ソフトウェア工学 125－3
（2000．1．14）

「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

紺田直幸† 中所武司†

近年，パソコンの普及と共に，業務の専門家が自ら情報システムを構築する必要性が高まっている．そのためには業務モデルを構築することが，そのまま情報システムの構築につながるような技術が必要であるという観点から，「業務モデルと計算モデルの一致」，「分析，設計，プログラミングの一体化」という二つの基本コンセプトを設定した．我々は，このコンセプトを達成するためにオブジェクト指向概念に基づくアプリケーション開発環境 **M-base** の実現を目指している．アプリケーションの開発はコンポーネントベースのビジュアルツールを用いて行い，ユーザインターフェースを自動生成することで開発におけるエンドユーザの負担を減らし，また，自動生成されたユーザインターフェースの遷移図を用いることでアプリケーションの実行の流れをエンドユーザにも掴みやすくした．それにより，エンドユーザが主体となってアプリケーションを開発することを可能とした．本報告では，このアプリケーション開発環境に国際会議のプログラム委員長業務を例題として適用し，その有効性を示す．

Application development and test process using an application development environment M-base, on the concept of "a domain model ≡ a computation model"

NAOYUKI KONDA† and TAKESHI CHUSHO †

Explosive increase in enduser computing on distributed systems requires that endusers develop application software by themselves. One solution is given as a formula of "a domain model ≡ a computation model" and "analysys ≡ design ≡ programming". This formula implies that one task in a domain model of cooperative work corresponds to one object in a computation model based on an object-oriented model. Application development environment, **M-base**, supports this formula for cooperative systems such as groupware and work flow systems. Application is developed with visual tool based on components. Automatic User-interface generation. and transition diagram is also generated automatically. end-users can understand application-flow with this diagram. Therefore Enduser-initiative application development technique are archived.

1. はじめに

近年，ワークステーションやパソコンの普及およびそれらをつなぐネットワークの普及と共に，業務の専門家が自ら情報システムを構築する必要性が高まっている．このような新しい動向に対応して，コンポーネントウェア [1] やビジュアルプログラミング [2] に代表されるような新しい開発技法が普及しはじめている．

業務の専門家が自ら作り，自ら使うようなエンドユ－ザコンピュ－ティングの促進のためには，プログラミングの概念を排した新しいソフトウェアパラダイムが必要であるという認識から，基本的コンセプトとして，

- A domain model ≡ a computation model
  （業務モデルと計算モデルの一致）
- Analysis ≡ design ≡ programming
  （分析，設計，プログラミングの一体化）

を設定した．

我々は，"モデリング&シミュレーション" によってこれらのコンセプトを実現する分散オブジェクト指向設計技法を確立し，それに基づくアプリケーション開発環境 M-base[3)~7)] を開発中である．エンドユーザコンピューティングの分野を対象としているため，M-base による開発では，まず核になる部分を試作し，それを改良しながら実用システムに仕立て挙げるプロ

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

トタイピングアプローチをとる．

本報告では，まず2章で本研究の目的と対象を，3章でモデリングプロセスの概要，4章で開発環境の概要を述べ，以降は「国際会議のプログラム委員長の業務[8]」を例題として用い，5章でモデリング手順，6章でUIの自動生成法を，7章でUIによるドメインモデルの検証プロセスについて，8章でシミュレーションによる検証方法について述べる．

2. 研究の目的と対象

2.1 エンドユーザ

一般に，エンドユーザの範囲は広いが，本研究では，ユーザ企業のエンドユーザ部門に所属する基幹業務担当者，および一般にオフィスワーカーといわれるような人達を対象とする．特に，後者の人達は，業務の専門家として，DB検索や表計算などに市販のアプリケーションパッケージを利用しており，今後急速に増加すると思われる．

2.2 対象ソフトウエア

本研究では，「すべての日常的な業務をコンピュータ化する」，即ち，「日常的業務はマニュアル化でき，マニュアル化できればコンピュータ化できる」という観点から，オフィスでの業務用アプリケーションを中心にグループウェアやワークフローシステムなども対象とする．規模的には，中，小規模のアプリケーションソフトウエアを想定することになるが，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．

2.3 開発・保守形態

本研究では，基本的コンセプトとして，

「ドメインモデル≡計算モデル」

「分析≡設計≡プログラミング」

を掲げた．これは，問題領域を分析してドメインモデルを構築した時点でソフトウエアの開発を完了させようというものである．即ち，

「ソフト開発＝モデリング＋シミュレーション」

という図式で表現されるように，問題領域のモデルを作成し，そのモデル上でシミュレーションしてモデルの妥当性を検証した後，実用に際しては，そのモデルをインタプリタにより実行するか，あるいは必要に応じて実際のプログラムを自動生成するという方法である．

この時，特定分野向きのコンポーネントウェアを導入して，プログラミングの複雑さを回避することが重要である．

このように最終的にはエンドユーザが自らの業務のアプリケーションソフトウエアを自ら開発し，自ら利用することを目標とするが，その実現に向けての技術課題は多い．そこで，研究のマイルストーンとして，まずシステムエンジニアが基本となるコンポーネントを用意し，エンドユーザはそれを使用して開発を行うが，その後のシステムやコンポーネントの保守・拡張はエンドユーザだけで行うレベルを設定する．

3. モデリングプロセスの概要

3.1 2階層モデル

今後増加していくと思われるエンドユーザコンピューティングの分野に分散オブジェクト指向設計技法を適用するためには，モデリングの初期の段階で，従来の技法であまり明確に示されていないオブジェクト群の動的な振舞いを記述することが重要である．我々は，次に示すようなマクロモデルとミクロモデルの2階層モデルを導入して，基本的コンセプトを実現した．

(1) マクロモデルは，「ドメインモデル≡計算モデル」を実現するレベルであり，オブジェクト指向の分散協調型モデルとして位置づけられる．

(2) ミクロモデルは，「分析≡設計≡プログラミング」を実現するレベルであり，オブジェクト指向のクラス定義として位置づけられる．

3.2 ドメインモデルの構築手順

オブジェクト指向概念の発生学的定義に基づいてモデリングプロセスの形式化を行う[3]．概要を図1に示す．

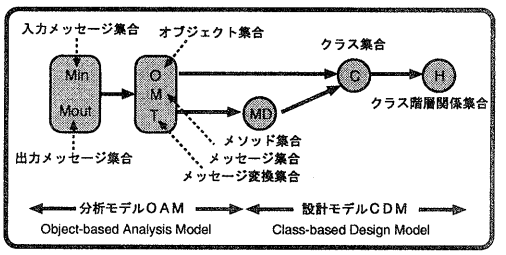

図1 M-baseのモデリングプロセス

分析フェーズで構築されるインスタンスベースのドメインモデルを以下のようにOAM (Object-based Analysis Model) として表現する．

OAM = { O, M, T }

O = o[i]

M = m[i,j,n]

T = t[r]

ドメインモデルOAMは，オブジェクトの集合O，メッ

セージの集合 M，メッセージ変換の集合 T の 3 つ組で規定する．o[i] は，ドメインモデルに含まれる i 番目のオブジェクトである．m[i,j,n] は，i 番目のオブジェクト o[i] から j 番目のオブジェクト o[j] へ送信される n 番目の種類のメッセージである．メッセージ変換の集合 T は，あるオブジェクト o[j] があるメッセージを受信した後，メッセージの列を送信するという関係を

$$t[r] : m[i,j,n] \rightarrow m[j,k_1,n_1], m[j,k_2,n_2], \ldots$$

と表現したメッセージ変換の集合である．また，ドメインモデル OAM の外界を仮想的にオブジェクトとみなして，外界からドメインモデルへのメッセージ集合 Min，ドメインモデルから外界へのメッセージ集合 Mout を決定する．

3.3 設計モデルの構築手順

ドメインモデルは，2 階層モデルの下位の層に対応する設計モデルに詳細化される．設計モデルはクラスに基づいて構築されるので，以下のように CDM (Class-based Design Model) として表現する．

$$CDM = \{MD, C, H\}$$

設計モデル CDM は，メソッドの集合 MD，クラスの集合 C，クラスの階層関係の集合 H の 3 つ組で規定する．

4. 開発環境の概要

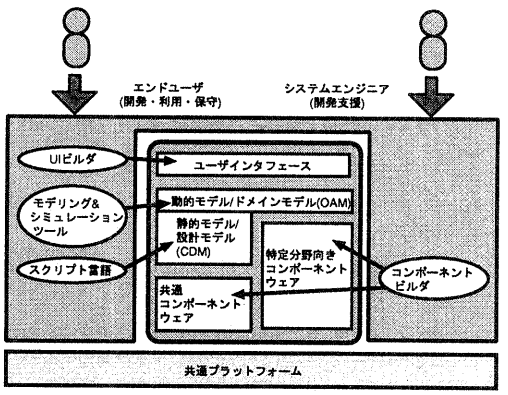

図 2 M-base の開発環境（外側）とアプリケーション・アーキテクチャ（内側）

M-base の開発環境とアプリケーション・アーキテクチャの関係を図 2 に示す．開発環境は主に次のような構成要素からなる．

- モデリング&シミュレーションツール
- スクリプト言語
- UI ビルダ
- コンポーネントビルダ
- 共通プラットフォーム

これらのツールを用いて開発されるアプリケーションは大まかには次の 3 つの部分から構成される．

- ユーザインタフェース
- モデル
- コンポーネントウェア

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る．動的モデル（ドメインモデル）に対応する OAM の部分をモデリング&シミュレーションツールを用いて作成する．静的モデル（設計モデル）に対応する CDM の部分をスクリプト言語を用いて作成する．

ユーザインタフェースはそのアプリケーションのユーザとの対話処理部分であり，モデルと独立させることにより，クライアント／サーバシステムなどのシステム構成，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．その構築ツールとして UI ビルダがある．

コンポーネントウェアには，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．後者を充実させることでエンドユーザによるアプリケーション構築が容易になる．このようなコンポーネント化を支援するためにコンポーネントビルダを用意する．共通プラットフォームはミドルウェアと基本ソフトウェアの部分で，オープンシステムや部品の流通の観点からは特に分散オブジェクト管理機能などが重要である．

5. モデリング手順

メッセージフローが本質的な意味を持つ分散型アプリケーションソフトウエアは，計算モデルもデータモデルも確立していないことがことが多い．そこで，オブジェクト間の静的な関係に先立ってメッセージの送受信によって規定されるオブジェクト間の動的な関係，すなわち，システムの振る舞いを定義する必要がある．

そのため M-base では，モデリングツールを使用することでシステムの動的な振る舞いをコンポーネントベースで定義することを可能とした．具体的には各業務処理を表すコンポーネントを矢印で結び付けていくことで簡単に業務プロセスを定義でき，修正も簡単に行えるような，ビジュアルツールを用いて定義を行う．

アプリケーションを作成する手順は以下の通りである．

(1) オブジェクトとメッセージの抽出
(2) ドメインモデルの定義
(3) 各コンポーネントの詳細を定義
(4) UI の作成
(5) シミュレーションによる妥当性の検証

5.1 国際会議のプログラム委員長の業務

ここからは,「国際会議のプログラム委員長の業務」を例題としてとりあげ，この業務に対し，M-baseのアプリケーション作成プロセスを適用しながら，説明していく．

この「国際会議のプログラム委員長の業務」は，ソフトウェア工学研究会要求工学ワーキンググループにおいて共通問題とされているもので，国際会議を開く際に，プログラム委員長が行なう業務である．この業務のうち，以下の部分を例題としてとりあげる．

(1) スケジュールの決定
(2) CFPの作成・配布
(3) プログラム委員の選出，登録
(4) 投稿されてくる論文に論文番号を割り当て，登録する．
(5) 論文の投稿者に投稿通知を出す．

この業務を自動化するアプリケーションの開発を想定する．

5.2 オブジェクトとメッセージの抽出

オフィスの日常的業務 (ルーチンワーク) は，メッセージ駆動型の分散協調型モデルを用いて次のように表現できる．

(1) ひとまとまりの日常的業務を割り当てることのできる，一人または複数の人からなるグループを一つのオブジェクトに対応させる．
(2) 一人またはグループの間で相互の通信手段として用いられている書類，メモ，電話，郵便，口頭連絡などはすべてメッセージとみなす．
(3) 日常的業務における協調作業はメッセージの送受信によって遂行される．

本研究では，一つの業務を擬人化したオブジェクトに割り当てることにより，メタファーベースのモデル化を行う．実世界と同じように一つのオブジェクトに複数の業務を割り当てることも可能であるが，柔軟性と保守性を重視して「1業務1オブジェクト」の割り当てを原則とした．

5.3 業務仕様の詳細化

この「1業務1オブジェクト」の原則をもとに，国際会議のプログラム委員長業務を詳細化し，オブジェクトとメッセージの抽出を行う．

- スケジュール
  スケジュールを決める．決めるべきスケジュールは以下の五つ．
  - 論文投稿締切日
  - プログラム委員会開催日
  - 著者への論文の採否通知日
  - Camera Ready 締切日
  - 印刷業者への発送日

  ただし，学会開催日を入力することで，過去の同様の会議のデータを基にこれらのスケジュールの原案が自動的に作成される．
- スケジュール表
  スケジュールを記録するためのスケジュール表．
- CFP 作成
  学会のテーマ，概要などを入力し，テンプレートを用いて Call for Paper を作成する．Call for Paper は HTML で作成され，配布形態に応じて，テキスト版，PDF 版が HTML 版から自動作成される．
- CFP 配布
  学会誌などへ Call for Paper の掲載依頼をしたり，配布を行う．
- 委員選出
  プログラム委員を選び，就任依頼を行う．また，プログラム委員用のメーリングリストを作成する．
- 委員名簿管理
  プログラム委員名簿を作成・管理する．プログラム委員の名簿データは，名前，住所，所属，電話番号，FAX 番号，E-mail アドレス，得意とする分野からなる．
- 論文受付
  投稿された論文に論文番号を割り当てて登録する．論文は印刷されたものが郵便で送られてくることになっており，その表紙には 題目，著者，連絡者の氏名，住所，所属，電話番号，E-mail アドレス，FAX 番号，Abstract，キーワードが書かれている．
- アブストラクト受付
  論文に先だって投稿される，アブストラクトや，題目，著者名などの項目を受け付ける．アブストラクトは電子メールで受け付けることとし，論文を受け付けた際の手作業入力の手間を省く．アブストラクトを受け付けた際，自動的に受領通知が返信される．
- 投稿論文管理
  先行して投稿されてくるアブストラクトと，論文番号を割り当てられた論文を管理する．
- メイル
  CFP 配布や，プログラム委員就任依頼，論文の受領通知の送信を行う．

5.4 ドメインモデルの定義

抽出したオブジェクトとメッセージをドメインモデ

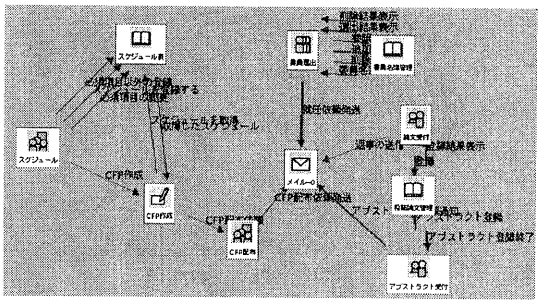

図 3　ドメインモデル定義

表 1　オブジェクトの定義例

オブジェクトの名前	CFP 作成
入力メッセージ（メソッド名）	CFP 作成，スケジュールの取得
出力メッセージ（イベント名）	スケジュール取得要求，CFP 配布依頼

ルにマッピングする．具体的にはモデリングツールを使い，オブジェクトをコンポーネントとして定義し，そのコンポーネント間に矢印を引くことでメッセージを定義する．

国際会議のプログラム委員長の業務に関するドメインモデルの定義は図 3 のようになる．

このコンポーネントは，ユーザが定義する方法とあらかじめ用意されているコンポーネントを使用する方法の 2 種類がある．

ユーザが定義する場合，業務の専門家はウィザードにしたがって以下のような入力を行うことによりコンポーネントの定義を簡単に行うことができる．

(1)　オブジェクトの名前の定義
(2)　入力メッセージの定義
(3)　出力メッセージの定義

CFP 作成オブジェクトのコンポーネント定義は表 1 のようになる．

一方，M-base であらかじめ用意されているものを特定業務向けコンポーネントという．これはエンドユーザの負担を減らすために特定業務領域に限定してあらかじめ定義されているものである．

ユーザ定義で業務の専門家が定義したコンポーネントを特定業務向けコンポーネントとして登録することも可能とした．これにより今後作成するアプリケーションでそのコンポーネントの再利用が可能になる．

5.5　モデリングツール

モデリングツールは，エンドユーザが以上の作業をコンポーネントベースでビジュアルに定義するツールである．各業務処理を表すアイコンを矢印で結び付けていくことにより，簡単に業務プロセスを定義でき，修正も簡単に行える．また，コンポーネント定義は再帰的な定義が可能となっており，コンポーネントのメソッド内に新規にコンポーネントを置くことができる．

業務プロセスを定義中のモデリングツールの画面を，図 4 に示す．

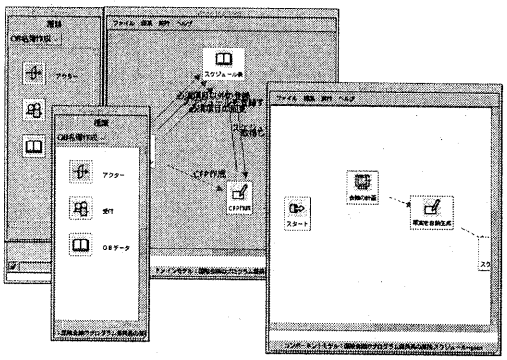

図 4　モデリングツールを用いたモデリング

奥のグレーの画面でドメインモデルの定義を行い，手前の白の画面でユーザ定義コンポーネントのメソッド定義を行っている．

6.　UI の定義

M-base による開発では，エンドユーザの負担を減らすために，ユーザインターフェースはシステムによって自動生成される．エンドユーザは必要に応じてこの UI をカスタマイズすることで，使い勝手の向上を図ることが可能である．

6.1　UI の作成手順

ユーザインターフェースは，以下の手順で作成される．

(1)　モデリングツール上での情報の定義
(2)　UI の自動生成
(3)　生成された UI のカスタマイズ

従来型のアプリケーション開発技法においても，ユーザインターフェースの作成はアプリケーション開発においてかなりの時間を占める [2]．それに加え，M-base のようなエンドユーザ主導型開発では，要求仕様やドメインモデルの変更が頻繁に起こることが予想される．そのため，UI の変更も頻繁に起こり，その度に UI を作成するのは，エンドユーザにとっては多大な負担になる．そこで，UI は自動生成する形を取るのだが，自動生成で作成された UI がアプリケーションを利用する立場のエンドユーザにとって使いやすいユーザイン

ターフェースであるとは限らない．それを解決するために，M-base による開発では，情報の定義，UI の自動生成，カスタマイズという三段階をとる．アプリケーション開発中は，UI 作成に手間がかからないようにし，完成後は利用者が使いやすいように，カスタマイズを行う．

6.1.1 (1) モデリングツールでの情報の定義

モデリングツールで UI 作成に必要な情報をウィザードに従い入力する．UI 作成に必要な情報は，名前と，入力するデータ型からなり，また，以下の二つのどちらであるかを指定する．

入力情報 利用者が画面から入力する情報の名前と説明を定義する．

出力情報 最初の画面表示時に出力する情報と，何らかのアクションを起こした後に結果として帰ってくる出力情報を定義する．

6.1.2 (2)UI の自動生成

モデリングツールで定義した情報に基づき，UI が自動生成される．この自動生成のためのツールを UI ビルダと呼ぶ．ここで自動生成された UI は，その時点でモデリングツールにより作成されるアプリケーションに接続可能である．

モデリングツールで入力された「データ型」は，モデリングの際の情報として用いられるのだが，UI 作成の際にも利用する．UI を作成するとき，日付型なら日付入力に適したカレンダウィジェットを，Boolean 型ならばチェックボックスをデフォルトで配置することで，使いやすい UI を自動で作成できるようにした．

また，データ型だけではなく，後述のカスタマイズ作業で行われるウィジェット交換作業の際に，ラベル名とそれに割り当てられたウィジェットの組合わせを保存しておき，その対応付けを UI の生成の際に利用することで，同じデータ型を入力するためのウィジェット群の中からより最適なものを選べるようにした．

このように，特定のデータ型，アイテムを入力するためのウィジェットを自動で適切に選ぶことで，自動生成された段階でデバッグを行う程度には充分使える UI を生成することが可能とした．

例として，スケジュールを決定する際に使用するユーザインターフェースを，図 5 に示す．モデリングツール上で，入力情報として，学会開始日，論文投稿締切日，プログラム委員会開催日，著者への論文の採否通知日，Camera Ready 締切日，印刷業者への発送日をそれぞれ与え，そのデータ型は日付型である．

データ型として日付型が設定されているため，入力用ウィジェットとしてカレンダが選択されている．

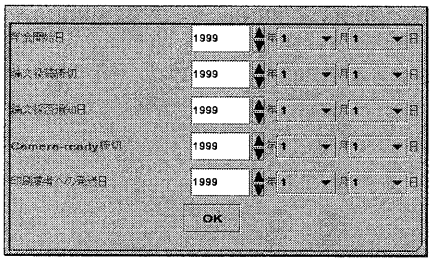

図 5 自動生成された UI

6.1.3 (3)UI のカスタマイズ

UI のカスタマイズは，次の二つを行う．このカスタマイズのためのツールを，UI カスタマイザという．

- 自動配置されたウィジェットの修正・変更
- 新たな GUI コンポーネントの追加
- 部品間の関連付けによるウィジェットの協調動作

修正とは，自動配置されたウィジェットの色や，大きさ，位置，デフォルトで表示されているテキストの修正のことを指す．部品自体を交換することもできる．また，部品間を関連づけることにより，「テキストを入力しないとボタンが押せない」といった動作を UI に組み込むことが可能である．

図 5 の UI をカスタマイズした例を図 6 に示す．学会開始日からスケジュールの原案を決定するので，学会開始日はカレンダで日付を選択しやすくし，その他の日付は原案が設定されたあと微調整を行いやすいような上下のスピンコントロールが設定されている．

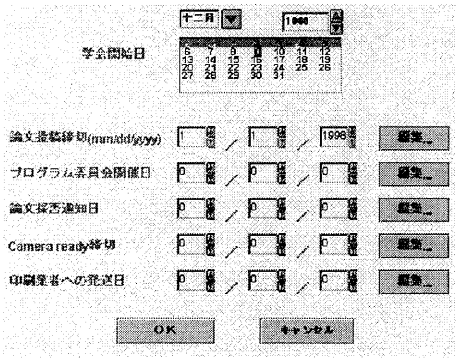

図 6 カスタマイズされた UI

6.1.4 新規ウィジェットの作成

既存のウィジェットの中に，自分の求めているものがあれば良いのだが，常に自分の考えているものと同じウィジェットがあるとは限らない．そこで，M-base による開発では，部品を新規に作成して UI の使いや

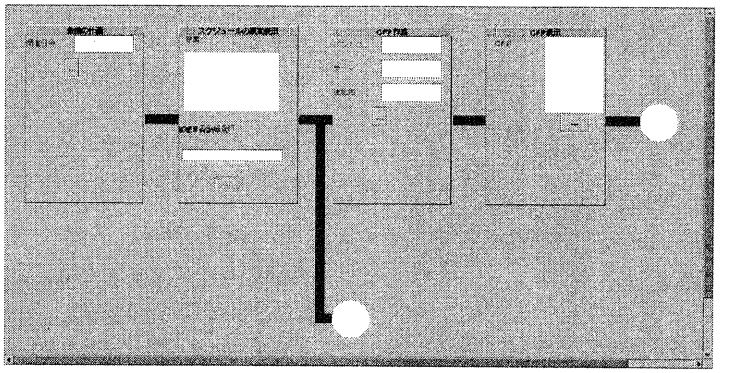

図 8　遷移図

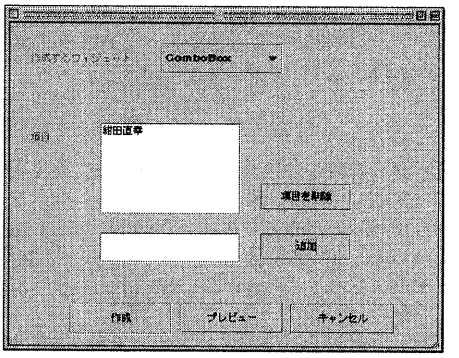

図 7　部品郡制作部

すさの向上を図ることができるようにした．

部品制作中の画面例を，図 7 に示す．この画面では，ComboBox を作成するために，ComboBox の項目を設定している．追加ボタンを押すことで項目が追加され，ComboBox が作成される．

7. UI の遷移図によるドメインモデルの検証

一般にアプリケーション開発の早期の段階で，ユーザインタフェースの詳細や，画面遷移などを実装し，アプリケーションを利用するエンドユーザに実際に使わせることがある．これは，エンドユーザに画面や画面の遷移を見せることで，画面操作性の確認や，エンドユーザの漠然とした要求を早期に具体化することを目的としている．また，画面単体としては，必要な情報が表示されているか，データが正しいかどうかを確認するために使われる．

これは，アプリケーション開発技法である M-base においても同様である．さらに，M-base のようなエンドユーザ主導型の開発では，エンドユーザ自らが開発を行うことを基本としているため，思いつくところから順に実装を行ってしまい，未実装部分を多く残す可能性がある．シミュレーションを行うことでその検証も可能ではあるが，全体が完成していなくても，遷移図を生成することができるので，部分的な検証を行うことが可能である．さらに，アプリケーションのクライアントの立場で操作手順を確認することで，アプリケーションのユーザの使い勝手の検証といったような，視点を変えた検証をすることもできる．

7.1 検証方法

国際会議のプログラム委員長の業務において，スケジュールの決定から CFP 配布までの実際の業務画面の流れは，

(1)　スケジュールの決定，
(2)　スケジュールのプレビュー，
(3)　CFP の作成 (必要項目の入力)，
(4)　CFP のプレビュー，
(5)　配布先の決定

というようになる．これを遷移図として表示させると，図 8 のようになる．作成の段階で，スケジュールのプレビュー後，CFP の作成に移るかどうかは任意に選択できるようにしてあるので，二分岐して，片方が CFP 作成へと続き，もう片方は終了となっている．

次に，図 9 に，遷移図の生成に失敗した例を示す．生成された遷移図が期待通りでなかったときは，モデリングの情報に誤りを含んでいることになる．その誤りにはモデリングツール上でのコンポーネント間の結線の脱落や間違い，UI の生成の失敗といったような，何らかの誤りがあったということである．

図 9 の例で言うと，スケジュールの結果表示から，CFP の作成画面へと移るはずのところで，終了してしまっている．

エンドユーザは，この自分の考えと違う遷移図が作成されたことで，何らかの誤りの存在を知ることがで

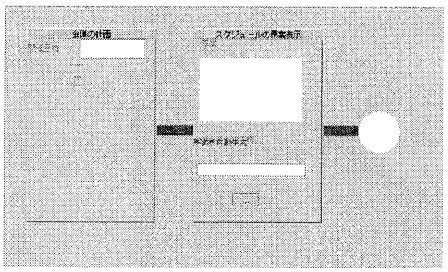

図 9　失敗した例

きる．終了地点がドメインモデル上でどの場所であるか，ツールにより特定することができるので，エンドユーザはそれに基づいてモデリングツール上で誤り箇所の発見・修正が可能である．

この例では，「スケジュールの原案表示」画面の後，終了しているので，結果表示の後から，次の画面である CFP 作成画面までの間に誤りがあることになる．そこでまずドメインモデルから調べ，ドメインモデルに欠落が見られない場合は，ドメインモデル上のコンポーネントや，コンポーネントビルダによってエンドユーザに定義されたコンポーネントであれば，メソッドの詳細定義まで調べ，誤りを発見する (図 10)．

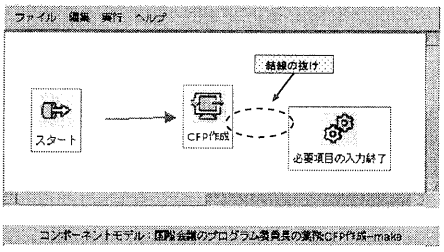

図 10　結線の脱落

8. シミュレーション

シミュレーションツールはモデリングツールにより作成した業務の流れの妥当性をビジュアルにチェックするためのものである．

この際シミュレーションを行う方法としては以下の 2 種類がある．

- オブジェクト間のメッセージパッシング表示
- イベントトレース

前者はオブジェクト間で実際にメッセージが流れている部分がドメインモデル図上で矢印となって表示され，現在行われている処理がわかる．この方法では，作成したモデル上での動作の確認を行うことができるので直感的に認識がしやすい．しかし，これでは現時点でのメッセージの流れしかわからないため，過去の事象や原因となるものが分かりにくいという点がある．

一方後者では，一つのオブジェクトに対して一つの区画が与えられすべてのイベントが事象トレース上に表されるので全体のシーケンスを見渡しやすい．

9. おわりに

本稿では，エンドユーザ主導型の開発技法と，それにおけるユーザインタフェースの自動生成方法，またユーザインターフェースを用いた要求仕様洗練，開発支援法を述べ，それを例題に適用することで，開発技法の有効性を示した．

今後は，さらに使い勝手のよい UI の自動生成法や，カスタマイズ時に用いるさまざまな新規ウィジェットの開発法について研究・考察を続けていく．

参考文献

1) 青山幹雄，中所武司，向山博：コンポーネントウェア，共立出版．(1998)
2) Daniel Tkach, Walter Fang, Andrew So: Visual Modeling Technique:Object Technology Using Visual Programming, Addison-Wesley(1996).
3) 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(1995).
4) 松本光由，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 におけるモデリング&シミュレーション技法，情報処理学会，ウインターワークショップ イン 恵那 論文集 (1998.1).
5) 岩田智彰，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 におけるコンポーネントベース開発技法，情報処理学会 第 120 回ソフトウェア工学研究会 (1998.6)
6) 紺田直幸，岩田智彰，中所武司: エンドユーザ主導型アプリケーション開発技法における要求仕様定義プロセス，情報処理学会ソフトウェア工学研究会要求工学 WG ウィンターワークショップ・イン・高知 論文集 (1999.1).
7) 紺田直幸，石榑久嗣，中所武司: エンドユーザ主導型アプリケーション開発技法におけるユーザインターフェースを用いた要求仕様の検証，情報処理学会ソフトウェア工学研究会要求工学 WG サマーワークショップ・イン・小樽 論文集 (1999.9).
8) 要求工学 WG 共通問題
http://www.selab.cs.ritsumei.ac.jp/~ohnishi/RE/problem.html