進めることができる．

（1）設計シナリオによる設計手順の表現

設計作業ユニットでは，入力情報にある変形を加えて出力情報を生成している．設計作業ユニットの詳細な仕事は，次のようにモデル化できる．

（a）入力情報から出力情報への変換は，ある「基準」に基づき行われる．ここで，「基準」は出力情報の選定条件などの役割を持つ．

（b）設計者は「基準」の中から，①出力情報の候補となる設計要素の抽出効率が高い，②適用し易いなどの性質を備えた「基準」をあらかじめ抽出している（これを着眼点と呼ぶ）．

（c）「基準」および「着眼点」を用いて，次の3ステップを繰り返すことにより出力情報を生成する．

（i）着眼点に基づき入力情報から設計要素を抽出する．

（ii）これに残りの「基準」を適用して絞り込む．

（iii）絞り込まれた候補にある「変更」を加える．

すなわち，設計作業ユニットは，図2に示すような構造を持つ．図2のようにモデル化した設計作業ユニットを作業順に接続することにより，各設計タスクで実施すべき作業内容を表現することができる．これを設計シナリオという．設計シナリオの図式表現例を図3に示す（なお，設計シナリオは図式表現の他に疑似プログラム表現も可能である）．

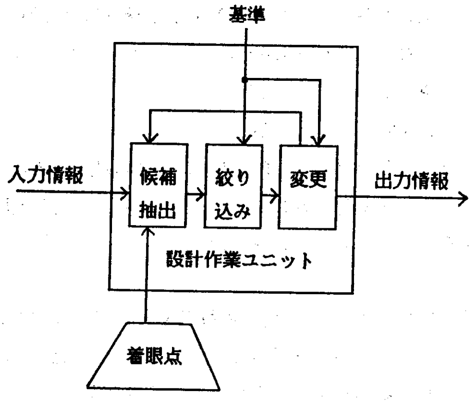

図2　設計作業ユニットの構造

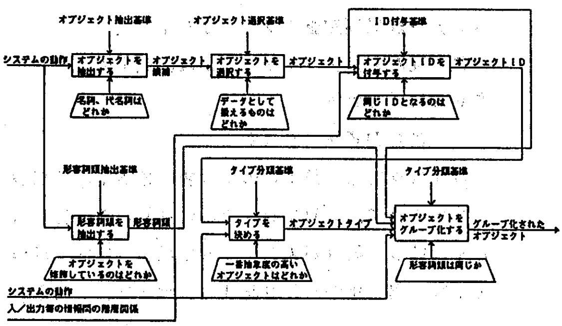

図3　設計シナリオの図式表現
（オブジェクト選別タスクの例）

（2）設計フォームによる設計情報の表現

設計フォームと呼ぶ定形的な枠組みを定め，各設計アクティビティおよび設計タスクの作業情報の記述を定式化した．設計フォームは，ほぼ設計タスク対応に用意した．設計フォーム例を図4から図8に示す．

4．設計例

前章の設計作業モデルに基づく設計作業例を述べる．設計例として「酒屋在庫管理問題」[7]を取り上げた．以下に，改良設計手順の主要部分である設計問題定義アクティビティ（設計問題記述タスク，シーケンスフロー記述タスク），設計問題解決アクティビティおよびインタフェース決定アクティビティ（システム構成図作成タスク）の設計作業について述べる．

（1）設計問題定義アクティビティ

（a）設計問題記述タスク

在庫管理プログラムの設計目的，システム動作などを記入し，設計対象システムの設計条件を明らかにする．設計問題記述フォームにシステムの目的，システムの動作および入出力情報の内容を記入する（図4）．

（b）シーケンスフロー記述タスク

システム動作の記述内容をシーケンスフローチャ

ートで表現し，処理順序の明確化を図る．これによりシステム動作の記述誤りや記述不足などを検出することができる（図５）．

（３）設計問題解決アクティビティ

（ａ）オブジェクト選別タスク

（ｉ）設計問題記述タスクのシステム動作記述からオブジェクト候補を抽出する．まず，システム動作記述上でオブジェクトに下線を引き，この結果をオブジェクト選別フォーム内の「名詞，代名詞，名詞句」欄に記入する．

（ⅱ）それぞれのオブジェクト候補に対して，オブジェクト削除基準を用いてその要否を記入する．

（ⅲ）オブジェクトを修飾している形容詞類をシステム動作記述から抜き出し，対応するオブジェクトの形容詞類欄に記入する．

（ⅳ）同一オブジェクト識別基準を適用してオブジェクトを識別し，オブジェクトＩＤを付与する．ついで，オブジェクトのグループ化基準を適用してオブジェクトをグループ化し，タイプ名を付与する（図６）．

（ｃ）オペレーション選別タスク

設計問題記述タスクで作成したシステム動作記述より，オペレーション候補を抽出する．オブジェクト選別タスクと同様に，①オペレーション候補，②オペレーションの要否，③オペレーションを修飾している副詞類を記入した後，④オペレーションＩＤを付与する．操作対象となるオブジェクトのタイプ名をタイプ欄に記入し，オブジェクトとの関係付けを行う（図７）．

（４）インタフェース決定アクティビティ

グループ化したオブジェクトおよびそれに対応するオペレーションの関係をシステム構成図作成フォームに記入する．システム構成図は，Booch記法に従ったチャートで表わす（図８）．参照関係は，シーケンスフローチャートを基に記述する．

以上の作業により，在庫管理プログラムのモジュール構造が設計できる．

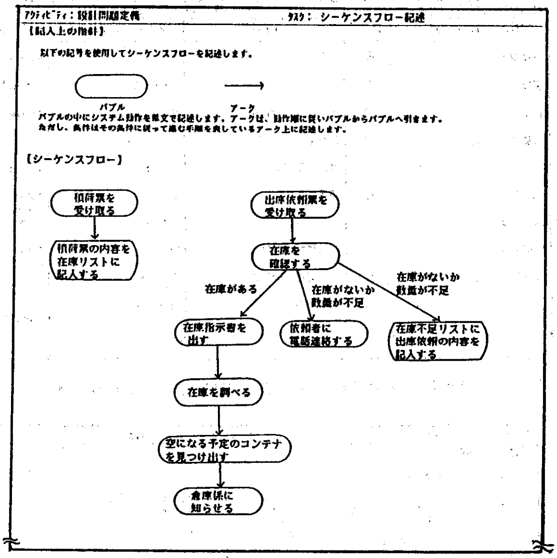

図５　シーケンスフロー記述フォームの記述例

ｱｸﾃｨﾋﾞﾃｨ：設計問題解決　　ﾀｽｸ：オブジェクト選別

名詞、代名詞、名詞句	要・否	形容詞類	オブジェクトＩＤ	タイプ
受付係	○		Receipt	
倉庫係	×			
積荷票	○		Burden	Burden
在庫リスト	○		List	List
出庫依頼	○		Request	Request
在庫	○		List	List
出庫指示書	○		Shipment	Shipment
その旨	○		Shortage	Shortage
依頼者	×			
在庫不足リスト	○		Lack	Lack
品名	○	出庫依頼の内容である	Req_Bur_Name	Request
数量	○	出庫依頼の内容である	Req_Bur_Number	Request
送り先名	○	出庫依頼の内容である	Req_Bur_Address	Request
コンテナ	○	空になる予定の	Empty_Container	Empty_Container

図６　オブジェクト選別フォームの記述例

ｱｸﾃｨﾋﾞﾃｨ：設計問題定義　　ﾀｽｸ：設計問題記述

システム名、システムの目的、システムの動作、入出力情報を以下に記述します。

システム名：ＩＮＶＥＮＴＯＲＹ

システムの目的：ある酒類販売会社の倉庫の受付係の仕事（在庫無し連絡，出庫指示書作成，在庫不足リスト作成）を自動化する．

システムの動作：

受付係は，倉庫係から積荷票を受け取る．受付係は，出庫依頼を受け，そのつど倉庫係へ出庫指示書を出す．在庫がないか数量が不足の場合には，その旨依頼者に電話連絡し，同時に在庫不足リストに記入する．空になる予定のコンテナを倉庫係に知らせる．受付係には，在庫リストが存在し，これを使って，積荷票の内容の記入，在庫の確認を行う．出庫依頼を受けた場合は，在庫を確認する．空になる予定のコンテナを見つけ出すために，在庫を調べる．在庫不足リストに記入するのは，出庫依頼の内容である品名，数量，送り先名である．

入出力情報：

・入力情報：積荷票，出庫依頼

・出力情報：出庫指示，在庫リスト，在庫不足リスト

システム名：[illegible]

作成者：[illegible]　　日付：[illegible]　　通番 No.1

（注）　：設計者の記述部分を示す

図４　設計問題記述フォームの記述例

アクティビティ：設計問題解決　タスク：オペレーション選別

動詞、動詞句、述語	要・否	副詞類	オペレーションID	タイプ
受け取る	○	倉庫係から	Receive1	Burden
受ける	○		Receive2	Request
出す	○	そのつど、倉庫係へ	Send	Shipment
電話連絡する	○	依頼者に	Telephone	Lack
記入する	○	在庫不足リストに	Enter1	Request
知らせる	○	倉庫係に	Inform	Empty_ Container
記入する	○	在庫リストに	Enter2	Burden
確認する	○	出庫依頼を受けたら	Confirm	List
調べる	○	出庫依頼を受けたら	Examine	List
見つけ出す	○	出庫依頼を受けたら	Find_out	Empty_ Container

図７　オペレーション選別フォームの記述例

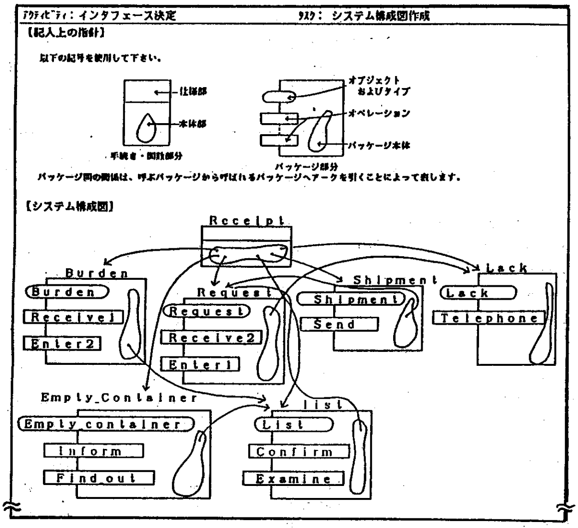

図８　システム構成図作成フォームの記述例

５．評価と考察

設計手順の定式化率は設計手順の再現性の評価尺度である．従って，本設計作業モデルに基づいて設計したとき，設計者が誤解なく実施できる設計作業ユニットの割合で設計手順の定式化率を近似することができる．そこで，定型業務処理，データベース処理および開発支援ツールなど，表２に示す５本の問題を対象として設計実験を行った． 被験者数は３名であり，本設計作業モデルに従って，基本仕様書からプログラム機能を実現するモジュールおよびそのインタフェースを決定するまでの「モジュール構造設計」を行った（なお，モジュール内の処理アルゴリズムを考える工程は，設計者の経験，知識に大きく依存する部分であるため，今回の実験対象外とした）．

表２　設計問題一覧

設計問題	問題概要
報告書作成プログラム	ソフト管理情報の図形，グラフ作成表示プログラム
ソフト管理情報登録プログラム	ソフト管理情報のデータベース登録プログラム
在庫管理プログラム	酒屋在庫管理問題
社内預金プログラム	社内預金業務の自動化プログラム
電卓計算プログラム	計算式を入力し，その答えを出力する電卓計算プログラム

この結果，本設計作業モデルを構成する３５個の設計作業ユニットの内，システム目的やシステム名称の記述，オブジェクトやオペレーションの抽出，オブジェクトＩＤ，オペレーションＩＤやデータＩＤなどのＩＤ付与およびモジュール構成図記述など２６個の設計作業ユニットは，３人の被験者のうち２人以上が３本以上のプログラムで誤解なく実施でき，表層的な違いを除き同一の結果が得られた（表３）．残りの設計作業ユニットは，設計者の業務知識や経験に委ねられる割合が多く，設計作業を機械的に進めることは難しかった．この結果，提案する設計作業モデルにより，モジュール構造設計作業の約７０％（２６／３５設計作業ユニット）を定式化することができた．

６．むすび

Boochのオブジェクト指向設計法の曖昧な設計手順を具体化した「改良設計手順」を示した．ついで，この改良した設計法に基づくモジュール構造設計作業を設計シナリオと設計フォームから構成される設計作業モデルで定式化した．本モデルにより，モジュール構造設計作業の約７０％が定式化されるとの実験結果を得た．従って，本設計作業モデルにより，設計手順の統一化，設計作業の効率化および設計結

果の均質化などの改善が期待できる．

謝辞　本研究の機会を与えて頂き、かつ終始御助言頂いたソフトウェア研究所磯田定宏ソフトウェア基礎技術研究部長に深謝致します。

表３　設計作業モデルに基づく設計作業の定式化

Boochの設計ステップ		改良オブジェクト指向設計法の設計ステップ			定式化できたユニット
		設計アクティビティ	設計タスク	設計作業ユニット	
問題の定義		設計問題定義	設計問題記述	システム名記述	●
				システム目的記述	●
				システム動作記述	
				入出力情報記述	●
			入出力情報解析	入力情報階層記述	
				出力情報階層記述	
			シーケンスフロー記述	バブル抽出	●
				バブル記述	●
				動作順序記述	●
非形式的戦略の記述		—	—	—	—
戦略の形式化	オブジェクトの識別	設計問題解決	オブジェクト選別	オブジェクト抽出	●
				オブジェクト選別	●
				オブジェクトＩＤ付与	●
				形容詞類抽出	
				タイプ名決定	
				オブジェクトグループ化	●
	オペレーションの識別		オペレーション選別	オペレーション抽出	●
				オペレーション選別	●
				オペレーションＩＤ付与	●
				副詞類抽出	
				操作対象オブジェクト決定	
				オペレーショングループ化	●
	インタフェースの定義	インタフェース決定	システム構成図作成	モジュール構成記述	●
				インタフェース関係記述	●
			モジュールＩＤ決定	モジュール名記述	●
				モジュールＩＤ付与	●
			データＩＤ決定	入力データ記述	●
				出力データ記述	●
				内部データ記述	●
				データＩＤ記述	●
				データ利用目的記述	●
			インタフェース決定	モジュールＩＤ記述	●
				呼び出しモジュールＩＤ記述	●
				データＩＤ記述	●
				データタイプ名記述	
				データ接近法記述	
プログラムの実現		プログラム実現	—	—	—
				３５	２６

文　献

[1]Booch,G.:Software Engineering with Ada,The Benjamin/Cummings Publishing Company,Inc., pp.502(1983).

[2]Booch,G.:Describing software design in Ada, SIGPLAN Notices,(Sep. 1981).

[3]Booch,G.:Solve Process-Control Problems with ADA's Special Capabilities,EDN,pp.143-152(June 1982).

[4]Buhr,R.J.A.:System Design with Ada,Prentice-Hall,Inc.,pp.256(1984).

[5]Abbott,R.J.:System Design by Informal English Descriptions,Comm.ACM,Vol.26,No.11, pp.882-894(Nov. 1983).

[6]Myers,G.J:RELIABLE SOFTWARE THROUGH COMPOSITE DESIGN,Mason/Charter Publisher, Inc.,(1975).

[7]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,pp.934(1984).

[8]Wolfe,A.:SOFTWARE PRODUCTIVITY MOVES UPSTREAM,Electoronics,July,10,pp.80-86 (1986).

[9]Kuo,H.C.,Li,C.H. et al.:A FORM-BASED APPROCH TO HUMAN ENGINEERING METHODOLOGIES, Proc.6th ICSE,pp.254-263(Sep. 1982).

[10]Ross,D.T.,Scoman Jr,K.E.:Structure analysis for requirements definition,IEEE Trans. Softw. Eng.,Vol.SE-3,No.1,pp.6-15 (Jan. 1977).

[11]DeMarco,T.:Structured Analysis and System Specification,Prentice-Hall Inc.,1979.

[12]ANSI:Reference Manual for the Ada Programming Language,ANSI/MIL-STD-1815A-1983(1983).