動的解析によるWebアプリケーション・モデル抽出支援手法

安部麻里[†,††] 福田健太郎[†] 堀雅洋[†††]
田井秀樹[†] 根路銘崇[†]
小野康一[†] 大野義夫[††]

既存 Web アプリケーションの再構築に際し，MVC に基づくモデルを利用した開発へ移行することで，今後の開発効率および再利用性の向上が期待される．しかし，サーバ側リソースの解析だけではアプリケーション全体の振舞いが明らかにならない．本論文では，Web アプリケーションの振舞いを，HTTP 上で授受される情報に着目した動的解析により明らかにし，モデルの抽出を支援する手法を提案する．実際の Web アプリケーションを対象としたモデル抽出実験の結果，提案手法を用いることにより，Web アプリケーションの振舞いを表すモデルの抽出が可能であることが明らかとなった．

Elicitation Method for Web Application Models Based on Dynamic Analysis

Mari Abe,[†,††] Kentarou Fukuda,[†] Masahiro Hori,[†††]
Hideki Tai,[†] Takashi Nerome,[†] Kouichi Ono[†]
and Yoshio Ohno[††]

Web application development using an MVC-based Web application model improves efficiency and reusability when restructuring an existing Web application. However it is difficult to clarify the whole behavior of a Web application only with analysis of the server-side resources. We propose a method for eliciting an MVC-based Web application model clarifying the behavior of the application by using dynamic analysis focusing on the HTTP information exchanges. We experimented with eliciting models from actual Web applications. The results show that it is possible to elicit models that express the behavior of Web applications by using the proposed method.

1. はじめに

World Wide Web（以下，Web）は，静的ページ（HTML）を中心とした情報発信を実現する手段にとどまらず，オンライン取引やビジネス業務等，アプリケーション提供手段として広く用いられている．近年，Web アプリケーション開発には，Model 2 アーキテクチャ[1] と呼ばれる MVC（Model-View-Controller）パラダイムに基づく開発手法が用いられつつある．たとえば，MVC に基づく Web アプリケーション・フレームワーク[2),3)] を用いることで，アプリケーションに固有な処理の開発に専念することが可能となり，開発期間の短縮が期待できる[4)]．さらに，MVC に基づいて Web アプリケーションのモデル化を行い，設計情報として用いる手法が提案されている[5)~7)]．これらの手法においては，Web アプリケーションの振舞い，すなわち Web ブラウザからのリクエスト処理，アプリケーションの状態やデータの変更，次ページへの遷移，表示用のデータ送信，という一連の処理の流れを，実行環境に依存しない表現で定義し，設計段階での不整合検出や，フレームワークの利用，並行開発の実現による作業の効率化を目指している[8)]．

また，実行環境の移行や機能追加等アプリケーションの再構築が必要となった場合においても，MVC に基づいたアプリケーションでは，構成要素が機能ごとに分類され，その依存関係や振舞いが明らかにされているため再利用が容易である[4)]．

しかし，既存の Web アプリケーションの中には

† 日本アイ・ビー・エム株式会社東京基礎研究所
Tokyo Research Laboratory, IBM Japan
†† 慶應義塾大学大学院理工学研究科
Graduate School of Science and Technology, Keio University
††† 関西大学総合情報学部
Faculty of Informatics, Kansai University

MVC に基づいて開発されていないものも多数存在する．これらのアプリケーションの再構築に際し，MVC に基づくモデルを利用した開発へ移行することで，今後の開発効率および再利用性の向上が期待される．

これまで，Web アプリケーションの再構築支援手法としては，業務ロジックやページデザイン等の資源の再利用を支援するための研究がなされてきた[9)~11)]．たとえば，文献 11) では，Web アプリケーションの構成の把握を支援するために，リバースエンジニアリング手法を用いた視覚化手法が提案されている．この手法では，Web サーバにおけるアプリケーションの構成要素を，静的ページ（HTML 等），動的ページ（ASP，JSP☆等），Web オブジェクト（CORBA，EJB 等），データベース等に分類し，各要素に対して解析プログラムを用意し解析を行う．次に，それぞれの解析結果を集約することにより構成要素間の依存関係を明らかにしている．

このように，リバースエンジニアリング手法を用いることで，Web アプリケーション構成要素間の依存関係の把握や，構成要素の再利用を支援することが可能となる[9)~11)]．しかしながら，Web アプリケーションは Web ブラウザから送信されたリクエストに基づいてサーバ側の構成要素を呼び出すことにより実行される．このようなリクエストを介した参照関係は，構成要素のみの解析からは把握することが難しく，従来のリバースエンジニアリング手法では Web アプリケーション全体の振舞いが明らかにされないという問題点が指摘されている[12)]．

この主な原因として，Web アプリケーションは一種のサーバ/クライアント型アプリケーションであること[13)] があげられる．サーバ/クライアント型アプリケーションでは，サーバ側リソースだけでなくサーバ，クライアント間の通信内容やクライアント側のリソースを含めた動的解析を行わなければ，アプリケーション全体の振舞いが明らかとならない[14)]．

そこで本論文では，既存 Web アプリケーションの実行時の振舞いを，HTTP 上で授受される情報に着目した動的解析により明らかにし，MVC に基づくモデルの抽出を支援する手法を提案する．本手法の対象とする Web アプリケーションは，業務ロジックがサーバ側で実行され，クライアントに相当する Web ブラウザ側ではフォームおよびハイパーリンクを用いたリクエスト送信が実行されるような Thin Web Client[15)] 型の Web アプリケーションである．

提案手法では，まず，HTTP 上で授受される情報の中から実体ページ，すなわち HTML 文書をそれぞれの類似度等に基づいてグループに分類する[16)]．次に，各グループ間の呼び出し関係および HTTP リクエストとして送信される各種パラメータを解析し，Web アプリケーション全体の処理の流れを表すページ・フローを導出する．ここで得られたページ・フローにより，ページ間の遷移関係，サーバ側ロジックの呼び出し関係，パラメータを介したインタフェース等，Web アプリケーション全体の振舞いが明らかとなる．提案手法では，ページ・フローと Web アプリケーション・モデルのスキーマを照らし合わせることにより，MVC に基づくモデルを抽出している．

本論文では，提案手法を実装したモデル抽出支援ツールを用いた実験により，実際の Web アプリケーションから，その振舞いを表す Web アプリケーション・モデルの抽出が可能であるか否かを検証した．実験では，仕様書等の情報はいっさい用いず，実際の Web アプリケーションの実行結果のみに基づいてモデル抽出を行った．抽出されたモデルの検証にあたっては，Web アプリケーションの要件定義のうち，本手法で抽出可能でありモデル上での記述が可能である項目を対象に，要件定義の充足可否に基づいて評価を行った．実験の結果，提案手法を用いることにより，より多くの要件定義を満たすモデルの抽出が可能となることが明らかとなった．

以下，2 章で提案手法であるモデル抽出支援システムについて説明し，3 章で提案手法を用いた実験について述べ，実験結果の検討を行う．最後に 4 章で本論文についてまとめる．

2. モデル抽出支援システム

本章では，提案手法であるモデル抽出支援システムの構成と，システム内で実体ページ・フローの生成に利用される遷移情報解析のアルゴリズムについて説明を行う．ここで実体ページ・フローとは提案システムで用いる中間表現で，HTML 文書とその呼び出し関係を記述し MVC との対応関係を表すものであり，HTML 文書やサーバ・ロジック呼び出しをグループ分割した結果を保持する．また，本章では提案手法を用いたモデル抽出支援ツールの実装についても述べる．

2.1 システム構成

本論文で提案するモデル抽出支援システムの構成を**図 1** に示す．提案システムは Transaction Recorder，Analyzer，Model Generator，および開発者がこれら

☆ Java およびすべての Java 関連の標語およびロゴは，米国 Sun Microsystems, Inc. の米国およびその他の国における商標または登録商標である．

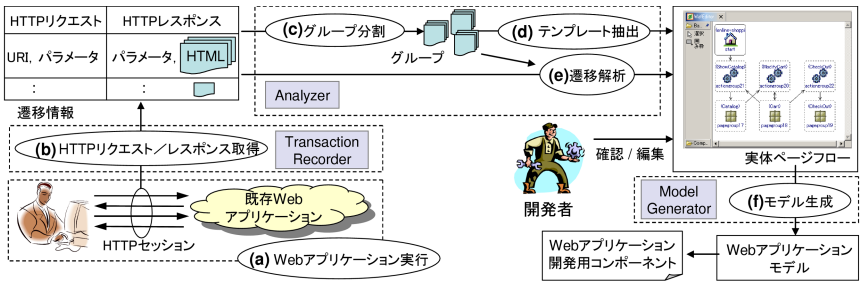

図 1 システム構成
Fig. 1 System architecture.

表 1 MVC と提案システムとの関連
Table 1 Correspondence of MVC and proposed system.

MVC	提案システム（Analyzer）		抽出 Web アプリケーション モデル	開発用 コンポーネント
	解析 モジュール	解析対象		（Struts の場合）
Model	遷移解析	URI, パラメータ	サーバ・オブジェクト	内部ロジック雛形（EJB 等），FormBean
View	グループ分割，テンプレート抽出	HTML 文書，パラメータ	サーバ・ページ	JSP 雛形，TagLib 宣言等
Cont-roller	遷移解析	URI，グループ分割結果	ページ・フロー，サーバ・ロジック呼出/分岐	構成ファイル（struts-config 等），Action クラス雛形

のモジュールを操作し解析結果を確認/調整するための GUI から構成される．また，提案システムと Web アプリケーションの MVC との関連を**表 1** に示す．

まず，Transaction Recorder は，Web アプリケーションを実行した際の HTTP リクエスト/レスポンスを，URI やクエリ引数，POST されたデータ等の各種パラメータと HTML 文書に区分し，遷移情報として保存する（図 1 (a)，(b)）．ここで，アプリケーションを実行するのは，システム開発者でなく一般のアプリケーション利用者でもかまわない．

Analyzer は，Transaction Recorder で取得した遷移情報を解析することにより，MVC の各要素を抽出する．具体的には，HTML 文書群を構造/役割の類似度に基づいてグループに分割し（図 1 (c)），同一グループに分離されたページ群から表示形式の雛形や入出力データを抽出する（図 1 (d)）．この結果，MVC における View に相当するサーバ・ページが抽出される．次に，遷移情報中の URI とグループ分割の結果に基づいて解析を行い，グループ間の呼び出し関係を明らかにする（図 1 (e)）．遷移解析の結果，MVC における Model に相当するサーバ・オブジェクト，および Controller に相当するページ・フローやサーバ・ロジック呼び出し/分岐が抽出される．

以上の解析結果に基づいて，Analyzer は実体ページ・フローを生成する．開発者は実体ページ・フローエディタ上で，実体ページやサーバ・ロジック呼び出しの属性を確認したり，ページ・フローの調整を行ったりすることも可能である．ただし，開発者が初期グループ分割結果に変更を加えた場合は，該当する箇所に対しテンプレート抽出（図 1 (d)）および遷移解析（図 1 (e)）が再度実行される．

開発者による確認/調整が終了した後，本システムは実体ページ・フローと Web アプリケーション・モデルのスキーマを照らし合わせ，モデルを抽出する（図 1 (f)）．抽出した Web アプリケーション・モデルを用いることで，様々な開発用コンポーネントを生成することができる．たとえば，Struts [2] を対象フレームワークとした場合の開発用コンポーネントの例を表 1 の最右列に示す．ここで，MVC における Model に相当する内部ロジック（EJB 等）に関しては，従来のリバースエンジニアリング手法[11] を用いた解析結果を再利用することも可能である．

2.2 Analyzer における解析

本節では，HTTP リクエスト/レスポンスを記録した遷移情報から，HTML 文書のグループ分割，グループからのテンプレート抽出，および遷移解析を行う Analyzer の詳細について述べる．

2.2.1 グループ分割

グループ分割モジュール（図 1 (c)）では，HTML 文書間のページ構造/役割に関する類似度に基づいて HTML 文書群をグループに分割し，MVC における View に相当するサーバ・ページの抽出を実現する．

本論文では，HTML 文書の類似度を導出するため，TABLE，TD，DIV 等のレイアウトに影響を与えるタグ（以降，レイアウトタグ）の構造を比較し，コンテンツ間の距離を数値化する手法[17] を利用した．ただし，文献 17) の手法はレイアウト，すなわちページ構造が同一か否かを判定することに着目していたため，本論文ではページ構造および役割に関する類似度を導出可能とするための拡張を行った．以下では，HTML

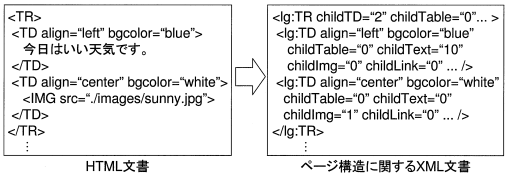

図 2 レイアウトタグの抽出
Fig. 2 Extraction of layout-tags.

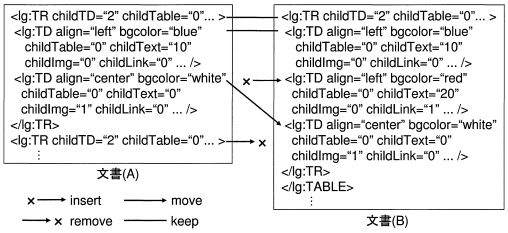

図 3 レイアウトタグの対応関係の導出
Fig. 3 Finding mapping between layout-tags.

文書の類似度導出手法の概要および，本論文で行った拡張について説明を行う．

HTML 文書の構造に関する類似度の導出では，まず，HTML 文書からレイアウトタグのみを抽出した XML 文書を作成する．この際，各タグの属性値，当該タグのサブツリー内のエレメントに関する情報等を特徴値として関連付ける（**図 2**）．次に，本論文ではコンテンツの類似度をより詳細に導出するため，差分計算[18] によりレイアウトタグの対応関係を明らかにした（**図 3**）．最後に，これらの対応関係を用いて文書 (A)，(B) 間の距離 D を以下の式により導出する．

$$D = \sum_{i=0}^{O} \Big\{ W_c(X_i) * \big(m(R_i) + \sum_{j=0}^{K} f(A_{ij}, B_{ij}) \big) \Big\} \quad (1)$$

式 (1) 中，O は 2 文書間のタグの対応関係の総数である．また，i 番目の対応関係におけるタグを一意に指示する XPath[19] 表現の組を X_i，対応関係の種類を R_i としている．さらに，タグ i に関連付けられる特徴値の総数を K，文書 (A)，(B) 中でのタグ i における j 番目の特徴値をそれぞれ A_{ij}，B_{ij} とする．

また，式中の W_c はタグの種類および文書内での位置 X_i に基づく重み付けで，タグツリーの上位に位置するほど大きな値をとる．次に，$m(R_i)$ は文書間でのタグの対応関係に関する距離を返す関数で，以下のような条件を満たすものとする．

$$\begin{aligned} m(R_i) &= 1 \quad (\text{insert}, \text{remove}) \\ 0 < m(R_i) &< 1 \quad (\text{move}) \\ m(R_i) &= 0 \quad (\text{keep}) \end{aligned} \quad (2)$$

すなわち，$m(R_i)$ は対象となるタグが一方の文書にしか存在しない場合には 1，双方の文書で同一の位置に存在する場合には 0，移動している場合にはその移動量に応じて 0 から 1 の間の値を返す関数である．

最後に，式 (1) 中の $f(A_{ij}, B_{ij})$ は文書 (A)，(B) 間の特徴値の差異に応じた距離を与える関数で，特徴値の差が大きい場合ほど大きな値を返す．また，$f(A_{ij}, B_{ij})$ は以下の式に示すように，その総和が 1 を超えないよう設定する．

$$0 \leq \sum_{j=0}^{K} f(A_{ij}, B_{ij}) \leq 1 \quad (3)$$

ただし，対象となるタグが一方の文書にしか存在しない場合には，タグに関連付けられた特徴値のうち，当該タグの属性値に関する距離はすべて最大値をとるものとし，当該タグのサブツリー内のエレメントに関する特徴値に関しては，比較対照が空のサブツリーであるものと仮定して距離を導出している．

以上の式を用いることにより，HTML 文書の構造に関する類似度を数値として導出することが可能となる．

次に，本論文ではフォームの送信方法や送信先 URI 等のパラメータに着目して HTML 文書の役割に関する類似度を求める手法を導入した．本手法では，HTML 文書から FORM タグおよびその構成要素である INPUT，SELECT 等を抽出し，それぞれの属性やタグ内の文字列等を特徴値として関連付け，文書間で比較することにより類似度を求める．ただし，役割に関する類似度においてはタグの出現順序等，HTML 記述上での構造の相違に関しては無視するものとした．また，HTML 文書内に含まれるリンク先 URI を役割に加味することも可能である．

以上の手法により導出された HTML 文書の構造および役割に関する類似度を用いることで，HTML 文書のグループ分割が可能になる．たとえば，類似度に閾値を設けること等により自動分割を行うことも可能であるが，対象となる Web アプリケーションの特徴により，適切な閾値の値や，構造・役割の距離に対する重み付けは変動する．このため，最適な分割結果を自動で得ることは難しい．そこで，分割結果を GUI を用いて開発者に提示し，閾値や重み付けの調整および結果の確認を行うことで，グループ分割の精度を高めることとした．GUI を利用した確認・調整ツールの詳細に関しては 2.3 節で述べる．

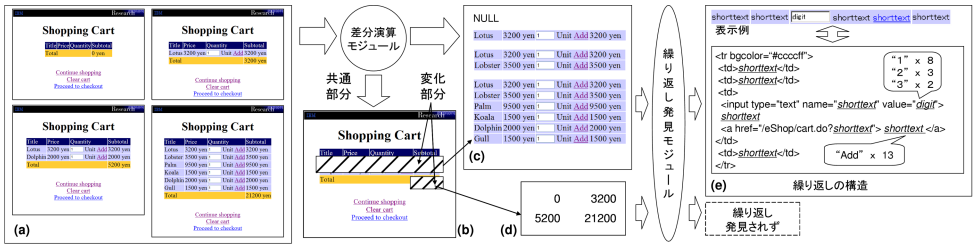

図 4 差分演算およびコンテンツ抽象化を用いたページテンプレートの抽出
Fig. 4 Extraction of design templates based on the difference algorithm and contents abstraction.

2.2.2 テンプレート抽出

テンプレート抽出モジュール（図 1 (d)）では，2.2.1 項の手法を用いて分割された HTML 文書のグループにおいて，ページ全体のテンプレート，特定表示形式の繰返し，および状況に応じて変更される部分を特定することで，MVC における View に相当するサーバ・ページの再構築や，View と Model 間の呼び出し関係の抽出を支援する．

まず，グループ化された HTML 文書群において，グループ内で共通に利用されている固定部分を差分演算[18] により抽出する．**図 4** (b) はグループ内（図 4 (a)）で共通に利用されていた部分を HTML のテンプレートとし，状況により変化が生じる部分を斜線を用いて表示したページテンプレートの例である．

次に，状況により変更が発生した部分（図 4 (c)，(d)）に対し，コンテンツ抽象化手法[20] を用いて特定書式の繰返しを検出し，その書式を抽出する．ここで，発見された繰返し表現の書式とその表示例を開発者に提示し，編集・確認を行うことで表示形式のテンプレートとして用いることができる（図 4 (e)）．また，書式中で抽象化されている部分（図 4 (e) 下線部）に対し，実際に用いられた値（図 4 (e) 吹き出し部分）を開発者に提示することで，書式の修正や再構成を支援する．

最後に，状況により変更が発生するが繰返しには属さない部分（図 4 (d)）は，アプリケーションの利用状況や時刻等に応じて変更される部分（Welcome メッセージや買い物かごの合計金額等）と判断できる．

以上の分類を行うことにより，サーバ・ページの再構築に際し，既存 Web アプリケーションで利用されているページの書式を再利用することが可能となる．

2.2.3 遷 移 解 析

遷移解析モジュール（図 1 (e)）では，Transaction Recorder が取得した遷移情報と 2.2.1 項で得られたグループ分割の結果から，MVC における Model に相当するサーバ・オブジェクト，および Controller に相当するサーバ・ロジック呼び出しやその入力変数，サーバ・ロジック分岐を導出する．

まず，ある 1 つのグループから呼び出される URI 群から，共通の URI を抜き出し，その集合を 1 つのサーバ・ロジック呼び出しと考える．この際，遷移情報内のパラメータから得られるクエリの引数や POST データ等は，サーバ・ロジック呼び出しの入力変数および，サーバ・オブジェクトの保持する変数として保存する．

さらに，既存アプリケーション固有の知識を用いて URI を解析することにより，サーバ・ロジック呼び出しを詳細に推定する．たとえば，URI のパスが “.html” で終端する場合は静的リンクとし，“.cgi” で終わる場合は CGI のスクリプト呼び出しと推定し，サーバ・ロジック呼び出しを割り当てる．

また，既存の Web アプリケーションにおいては，サーバ・ロジックがモジュール別に分かれておらず，複数のロジックが単一 URI パスで表現されている場合も存在する．そのような場合には，クエリの引数や POST データを利用してサーバ・ロジック呼び出しを分割する．たとえば，URI “`foo.cgi?userinfo=guest&opt=email`” と “`foo.cgi?item=book1&count=2`” において，URI パス（`foo.cgi`）は共通であるが，クエリの引数名の相違（`userinfo`，`opt` と `item`，`count`）から異なるサーバ・ロジック呼び出しと判断する．この際，解析に利用するパラメータを，正規表現等を用いて取捨選択することも可能である．

Web アプリケーションにおいて，サーバ・ロジックを実行した結果，遷移先のグループが複数存在することがある．たとえば，あるロジックを実行した結果，処理続行を促すグループに遷移する場合と，エラーを表示するグループに遷移する場合である．このような場合は，遷移先のグループ数に応じてサーバ・ロジック分岐数を割り当てる．

以上の解析によりグループ間の呼び出し関係が抽出され，Web アプリケーションの振舞いである実体ページ・フローが明らかとなる．

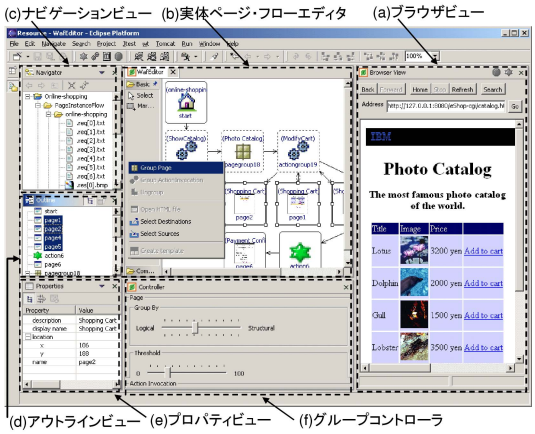

図 5 Web アプリケーションモデル抽出ツール
Fig. 5 Web application model elicitation tool.

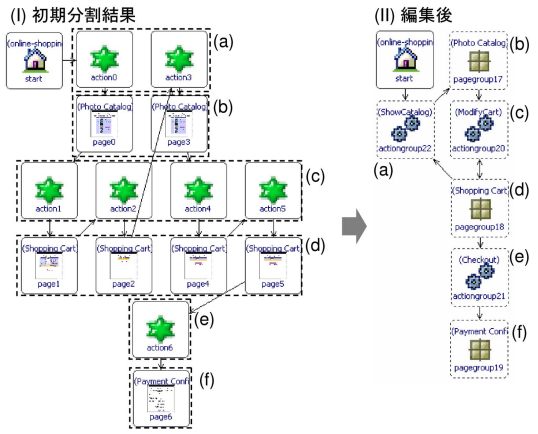

図 6 実体ページフローの編集
Fig. 6 Step of editing page instance flow.

2.3 モデル抽出支援ツール

我々は，提案システムを統合開発環境である Eclipse[21] 上にモデル抽出支援ツールとして実装した．このツールを用いることにより，モデル抽出やモデル抽出後の編集/開発を統一された環境で実施することが可能となっている（**図 5**）．

ツールを利用する際には，まず，ブラウザビュー（図 5 (a)）を用いて Web アプリケーションを実行する．実行内容は Transaction Recorder により遷移情報として記録され，ナビゲーションビューに表示される（図 5 (c)）．Web アプリケーションの実行が終了した時点で，解析実行を指示すると，Analyzer によるモデル抽出が実行され，生成された実体ページ・フローが，エディタ上に表示され（図 5 (b)），エディタ左側のアウトラインビュー（図 5 (d)）には，実体ページ・フローの概要が表示される．アウトラインビューを用いることで，各グループに含まれる実体ページ群やロジック呼び出し等がツリー表示により確認できる．

また，本ツールではグループコントローラ（図 5 (f)）を用いて 2.2.1 項のグループ分割における閾値や，構造/役割間の重み付けの変更を行いながら，実体ページ・フローを確認/調整することが可能である．この際，グループ分割の結果に影響を及ぼす閾値や重み付けの境界をあらかじめ計算し，グループ分割のパターンを開発者に提示することが可能である．

実際の実体ページ・フロー編集の様子を**図 6** に示す．図中の左上端にあるアイコンはアプリケーションの開始点を示している．図 6 (I) は，初期設定値に基づいて実体ページ・フローを生成した結果で，実体ページ（ページ画像のアイコン）やロジック呼び出し（星型のアイコン）がグループごとに横 1 列にレイアウトさ

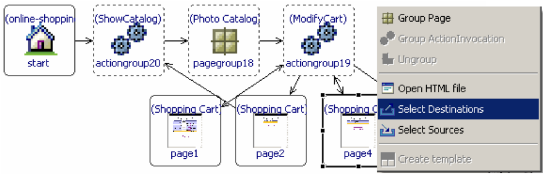

図 7 グループ化支援のためのメニュー
Fig. 7 Context menu for grouping.

れ，それぞれの呼び出し関係が矢印で表されている．図中，（b，d，f）は実体ページ，（a，c，e）はロジック呼び出しのグループの初期分割結果を示している．

次に，エディタ上のアイコンを選択すると論理名や座標等の属性がエディタ左下のプロパティビュー（図 5 (e)）に表示される．また，実体ページを表すアイコン中には，HTML の TITLE タグから抽出された名前や，各ページの画面イメージが表示され，グループを調整する際の指針に用いることができる．さらに，アイコンに対応する HTML 文書を実際に開いて確認することもできる．

また，本ツールでは Analyzer によるグループ分割結果を詳細に調整するため，各要素の遷移元や遷移先を選択するメニューが用意されている（**図 7**）．たとえば，異なる認証処理へのロジック呼び出しを行う類似したページが多数存在する場合，遷移先のロジック呼び出しが一般ユーザ認証用か管理者認証用か等に応じて，さらに詳細なグループ分割を行うことができる．

エディタ上での確認/調整が終了したグループに対し，グループ化メニューを選択することで，グループが決定され，実体ページは四角のアイコン，ロジック呼び出しは歯車のアイコンに置き換えられる．閾値や構造/役割間の重み付けの変更によるグループ分割結

果の調整は，グループが決定される前の実体ページおよびロジック呼び出しのみに適用される．

以上の作業を経て，グループ化された実体ページ（四角のアイコン）およびロジック呼び出し（歯車のアイコン）により実体ページ・フローが表現される（図 6 (II)）．なお，図 6 では，初期分割結果と編集後のグループの対応関係をアルファベットで示している．

最後に，実体ページ・フローを入力として Model Generator（図 1 (f)）を起動する．現在，Model Generator はスキーマとして WAD[5] もしくは UX model[15] のいずれかを選択し，Web アプリケーション・モデルを抽出できるよう実装されている．たとえば，グループ化された実体ページは，WAD における LogicalPage に対応するというように，実体ページ・フローと Web アプリケーション・モデルのスキーマを照らし合わせることにより，Web アプリケーション・モデルを抽出している．このようにして抽出されたモデルから Action クラスの雛形等開発用コンポーネントを生成することが可能となる[8]．

3. Web アプリケーション・モデル抽出実験

本章では，提案手法を利用することにより，Web アプリケーションの振舞いを表すモデルの抽出が可能となるか，実験を通して検証を行う．実験では，仕様書等の情報はいっさい用いず，Web アプリケーションの実行結果のみに基づいてモデル抽出を行った．また，比較の対象として，提案手法を用いずに，モデル・エディタ上[8] で直接モデルを作成する手法に関しても実験を行った．抽出されたモデルの検証にあたっては，Web アプリケーションの要件定義のうち，本手法により抽出可能であり，モデル上での記述が可能である項目を対象に，その充足可否に基づいて評価を行った．

以降，3.1 節では実験対象となる Web アプリケーションの概要を述べ，3.2 節で実験方法の詳細について説明を行う．最後に，3.3 節で実験の結果について述べ，考察を行う．

3.1 対象アプリケーションと要件定義

実験では，対象アプリケーションとして，CGI を実行環境とする “オンラインショップ”，“掲示板” および “アンケート” の 3 種類を用いた．これらのアプリケーションには，ログオン認証，データの参照・検索・作成・更新・削除等，Web アプリケーションの基本的な機能が含まれている．また，フォームの基本的な機能をすべて利用している．

表 2 に，オンラインショップの要件定義の一部を示す．この要件定義は，実際の開発に用いられている

表 2 オンラインショップの要件定義例
Table 2 An example of requirement definition for online-shopping application.

種別	要件	従属項目	説明
画面要件	[A01] ログオン画面 (A02, A03)	◇ メッセージ	ログオン時のメッセージを表示
		ユーザ名入力フィールド	
		パスワード入力フィールド	
		送信ボタン	
	[A02] ログオン失敗画面 (A01)	◇ メッセージ	ログオン失敗時のメッセージを表示
		ログオン画面へのリンク	
	[A03]Welcome ページ (A04)	◇ メッセージ	ログオン成功時のメッセージを表示
		◇ ユーザ名	
		製品一覧画面へのリンク	
機能要件	[B01] ログオン認証	認証処理	
	[B02] 製品情報一覧取得	＊ 製品名称取得	製品一覧情報より製品項目リストを取得
		＊ 製品画像取得	
		＊ 製品単価取得	
	[B03] カート内容照会	◇ 製品名称取得	カート内容に含まれる製品項目より取得
		◇ 製品単価取得	
		◇ 製品個数取得	
		◇ 小計計算	小計は単価と個数を製品ごとに計算
		◇ 合計金額計算	小計の総和を計算
		カート ID 取得	
データ要件	[C01] 製品項目	製品 ID	
		◇ 単価	
		◇ 画像	
	[C02] 製品一覧	＊ 製品項目リスト	製品項目がリストとして登録されている
	[C03] ユーザ情報	ユーザ名	
		パスワード	
		◇ 配送先	

表 3 実験対象アプリケーションの概要
Table 3 A summary of Web applications evaluated.

	画面要件	機能要件	データ要件	対象要件合計
オンラインショップ	7(26) [7(14)]	6(14) [5(5)]	3(7) [2(3)]	14(22)
掲示板	9(45) [9(33)]	7(12) [6(8)]	3(10) [3(7)]	18(48)
アンケート	7(27) [7(21)]	3(4) [2(3)]	3(9) [2(8)]	11(32)

要件定義の一例に則して作成したものであり，画面要件，機能要件，データ要件に分類される．画面要件は MVC の View，Controller に，機能要件，データ要件は Model に相当する．また，各要件には鍵括弧で示される識別名が付加され，従属項目として要件の内容が与えられる．さらに，一部の要件には必要に応じて説明が加えられている．また，画面要件に対しては，遷移先画面の識別名が括弧内に記されている．

次に，各アプリケーションの画面要件，機能要件，データ要件の数を**表 3** に示す．表では，各要件の従属項目数の合計を括弧内に示している．たとえば，オンラインショップの画面要件は，7 個（すなわち 7 画面で構成される）であり，各画面に対する従属項目の合計が 26 個である．さらに，これらの要件定義のうち，提案手法により抽出可能であり，モデル上での記述や充足可否の判断が可能である要件数を表 3 の鍵括弧内に，その合計を表の最右列に示す．このように評価の

対象となる要件を絞った理由を以下に述べる．

まず，機能要件に関しては，内部ロジックの実装が完了した後でなければ要件を満たすか否か判断できない．そこで，本実験では，入力変数の有無等，ロジックの実装に必要な条件を満たすかどうかに基づいて評価を行うものとした．また，データ要件に関しては，システム内で生成可能なもの（日時や識別番号など）や，データベースに格納されている項目に関しては評価の対象から除き，HTTP 上で授受されるデータを対象にした．また，HTTP 内のパラメータとして明示的にリクエストが送信されない機能要件に関しても対象から除外した．これらの条件により，評価の対象外とした従属項目を ∗ で示した．

また，画面要件中の可変部分，たとえばログオンメッセージ等の生成に関連する項目（表 2 中 ◇）は，2.2.2 項のテンプレート抽出を用いることにより，実体ページ群から推定することが可能であるが，本実験では充足可否が明確に判断できる項目に重点を置き，これらの項目も評価の対象外とした．

3.2 実験方法

本実験では開発者として，Struts[2] を用いたアプリケーションの実装経験がない人（開発者 a，d），Struts を用いたアプリケーションの実装経験を持つ人（b，c，e，f）の合計 6 人の協力を得た．実験前の準備として，各開発者に簡単な Web アプリケーションからのモデル抽出を実施させ，提案手法を実装した抽出支援ツールの基本的な使い方等を習得させた．

実験では，開発者を 2 グループに分け，提案手法を利用する場合と利用しない場合のそれぞれについてモデル抽出を実行させた（**表 4**）．この際，実験結果がアプリケーションに対する知識に依存しないよう，3.1 節で述べた 3 種類の対象アプリケーションをオンラインショップ，掲示板，アンケートの順で一度ずつ用いた．

提案手法の利用にあたっては，グループ分割の初期値として，経験的に良い分割結果の得られる構造/役割間の重み付け（1:1）と閾値（距離の最大値の 30%）を用いて開発者に実体ページ・フローを提示した．開発者は，必要に応じて初期値から分割パターンを調整した後グループを決定し，モデル抽出を行った．

3.3 実験結果

本実験で各開発者が抽出したモデルの概要と，要件定義との関係を**表 5** に示す．また，抽出されたモデルの例として，オンラインショップ（S_T-f），掲示板（B_T-c），アンケート（Q_T-d）のモデルを**図 8**，**図 9**，**図 10** にそれぞれ示す．図中，四角いアイコンはグループ化された実体ページを，丸いアイコンはグループ化されたロジック呼び出しを示している．また，各サーバ・ページの画面要件名を吹き出しとして付加してある．

表 4 実験内容と提案手法利用の有無
Table 4 Contents of an experiment and whether proposed method is used.

開発者	実験内容		
	1 回目（オンラインショップ）	2 回目（掲示板）	3 回目（アンケート）
a, b, c	提案手法なし（S_N）	利用（B_T）	なし（Q_N）
d, e, f	提案手法利用（S_T）	なし（B_N）	利用（Q_T）

表 5 作成されたモデルと要件定義との関係
Table 5 Relation between designed models and their requirement definitions.

実験	開発者	ページ数	ロジック呼び出し数	要件定義充足数	誤りの原因（従属項目）	
					実行忘れ	モデル間違い
オンライン	a	7	2	12(19)	0	3
ショップ	b	7	6	14(22)	0	0
（S_N）	c	7	6	14(22)	0	0
オンライン	d	7	8	14(22)	0	0
ショップ	e	7	6	14(22)	0	0
（S_T）	f	7	6	14(22)	0	0
掲示板	d	7	9	13(42)	1	5
（B_N）	e	6	8	13(35)	10	3
	f	8	9	17(47)	0	1
掲示板	a	8	13	14(44)	4	0
（B_T）	b	9	10	18(48)	0	0
	c	9	12	18(48)	0	0
アンケート	a	7	2	7(28)	0	4
（Q_N）	b	8	5	6(27)	0	5
	c	7	4	7(28)	0	5
アンケート	d	7	5	11(32)	0	0
（Q_T）	e	7	5	11(32)	0	0
	f	7	5	10(31)	1	0

表 5 の左列は，生成されたモデル中のページ数およびロジック呼び出し数を示している．次に，各モデルが満たしている要件の数を表 5 の要件定義充足数に示す．ここで，括弧内は従属項目数を表しており，従属項目が 1 つでも満たされない要件は，要件を満たさないものとした．最後に，表 5 の右列には各開発者が要件を満たすことのできなかった原因をあげた．本実験でモデルが要件を満たせなかった原因は 2 種類存在し，HTML のリンクやボタン等をたどり忘れた場合（実行忘れ）と，実行はしたもののモデルとして記述し忘れたり，呼び出し関係の記述を間違えた等，モデルに間違いがあった場合（モデル間違い）であった．

表 5 の結果から，提案手法を利用しない場合に，モデル間違いが多く発生しているのに対し，提案手法を用いることでモデル間違いを防止できていることが分かる．たとえば，オンラインショップ（S_N，S_T）では，提案手法を利用しない S_N-a にモデル間違いが 3 カ所存在した．実験中，開発者 a はアプリケーションをくまなく実行しているにもかかわらず，相当するモ

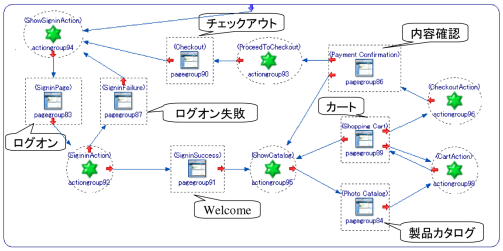

図 8 オンラインショップ（S_T-f）
Fig. 8 Online-shop (S_T-f).

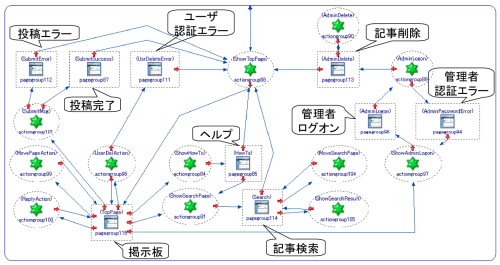

図 9 掲示板（B_T-c）
Fig. 9 Bulletin board (B_T-c).

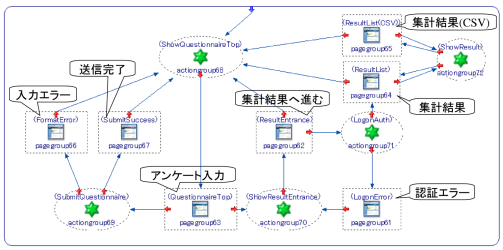

図 10 アンケート（Q_T-d）
Fig. 10 Questionnaire survey (Q_T-d).

デル構成要素を作成していなかった．一方，提案手法を利用した場合には，実行したページやロジック呼び出しは必ず実体ページ・フロー上に出現するため，このようなモデル記述の抜け落ちを防ぐことができる．

また，提案手法を利用せず掲示板，アンケートのモデル抽出を行った場合，全員が何らかの形でモデル間違いを発生している（B_N，Q_N）．このようなモデル間違いの原因としては，提案手法を用いない場合には，受け渡されるパラメータの見落としが発生することや，実体ページやロジック呼び出しの役割を把握することが困難であること等があげられる．

特に掲示板は，Web アプリケーションの全ページで同一のタイトル “掲示板” が用いられており，HTML のタイトルから要件を推測することが困難であった．さらに，ページのデザインも見た目では区別がつきにくく，ロジック呼び出しはすべて同一 URI パスを用いたうえでクエリのみが異なるものであった．このため，開発者がアプリケーションを実行しながらモデルを記述している間に，複数のサーバ・ページやロジックを混同してしまい，結果として誤ったモデルを生成していた．

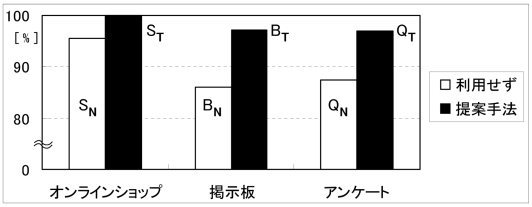

図 11 要件定義充足率（従属項目）
Fig. 11 Percentage of requirement accomplished.

一方，提案手法を利用した場合（B_T）においても，ページのデザインや呼び出し URI が類似しているため，初期分割結果では複数のサーバ・ページやロジックが混同されて表示される．しかしながら，実体ページ・フローエディタにおいて，グループ分割における閾値変更機能や，遷移元/先確認機能を用いることにより，各要素の詳細な役割や振舞いを把握し，実体ページ・フローを調整することが可能である．実際，開発者 a，b，c はこれらの機能を利用して実体ページ・フローの調整を行い，結果として多くの要件を満たすモデルを抽出している（表 5）．

以上のように，提案手法を用いることで Web アプリケーションの振舞いの把握を支援し，正確なモデルの抽出を可能としていることが分かる．この結果，提案手法を用いた場合に充足可能な要件（従属項目）数の平均値は，提案手法を用いない場合と比較して高い値となっている（**図 11**）．

ただし，本実験では提案手法の利用の有無を問わず複数の開発者に実行忘れが見られた（表 5）．実行忘れが発生すると，対応するサーバ・ページやロジックに関する情報が欠落してしまうため，要件定義の充足に大きな問題を引き起こしてしまう．実行忘れの多くは，ページ内に判別しにくいリンクが存在する，ページ内に多数のリンクが存在する，フォーム内の選択肢の組合せが多岐にわたる等の原因で発生する．たとえば，掲示板において開発者 a，d，e に共通した実行忘れは，図 9 中のヘルプページから記事検索ページへのリンクのたどり忘れであり，その原因としては，このリンクがページ上ですぐにはリンクと判別しにくいものであったことがあげられる．このような実行忘れを防止するためには，Web アプリケーション実行中に

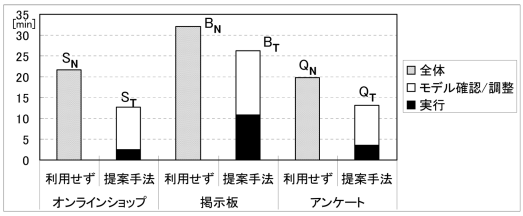

図 12 モデル作成の所要時間
Fig. 12 Time spent for model design.

HTML 文書の解析を行い，まだ実行していない URI に対し警告を表示したり，自動実行を行ったりする等の方法が有効であると考えられる．

最後に，実験所要時間の平均を**図 12** に示す．提案手法を利用した場合については，作業内容をアプリケーション実行とモデルの確認/調整の 2 項目に分けて結果を示している．図から，提案手法を用いることにより，3 種類のアプリケーションいずれにおいてもモデル抽出にかかる所要時間が短縮されることが分かる．特に，遷移元/遷移先等の情報を用いた調整を必要としないオンラインショップでは，所要時間が約 40%短縮されている．また，詳細な調整作業が必要となる掲示板，アンケートにおいても，作業時間は約 20%以上短縮された．

以上の結果から，提案手法を利用することで，Web アプリケーションの振舞いの把握を支援し，より多くの要件定義を満たすモデルを，短時間で抽出することが可能となることが明らかとなった．

4. ま と め

本論文では，Web アプリケーションの振舞いを，HTTP 上で授受される情報に着目した動的解析により明らかにし，MVC に基づくモデルの抽出を支援する手法を提案した．実験の結果，提案手法を用いることにより，Web アプリケーションの振舞いを表すモデルの抽出を効果的に支援できることが明らかとなった．

今後の課題としては，大規模なアプリケーションでは，たとえばログオン等の特定シナリオに限定してモデル抽出を行う等，実行単位を区分して扱う仕組みを実装する必要があると考えられる．このような区分は，実体ページ・フロー抽出の精度を向上させるうえでも有効である．さらに，アプリケーション実行時の画面操作を再現する機能[22] 等も，ページ・フローの調整に有用であると考えられる．

また，提案手法と従来のリバースエンジニアリング手法とをより緊密に連携させることにより，さらなる再利用性の向上が期待される．今後，検討を行っていきたい．

参 考 文 献

1) Singh, I., Stearns, B., Johnson, M. and Enterprise Team: *Designing Enterprise Applications with the J2EE Platform*, 2nd Edition, Addison-Wesley (2002).
2) Apache Software Foundation: Jakarta Struts Web Application Framework. http://jakarta.apache.org/struts/
3) Apache Software Foundation: Jakarta Turbine Web Application Framework. http://jakarta.apache.org/turbine/
4) 津久井浩，中所武司：Web アプリケーションにおける予約業務フレームワークの実現と再利用性の評価，オブジェクト指向最前線 2003，pp.33–40，近代科学社 (2003).
5) 堀 雅洋，田井秀樹：モデルに基づく Web アプリケーション開発，情報処理学会学会誌，Vol.45, No.1, pp.16–21 (2004).
6) Fraternali, P. and Paolini, P.: Model-driven development of Web applications: The Autoweb system, *ACM Trans. Inf. Syst.*, Vol.18, No.4, pp.323–382 (2000).
7) 位野木万里，山田広佳：Web アプリケーション開発における設計・設計検証・テストプロセスの提案，情報処理学会研究報告 SE-143-007, pp.45–52 (2003).
8) 田井秀樹，根路銘崇，安部麻里，堀 雅洋：モデルに基づく Web アプリケーション開発支援環境，情報処理学会論文誌，Vol.44, No.6, pp.1498–1508 (2003).
9) Tilley, S. and Huang, S.: Evaluating the Reverse Engineering Capabilities of Web Tools for Understanding Site Content and Structure: A Case Study, *Proc. 23rd ICSE 2001*, pp.514–523 (2001).
10) Antoniol, G., Canfora, G., Casazza, G. and Lucia, A.D.: Web Site Reengineering Using RMM, *Proc. 2nd International Workshop on Web Site Evolution* (2000).
11) Hassan, A.E. and Holt, R.C.: A Visual Architectural Approach to Maintaining Web Applications, *Software Visualization*: *From Theory to Practice*, Kluwer Academic Pub. (2003).
12) Han, M., Hofmeister, C. and Nord, R.L.: Reconstructing Software Architecture for J2EE Web Applications, *Proc. 10th WCRE'03*, pp.67–79 (2003).
13) Conallen, J.: Modeling Web Application Architectures with UML, *Comm. ACM*, Vol.42, No.10, pp.63–70 (1999).
14) Salah, M. and Mancoridis, S.: Toward an Environment for Comprehending Distributed Sys-

tems, *Proc.10th WCRE '03*, pp.238–247 (2003).
15) Conallen, J.: *Building Web Applications with UML, 2nd Edition*, Addison-Wesley (2002).
16) 安部麻里，福田健太郎，田井秀樹，根路銘崇，堀雅洋：動的逆解析による Web アプリケーション・モデルの抽出，日本ソフトウェア科学会第 20 回大会論文集，pp.546–550 (2003).
17) Fukuda, K., Takagi, H., Maeda, J. and Asakawa, C.: Layout Group Extraction from Web Content for Effective Adaptation, *IBM Research Report* (RT0493) (2002).
18) Lindholm, T.: A 3-way merging algorithm for synchronizing ordered trees — The 3DM merging and differencing tool for XML, Master's thesis, Dept. of Computer Science, Helsinki University of Technology (2001). http://www.cs.hut.fi/~ctl/3dm/
19) W3C Recommendation: XML Path Language (XPath) Version 1.0 (1999). http://www.w3.org/TR/xpath
20) Fukuda, K., Takagi, H., Maeda, J. and Asakawa, C.: An assist method for realizing a Web page structure for blind people, *Proc. UAHCI 2003*, pp.960–964 (2003).
21) Eclipse Foundation: Eclipse Platform. http://www.eclipse.org/
22) NetResults Corporation: WebVCR. http://www.netresultscorp.com/

(平成 16 年 6 月 21 日受付)
(平成 17 年 1 月 7 日採録)

安部　麻里（正会員）
平成 12 年慶應義塾大学大学院理工学研究科計算機科学専攻修士課程修了．同年より日本アイ・ビー・エム（株）東京基礎研究所勤務．現在，慶應義塾大学大学院後期博士課程在籍．Web アプリケーション開発環境の研究に従事．日本ソフトウェア科学会会員．

福田健太郎（正会員）
平成 12 年大阪大学大学院基礎工学研究科情報数理系専攻博士課程修了．同年より日本アイ・ビー・エム（株）東京基礎研究所勤務．アクセシビリティおよび Web アプリケーション開発に関する研究に従事．博士（工学）．電子情報通信学会，ヒューマンインターフェース学会各会員．

堀　　雅洋（正会員）
平成元年大阪大学大学院基礎工学研究科情報工学専攻博士課程修了．同年より日本アイ・ビー・エム（株）東京基礎研究所勤務．平成 15 年より関西大学総合情報学部教授．Web 情報の高度利用，認知モデルに基づくユーザビリティ評価の研究に従事．工学博士．平成 4 年度，9 年度人工知能学会研究奨励賞．人工知能学会，ACM 等各会員．

田井　秀樹（正会員）
平成 9 年筑波大学理工学研究科理工学専攻修士課程修了．同年より日本アイ・ビー・エム（株）東京基礎研究所勤務．ミドルウェアに関する研究およびアプリケーションの開発プロセスやモデリングに関する研究に従事．

根路銘　崇（正会員）
平成 8 年琉球大学大学院工学研究科情報工学専攻修士課程修了．同年日本アイ・ビー・エム（株）入社．現在同社東京基礎研究所勤務．Web アプリケーション開発環境の研究に従事．

小野　康一（正会員）
平成元年早稲田大学大学院理工学研究科修士課程修了．平成 2 年から 4 年まで同大学情報科学研究教育センター助手．平成 6 年同大学大学院理工学研究科後期博士課程単位取得退学．同年より日本アイ・ビー・エム（株）東京基礎研究所勤務．ソフトウェア工学，アプリケーション開発支援技術の研究に従事．IEEE-CS，ACM 等各会員．

大野　義夫（正会員）
昭和 48 年慶應義塾大学大学院工学研究科管理工学専攻博士課程単位取得退学．昭和 45 年同大学情報科学研究所に助手として入所．現在，同大学理工学部情報工学科教授．コンピュータグラフィックスのモデリングやレンダリングに関心を持つ．電子情報通信学会，芸術科学会，日本ソフトウェア科学会，IEEE-CS，ACM 等各会員．