ソフトウェア工学　136－2
（2002．3．7）

エンドユーザ主導型アプリケーション開発環境 M-base におけるユーザインタフェースの自動抽出・生成技法の実現と評価

横沢　邦一　　中所　武司
明治大学大学院理工学研究科基礎理工学専攻情報科学系

業務の専門家自らが情報システムを構築する必要性が高まっている。なかでも UI はコンピュータシステムのなかで重要な役割を担うようになった。本研究では、エンドユーザである業務の専門家によるアプリケーション開発時の UI 構築技法として、エンドユーザが構築した業務モデルから UI が必要となる箇所を自動で判断し、UI を自動生成するシステムを実現した。業務モデル内の各オブジェクトとそれに関連のある入出力メッセージとの関係を正確に認識することで、必要となる適切な UI とそれを構成する適切な UI 項目群を自動抽出し、UI の自動生成を行なう。また、それにより業務モデルの妥当性の検証も同時に行うことができる。さらに、エンドユーザの UI 生成履歴を用いることで、使えば使うほどエンドユーザに適応した UI の自動生成を可能にした。

Automatic User Interface Generation in Enduser-Initiative Application Development Environment M-base

Kunikazu YOKOZAWA　and　Takeshi CHUSHO

Computer Science Cource, Major in Sciences,
Graduate School of Science and Technology, Meiji University

Explosive increase in end-user computing on distributed systems requires that end-users develop application software by themselves. User interface plays an important part in distributed systems. In this paper, we propose automatic user interface generation technique. This system automatically extracts user interfaces from a domain model and verifies the model by the relation between objects and messages in the model. It also automatically generates user interface by user interface information that it extaracts.

1　はじめに

近年のコンピュータおよび、それらをつなぐネットワークの普及によってオフィスや家庭にもコンピュータシステムが積極的に導入されている。それに伴い、オフィス業務の効率化という観点から業務の専門家が自らの業務をシステム化し作業の負担を軽減する必要性が高まっている。

そこで我々は、「ドメインモデル ≡ 計算モデル」というコンセプト [1] の下、プログラミングの分からないエンドユーザでも自らアプリケーションを構築できるようなアプリケーション開発技法の確立を目指す。このコンセプトはドメインモデルを計算モデルとして扱うことにより設計やプログラミング作業を回避しようとするものである。その実現のために、我々はプロトタイピングアプローチを支援するエンドユーザ主導型アプリケーション開発環境 **M-base** を実現した [2–4]。

本研究では **M-base** の中でも UI の構築作業に着目する。近年の UI は機能性に加え、高い使用性が求められるようになり、UI の構築は年々複雑化の一途をたどっている。そのため UI の構築を支援

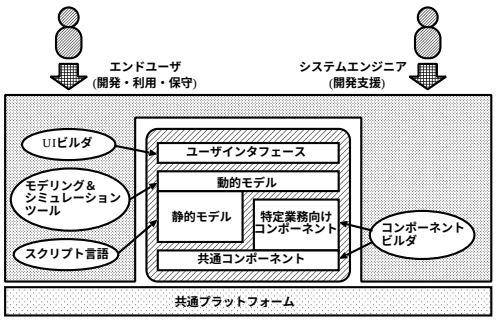

図 1: **M-base** の開発環境（外側）とアプリケーション・アーキテクチャ（内側）

するようなアプリケーションが数多く研究 [5–7] 、実用化されている。しかしプログラムの分からないエンドユーザにはまだ難しいものが多い。

M-base においても UI ビルダ [4] を用いることでエンドユーザでも UI を容易に構築できることを目指している。しかし、従来の UI ビルダは必要な情報を全て手作業で入力する必要があるため、情報が正確な反面、エンドユーザへの負担が大きい。

そこで、本システムでは UI 生成に必要な項目を全て自動で抽出し、UI を生成する。また、エンドユーザが今までに構築してきた UI の履歴情報を利用することで、エンドユーザに特化した UI の生成を行なう。これにより、ソフトウェア開発の専門家ではないエンドユーザでも簡単に誤りのないソフトウェアを作ることができる。

本稿では、2 章で **M-base** の概要を説明したあと、3 章で UI の自動抽出・生成技法の概要を述べ、4 章で例題を用いて実際の使用例を示す。5 章で UI 自動抽出・生成精度の評価を行なう。

2 アプリケーション開発環境 M-base

2.1 対象アプリケーション

M-base におけるエンドユーザとは、業務の専門家である。業務の専門家とは、例えばユーザ企業のエンドユーザ部門に所属し、市販のアプリケーションパッケージを利用しているような人たちを指す。また対象ソフトウェアとしては、オフィスでのワークフローシステムなどが中心となるが、市販のパッケージソフトでは対応できないような細かい機能の実現が必要な場合を想定している。規模的には中小規模が中心となるが、ネットワーク接続することによりシステムとしては大規模化することもある。

2.2 モデリングとアーキテクチャ

M-base ではエンドユーザ主導の開発を目的としているので、実際のモデリングは業務コンポーネントの組み合わせを基本としている。そのため、開発対象のアプリケーションのソフトウェア・アーキテクチャは図 1 の内側で示すように、ユーザインタフェース、モデル、コンポーネントウェアの三つの部分から構成される。ユーザインタフェースはユーザとの対話処理部分である。モデルと独立させることにより、クライアント／サーバシステムの構築、あるいはクライアント端末のマルチプラットフォーム化が容易になる。モデルはアプリケーションに固有の処理を行う本体であり、動的モデルと静的モデルからなる。コンポーネントウェアは分野に共通の基本部品と特定業務分野向きの部品がある。後者が充実してくるとエンドユーザによるアプリケーションの構築が容易になる。

2.3 開発環境

M-base の開発環境は、図 1 の外側部分に示すような 4 つのツールから構成される。モデリング&シミュレーションツール [3] は動的モデルの作成に用いる。動的モデルはオブジェクト指向の分散協調型モデルとして表現するので、オブジェクト間のメッセージフローの定義が重要な機能となる。スクリプト言語は静的モデルの定義に用いる。動的モデルから自動生成される場合と一から記述する場合とがある。UI ビルダはユーザインタフェースの構築に用いる。特徴として、モデリング&シミュレーションツールとの連係処理がある。本稿の UI 生成技法はこの UI ビルダに適用した。コンポーネントビルダはコンポーネント構築用のエディタである。コンポーネントは、既存部品の組合せによる再帰的な定義が可能である。

2.4 M-base におけるモデリングプロセス

M-base によるアプリケーション開発は以下の手順で行なわれる。

1. 業務仕様の詳細化
2. ドメインモデルの構築
3. オブジェクトの詳細化
4. ユーザインタフェースの構築
5. シミュレーションによる動作の確認、検証

まずエンドユーザはシステム化したい業務の仕様を詳細に分析し、それをもとにドメインモデル（業

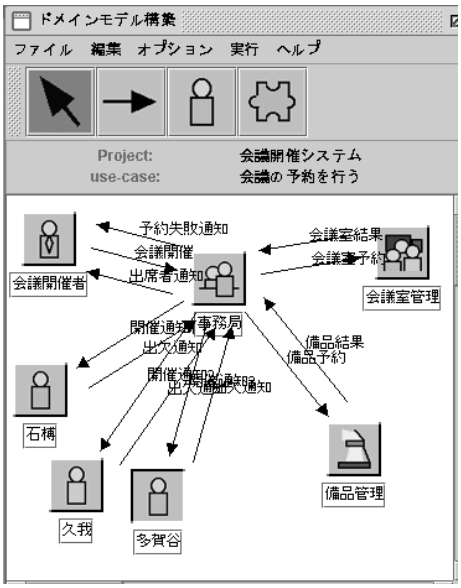

図 2: 業務モデル構築キャンバスと業務モデル例

務モデル）を構築する。その際は、業務仕様の詳細化の段階で決めた一つの処理のまとまりを一つのオブジェクトと考え、オブジェクト間を流れるメッセージとともにキャンバスに貼り付けていく。キャンバスは図 2 のようなものでマウス操作を基本としたドローツール風のものである。このなかで一つのオブジェクトは一つのアイコンで表され、オブジェクト間のメッセージは矢印で表現される。

次に、定義したオブジェクトの内部をスクリプトやルール記述を用いて詳細化する。さらに、オブジェクトのなかで人間の判断が必要ないオブジェクトを自動オブジェクトと指定する。

UI はこの自動オブジェクト以外のオブジェクトのメソッドごとに定義する。この段階に本技法を適用することで UI の定義から生成までを自動化する。そして完成した業務モデルをシミュレーションし、その動作を検証する。業務仕様を満たしたシステムを構築することができれば終了する。そうでなければ仕様を満たしていない手順に戻り改善する。

3 UI 自動抽出・生成技法

3.1 対象業務モデル

M-base における業務モデルは、オブジェクトとメッセージで構成される。オブジェクトにはひとつ以上のメソッドが存在し、このメソッドごとに UI が定義される。メッセージにはメソッドを起動する

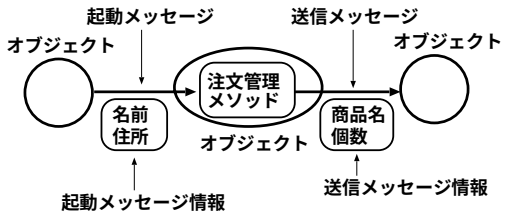

図 3: オブジェクトとメッセージ

起動メッセージと他のメソッドに向け送られる送信メッセージがある。各メッセージに付加される情報をそれぞれ起動メッセージ情報、送信メッセージ情報と呼ぶ。例えば、図 3 では名前と日付が起動メッセージ情報に、商品名と個数が送信メッセージ情報になる。

メソッドが起動された場合は、まずスクリプトが処理される。次に、UI が定義されていれば UI が表示され、ユーザからの入力を待つ。最後に送信メッセージの分岐ルールが実行され、送信メッセージが決定する。

本抽出技法ではスクリプトの解析が困難であるため、スクリプトを理解しなければ得られない情報は抽出の対象としない。例えば、スクリプトにより起動メッセージ情報の値を合計し、その値を UI で表示する場合、合計された値は抽出対象としない。また、処理の経過を表示するのみの UI についても抽出対象としない。これは、経過を表示する UI は全てのメソッドに付加することができるので、業務モデルの構造からだけでは、UI を必要とするメソッドを判断することが困難なためである。

3.2 UI の分類

業務モデルで用いられる UI は用途に応じて次のように分類される。

1. **送信メッセージ情報を入力する UI**
2. **ユーザに処理の結果を表示する UI**
3. **ルール分岐基準情報を入力する UI**

3.3 UI 抽出技法

分類した UI を抽出するために、以下の四つの観点から業務モデルを分析する。

1. **新規情報が必要である**

   分類 1 の UI を抽出できる。新規情報である送信メッセージ情報を入力項目とし、新規情報の値を決定するために必要な起動メッセージ情報を出力項目として UI を生成する。

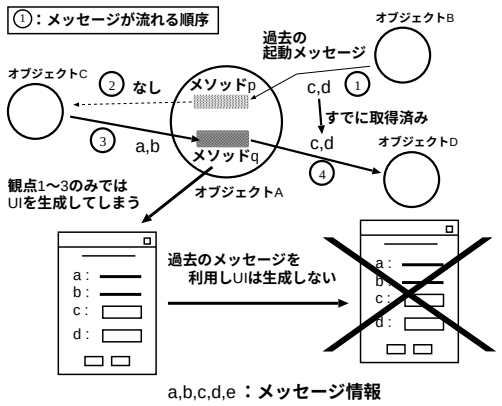

図 4: 過去の起動メッセージ情報も利用した UI 生成例

2. **起動メッセージのみである**

   分類 2 の UI を抽出できる。起動メッセージ情報を出力項目として UI を生成する。

3. **ルール分岐基準情報が必要である**

   分類 3 の UI を抽出できる。分岐基準情報を入力項目として UI を生成する。

4. **過去のメッセージ情報を利用する**

   上記三つの観点だけでは必要ない UI も抽出してしまう。例えば、図 4 のような場合、情報 **c,d** は過去の起動メッセージ情報からすでに取得しているため、この情報を取得する UI を抽出する必要はないことが分かる。

3.4 業務モデルの妥当性検証技法

UI を用いる業務モデルの検証方法として **M-base** では UI 遷移図を用いた手法がある [4]。しかし、この方法は UI をエンドユーザ自らが手作業で構築することを前提に考えられており、検証範囲も UI を構築した部分に限定される。本システムは UI を自動で抽出・生成するため、業務モデルの検証も自動で行なう。

本システムにおいて、妥当性がない可能性が高いと判断できる基準は次の二つである。

1. **メッセージ情報の消失**

   情報が何の処理にも利用されないまま途中のオブジェクトで消失している場合。例えば、図 4 のメソッド **p** は新規の送信メッセージ情報がないため、起動メッセージ情報 **c,d** がオブジェクト内で消失していると考えられる。

2. **冗長なメッセージ情報**

   起動メッセージ情報がそのまま送信元のオブジェクトに返されている場合。

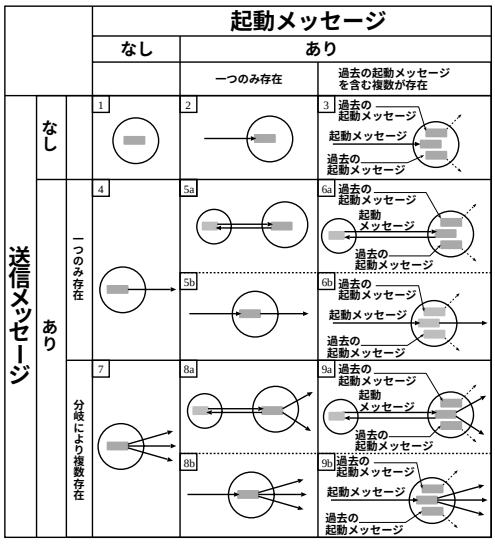

図 5: オブジェクトとメッセージの関係による分類

3.5 オブジェクトとメッセージの分類

3.3 項と 3.4 項で述べた基準で UI の抽出と妥当性の検証を行なうために、業務モデル内のオブジェクトと各メッセージ、そしてメッセージ情報を分類した。分類は、業務モデルの構造面から分類する**オブジェクトとメッセージの関係**と、特定の構造におけるメッセージ情報の内容により分類する**メッセージ情報**という二つの観点で行う。

オブジェクトとメッセージの関係による分類は、以下の基準で行なう。

- **起動メッセージの有無と数**
- **送信メッセージの有無と数**
- **起動メッセージの送信元に送信メッセージが返されている (図 5 の a)、いない (図 5 の b)**

さらに、これらの各々についてメッセージ情報の観点から以下の基準で分類する。

- **送信メッセージ情報が起動メッセージ情報を全て含むかどうか**
- **新規情報を含むかどうか**
- **ルールの分岐基準情報が起動メッセージ情報、または当該オブジェクトが持つ他の起動メッセージ情報に含まれていない (図 6 の a)、含まれている (図 6 の b)**

例えば、図 5 の 8a を分類すると図 6 のようになる。

このようにして図 5 の各々について調べた結果、合計 66 種類に分類された。そのうち、UI が必要と判断されたのは 42 種類であるが、その 12 種は業務モデルに何らかの誤りがあると考えられる。一方、

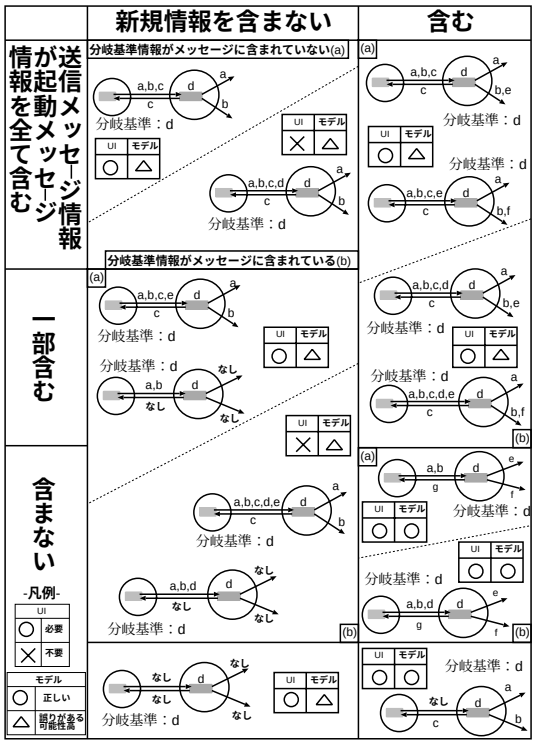

図 6: メッセージ情報による分類

UI が必要でないと判断された 42 種類についても 18 種はモデルに何らかの誤りがあると考えられる。

3.6 UI 自動生成技法

3.5 項の分類により抽出した UI に関する情報から、UI を自動生成する。そのために、本システムではエンドユーザの UI 構築手順 [8] から UI 構築プロセスを以下のように策定した。

1. UI 情報の収集
2. UI 情報のグループ化と優先順位の定義
3. UI 情報の可視化・インタフェース化
4. 構築した UI の確認と評価

4 UI 自動抽出・生成システム

4.1 エンドユーザによる UI 構築手順

本技法を利用した UI 自動抽出・生成は 2.4 項のモデリングプロセスの **4** において以下の手順で行なわれる。

1. 本システムによる UI の抽出
2. 業務モデルの妥当性の検証
3. エンドユーザによる業務モデルの確認と修正
4. 本システムによる UI の自動生成
5. 生成された UI の確認とカスタマイズ

図 7: UI 例

4.2 UI 情報

UI 情報は、各 UI 項目の項目名、型名、UI オブジェクト名、UI オブジェクトの大きさと色、項目間のグループ化・優先順位情報、配置情報から構成される。UI 項目とは UI 項目名と型名、UI オブジェクトの対である。例えば、図 7 のような UI があった場合、名前、住所、性別が UI 項目名であり、名前、住所を入力するための テキスト入力欄、性別を入力するラジオボタンが UI オブジェクトとなる。

本システムは HTML 言語を用いた Web ページを UI として生成するので、UI オブジェクトとしては、Button、CheckBox、Label、RadioButton、TextArea、TextField、選択リストの 7 種類である。型名は入力型が二択、選択、日付、数値、長文、短文、文字列、時間の 8 種類で、出力型がラベル、長文、短文、表の 4 種類である。

4.3 UI 履歴情報

ユーザに特化した UI を自動生成するために、ユーザが今まで構築してきた UI を UI 履歴情報として保存し、UI 生成に利用する。UI 履歴情報には型、項目名、UI オブジェクト、配置情報がある。型履歴情報には、各型の履歴情報として各型に割り当てられた項目名と UI オブジェクトの出現頻度を保持する。項目名履歴情報は、ある一つの項目名と型名の対に対応する UI オブジェクトの出現頻度を保持する。UI オブジェクト履歴情報には、各 UI オブジェクトの、サイズと色の出現頻度を保持する。UI オブジェクト履歴情報には項目名と型名が指定された詳細データと型名のみが指定された全体データがある。詳細データは、以前に同様の項目名から UI オブジェクトを推論した経験がある場合に利用し、全体データは項目名が初出の場合に利用する。配置履歴情報は後述する配置ルールの出現頻度と類似度の履歴を保持する。

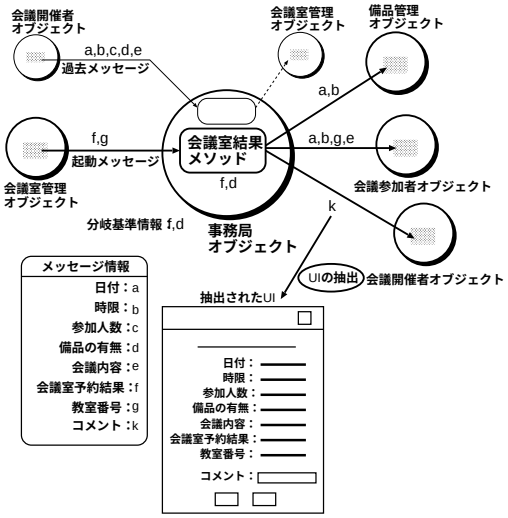

図 8: 会議結果メソッドと抽出される UI

4.4 例題の説明

例題として会議開催システムを取り上げ、実際に本システムで UI を自動抽出・生成しながら手順を説明する。まず、業務モデルを構築する。

会議開催システムは、会議開催者が会議を行ないたい日時や使いたい備品などを入力することで、会議室の予約や備品の手配、参加者への出欠の確認などの一連の作業を行なうシステムである。この会議開催システムには会議の予約と会議の取り消し、出席者の照会の三つユースケースが定義されている。そのうち今回は会議の予約ユースケースを使用する。図 2 は会議開催システムにおける会議の予約ユースケースの業務モデルである。抽出されたオブジェクトは 5 種類でメッセージも定義されている。このうち、会議室管理オブジェクトは人間の判断の必要のない自動オブジェクトと仮定し、UI 自動生成の対象外とする。

4.5 UI の自動抽出

事務局オブジェクトの会議室結果メソッドを例にとり、抽出作業を手順を追って説明する。このメソッドは図 8 のような構造をしており、これは図 5 の **9b** と同型である。

メッセージ情報を分析すると、分岐している送信メッセージの情報 **a,b,g,e,k** と分岐基準情報 **f,d** から起動メッセージ情報 **f,g** を除いた **a,b,d,e,k** を新規情報として抽出できる。次に、このオブジェクトの過去の起動メッセージ情報を分析すると、会議開催者オブジェクトからの会議開催に関する起動メッセージ情報として **a,b,c,d,e** をすでに入手しているので、実際の新規情報は **k** だけとなる。これより、図 8 の下部に示すような **k** を入力するための UI を抽出することができる。このメソッドでは送信メッセージをルールで分岐しているので、図 8 の UI が利用されるのは**コメント**の入力が必要となったときのみである。

なお、UI の抽出段階では**日付**や**時限**などの項目名だけでなく、型名も同時に抽出するが、該当する型名が業務モデルに定義されていない場合はエンドユーザが直接定義する。型名は項目名に関連付けられており、一度どこかのオブジェクトの型名を定義、変更すると、その項目名を持っている全てのオブジェクトの型名が定義、変更される。

4.6 妥当性の検証

UI を抽出中に次のような警告が出された場合を例として妥当性の検証方法について説明する。

警告【WARNING】
警告種別：送信メッセージ情報の消失
- 消失情報：i,j
- 疑惑メソッド：事務局. 出欠通知

まず、疑惑メソッドである事務局オブジェクトの出欠通知メソッドを調べる。事務局オブジェクトの出欠通知メソッドは図 9 のようになっている。そこで、そのメッセージ情報を分析すると、警告文の消失情報 **i,j** は起動メッセージ情報であることが分かる。一方、送信メッセージ情報 **g** は過去の起動メッセージ **2** によって以前にこのメソッドに送信されているので、新規の情報では無いことが分かる。そのため、警告文にあったように情報 **i,j** はどの情報を生成するのにも利用されていないので、業務モデルに何らかの誤りがあると推定される。この場合は、送信情報は **g** ではなく **k（出席者リスト）**であった。

4.7 UI 自動生成

本システムでは策定した UI 構築プロセスを基に以下の手順で UI の自動生成を行なう。

1. UI 情報の抽出
2. 適切な UI オブジェクトの選択
3. UI 項目のグループ化と優先順位の定義
4. UI オブジェクトの配置
5. UI の提示とユーザによるカスタマイズ
6. フィードバックによる履歴情報の更新

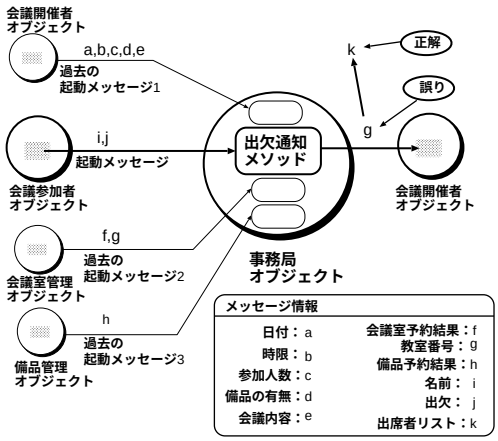

図 9: 事務局オブジェクト内の出欠通知メソッド

例えば、ある項目において項目名が**参加人数**で型名が**文字列型**という UI 情報を抽出した場合、まず型履歴情報を参照し、型に関して過去に同様の項目名と型名で UI オブジェクトを生成したことがあるか調べる。ある場合は型名と項目名の対から、無い場合は型名のみから出現頻度の高い順でポイント付けをした複数の UI オブジェクトを選択する。これを全 UI 項目に対して行ない組み合わせることで複数の UI の候補を生成する。

次は各 UI に対して同一オブジェクトで定義されている項目同士をグループ化した後に優先順位を業務モデルから定義する。例題では、**日付**と**時限**はともに会議室管理オブジェクトで定義されているので同一グループと見なすことができる。優先順位は各メッセージで最初に定義されたものを最重要項目と考え、高い優先度を与える。なお、グループ化と優先順位の定義に関してはポイント付けは行なわない。

そして、配置履歴から各 UI 項目に合った配置ルールを選択し、そのルールにそって配置する。配置ルールは 12 個のルールにより構成され、各ルールが取る値によって配置を表現する。一つの UI には 12 個のルール一組を適用する。ルールは Motif と OPEN LOOK のスタイルガイド [9,10] を参考にして作成した。配置ルールの選択基準は UI 項目との類似度と配置ルールの出現頻度である。

最後に全てのポイントを合計し、ポイントの高い順に提示する。ユーザはその中から最も理想に近いものを選び、必要ならばカスタマイズする。UI をカスタマイズするしないに関わらず、構築された UI をフィードバック情報と考え、UI オブジェクト履歴情報と配置履歴情報を更新し、今後の UI の自動生成に利用する。

表 1: 会議開催システムにおける UI の自動抽出結果

メソッド名	UI 項目抽出率	補足
予約	**100%**	
予約失敗通知	**100%**	
出席者通知	**100%**	余分に抽出
備品予約	**100%**	
開催通知	**100%**	
出欠通知	**100%**	余分に抽出

5 評価

5.1 UI 自動抽出精度の実験と評価

4 章で構築した会議開催システムを用いて UI 自動抽出精度を調査した。業務モデルは図 2 のようになっている。本例題にはメソッドが全部で 14 個存在する。そのうち UI が必要なメソッドは 6 つである。全メソッドに正しく UI を生成することができるのか、および、その UI が備えるべき UI 項目をどの程度の精度で抽出できるかについて調査した。

抽出結果は表 1 のようになった。抽出の結果、UI が必要な 6 つのメソッドを過不足なく全て認識でき、さらに各 UI の UI 項目についても 100%抽出できたことを確認した。これにより本システムの UI 自動抽出に関する有用性を確認できた。

2 つのメソッドについては本来抽出しなくても良い項目も抽出してしまったが、本来取得しなければならない項目は最低限取得しているので、特に大きな問題があるわけではない。例えば図 9 の場合、**a〜j** の UI 項目を全て出力項目としたが、新規情報 **k** を入力するのに本当に必要な情報は、**i** と **j** であった。本システムでは、不要な **a〜h** の出力項目は、カスタマイズ機能を用いユーザに削除してもらうことを想定している。

5.2 UI 自動生成精度の実験と評価

UI 自動生成技法の有効性を確認するために、学習量の異なるシステム間での UI 自動生成精度を比較した。比較はまったく学習していないシステムと 4 つの UI を学習したシステム、さらに 4 つ、合計 8 つの UI を学習したシステム間で行なった。そし

表 2: UI の自動生成精度の比較実験結果

	照会結果受理 UI	個人情報入力 UI
学習なし	11P(28P)	42P(50P)
UI を 4 つ学習	28P(45P)	50P(63P)
UI を 8 つ学習	100P	76P(84P)

て、これらの各システムで二つの UI を自動生成し、それぞれどの程度 UI を生成することができるかを比較する。生成対象の UI は、照会結果受理 UI と個人情報入力 UI である。照会結果受理 UI は、比較対象システムである 8 つの UI を学習したシステムのみが学習した経験のある UI である。一方、個人情報入力 UI はどのシステムも学習していない。実験の結果は表 2 のようになった。

評価は UI オブジェクトと UI 項目の配置がどの程度正解 UI と一致しているかを各々 50P（ポイント）満点で計算し、二つのポイントの合計（100P 満点）を比較することで行なった。UI オブジェクトの種類だけ正しい場合は 0.5 倍し結果は括弧内に示す。

表 2 より学習量を増やすことで UI 自動生成の精度が上がっていくことを確認した。また、一度も構築したことがない UI でも高い精度で UI を生成できることを確認した。これは項目名が履歴にない場合は型名全体の統計で UI オブジェクトを選択しているためである。つまり、本システムでは**型名全体の統計がエンドユーザの経験を表している**ため、初出の項目名に対してもエンドユーザの経験を参照することでエンドユーザが選ぶであろう UI オブジェクトを選択し、適切な UI を推論することができる。学習しない場合と、四つ学習した場合において個人情報入力 UI の方がポイントが高いのは良く利用され、履歴情報が集まりやすい TextField を多用しているからであると考える。

6 まとめ

本稿では、業務モデルを分析することで UI 情報を自動抽出し、抽出した情報とエンドユーザの UI 構築の履歴情報から最適な UI を自動生成する手法について述べた。

そして、例題を用いて UI 自動抽出精度と UI 自動生成精度に関する実験を行ない本システムの有効性を確認した。

参考文献

[1] 中所武司：**M-base**：「ドメインモデル ≡ 計算モデル」を志向したアプリケーションソフトウェア開発環境の基本概念、情報処理学会ソフトウェア工学研究会資料、No.95-SE-104-4 (May 1995).

[2] 中所武司, 小西裕治, 松本光由:メッセージフローに基づく分散協調型アプリケーション開発技法, 情報処理学会論文誌, Vol.40,No.5,pp.2387-2396 (May 1999).

[3] 石樽久嗣、紺田直幸、中所武司：メッセージフローモデルに基づくエンドユーザ主導型アプリケーション構築・検証技法、オブジェクト指向 2000 シンポジウム論文集、pp.133-140 (Sep. 2000).

[4] 紺田直幸、中所武司：「ドメインモデル ≡ 計算モデル」を志向したアプリケーション開発環境 **M-base** における開発・検証技法、ソフトウェア工学研究会、No.4-SE-125-3 (Jan. 2000).

[5] 寺岡照彦、京本寿美恵、押田秀治、秋吉 政徳：ユーザの操作履歴に基づく GUI 適応化の試み情報処理学会第 61 回全国大会講演論文誌第四分冊 (Oct. 2000).

[6] 白銀純子、深澤良彰：プロトタイプのビジュアル・カスタマイズによる GUI の自動生成手法情報処理学会論文誌、Vol.41,No.7,pp.1999-2009 (Jul. 2000).

[7] 木本陽介、服部進実：分散オブジェクト指向による作業融合支援プラットフォーム FusionWorks の開発、情報処理学会論文誌、Vol.37,No.12,pp.2352-2361 (Dec. 1996).

[8] 山岡俊樹、岡田明：ユーザインタフェースデザインの実践、海文堂 (Mar. 1999)

[9] Open Software Foundation、日立製作所ソフトウェア開発本部訳：OSF/Motif スタイル・ガイド リリース 1.2、トッパン (Jun. 1993)

[10] Sun Microsystems,Inc.: OPEN LOOK スタイルガイド、Graphical User Interface Application Style Guidelines、アジソン・ウェスレイ・パブリッシャーズ・ジャパン（Nov. 1990）