コンポーネントによるフォームフローシステムの部品化とボトムアップなシステム構築

中川光紀[†] 田中譲[††]

70年代に提唱されたフォームフローモデルは，オブジェクト指向の考えやInternet技術の導入によりIntranetやワークフローへと拡張されてきた．しかし，従来のシステムでは，まだトランザクションシステムとの統合やエンドユーザレベルにおけるフロー定義の支援が不十分である．この解決として本研究ではフォームフローシステムをコンポーネントの組合せで構築するためのフレームワークを提案する．さらにこの上でシステムを統合的に扱うための要素部品を定義する．プラント制御系やデータベースなど外部オブジェクトのProxy機能を持つコンポーネントの導入が外部オブジェクトとシステムとの容易な連係を可能にする．さらにこの機能をフォームに持たせることで，トランザクションシステムとフォームフローシステムの統合が可能となる．既存の稼動部品に対するProxyの作成でフロー定義のボトムアップな構築が可能となる．本論文では，このような拡張されたフォームフローシステムのフレームワークをIntelligentPadのアーキテクチャを基盤に用いて定義している．

A Component-based Framework for Form Flow System and Their Bottom-up Integration

Mitsunori Nakagawa[†] and Yuzuru Tanaka[††]

Form flow models proposed in 70's are recently extended to intranet systems and workflow systems by the introduction of object-oriented architectures and the Internet technologies. However, those extended systems cannot integrate transaction-based systems with themselves, nor support end-users to define form flow systems. This paper proposes an application framework for composing form flow systems as combinations of components, and clarifies what kind of components are required. Special components that work as a proxy of an external system such as a plant control system or a database will be introduced to easily integrate external systems with other components. We will extend the definition of a form using such a special component so that it works as a proxy to an external system. Such forms integrate transaction-based systems with their form flow system. Some type of proxy components allows us to integrate several existing form flow systems, in a bottom-up way, into a single large system. This paper clarifies such an extended form flow framework based on a media component architecture IntelligentPad.

1. はじめに

近年，企業内の情報を計算機内部で統合して扱うことを目的とした企業内情報処理システムのモデルが研究されている．この考えの1つとして，かつて1970年代にフォームフローモデルという考えが提唱された．Office by Example [1]は，リレーショナルデータベースの問合せ言語の1つであるQBEを拡張してデータベースに貯えられたデータをワードプロセッサや電子メールなどでの作成文書に挿入するといった処理を行うことができる．FORMANAGER [2]は，同様にデータベース内のデータを基にフォームを作成することを支援するシステムである．しかし，当時はデータや書類，それらの処理もまとめてオブジェクトとして統一的に扱うオブジェクト指向モデリングの考えもまだ出てきていなかったために，フォームフローモデルも十分に発展することはなかった．

90年代に入ってからは，オブジェクト指向の考えやInternet技術などが広く普及してきた．このことにより，これらを利用したIntranetやワークフローといった考えが企業内情報処理システムに登場するようになってきた．GUI技術とInternet/Intranet技術が著しく進んだ現在，フォームフローモデルには，すでに提案されているような基本機能に加え，これらの技

† K-Plex Inc.
K-Plex Inc.
†† 北海道大学知識メディアラボラトリー
Meme Media Laboratory, Hokkaido University

術を含んだ機能を追加拡張し，統合的に扱えるフォームフローモデルが必要と考える．

本研究の目的は従来のフォームフローモデル（以後，基本フォームフローモデルと記述）に下に述べるような機能を追加拡張した拡張フォームフローモデルを提案し，さらに可視的コンポーネントの合成を用いて，このモデルを実現するジェネリックなフレームワークを定義し，提案するものである．このフレームワークの実現可能性を示すために，1つの例として，実行時にもユーザの直接操作により動的にコンポーネントの組合せを変更可能である部品化メディア・アーキテクチャを基盤に用いて具体的にシステム構築を行い，提案するモデルとフレームワークの実現可能性を示した．

本研究と同様に従来のフォームフローモデルに機能を追加してフォームフローモデルを構築し直すことを試みる研究の例として，InConcert[3]，WorkFlo[4),5)]，$TriGS_{flow}$[6]，Exotica[7)]などがある．

以下に示すのは，本研究において必要と考える追加拡張機能と上にあげた研究との比較である．

A. GUIを用いて容易に自由なフロー定義が可能
B. Drag-and-Dropによる簡易にフロー定義が可能
C. トランザクションシステムとの統合
D. 定型処理に割り込んで再利用可能
E. システムをボトムアップに構築可能
F. 外部オブジェクトとの連係が容易
G. ツールをネットワークから配布，利用可能

このうち，AとBはユーザの操作性に基づく機能であり，システム利用者もGUIやDrag-and-Dropの機能を用いることで直接操作によって容易にフロー定義できることを示す．C〜Gはシステム統合面で重要な機能である．Cはトランザクションシステムをフォームフローシステムに組み込み，フォームへの処理形式でトランザクションシステムにアクセスを行い，参照・制御する機能である．Dは稼働中でもフロー定義を変更できるかを示す．Eは既存のフォームフローシステムを部品として統合し，より大きなシステムをボトムアップに構築可能か否かを示す．Fはプラント監視制御システムやデータベースなど外部オブジェクトをシステムで容易に利用可能か否かを示す．連係する外部オブジェクトの種類は前もって限定できないため，任意の外部オブジェクトと容易に連係できる柔軟性が必要となる．この機能はCの機能の実現に必要となる機能である．GはWWWなどネットワークを介してツールやフォームを提供する機能である．C〜Gの機能を提供することにより，将来，新たに必要となる機能が出てきた場合にもこれらの機能を用いることで，その新たな機能を組み込むことは可能である．これらの機能に関して従来のシステムの状況を**表1**に示す．

表1 他の研究との比較
Table 1 The comparison with related systems.

	A	B	C	D	E	F	G
Office by Example	△	×	×	×	×	△	×
InConcert	○	△	×	○	○	○	○
WorkFlo	○	○	×	?	○	?	×
$TriGS_{flow}$	×	×	×	○	×	△	×
Exotica	○	○	△	○	○	△	○
本研究	○	○	○	○	○	○	○

○：機能あり ×：なし △：一部機能あり ？：不明

これらの機能をすべて統合的に扱えるものはまだ存在していない．WorkFloは，フローの定義に関する機能の部分は用意されている．$TriGS_{flow}$ は定型処理への割込みと，外部オブジェクトとの連係の機能を持つがそれ以外は用意されていない．InConcertは，トランザクションシステムとの統合はなされていない．最近提案されたワークフロー管理システムの1つであるExoticaでは，GUIによるフロー定義は提供されていないが，その技術を用いた製品であるFlowMarkでは用意されている．また，ある特定のトランザクションシステムに結合する機構はあるが，任意のトランザクションシステムと結合するための汎用的な機構は用意されていない．

本研究ではこれらのA〜Gの機能仕様を統合的に満たす拡張フォームフローモデルを提案する．さらにその上でこれらの機能すべてを実現するために必要な要素部品のフレームワークを著者らが提唱している部品化メディア・アーキテクチャに基づいて提案する．

本論文では2章においてフォームフローの基本構成要素について述べる．3章で部品化メディア・アーキテクチャの概要について，4章以降は，各要素の構成について述べる．

2. フォームフローの基本構成要素

2.1 基本フォームフローモデル

基本フォームフローモデルを以下のように定義する．

(a) フォーム：フォームは固定した書式を持つ書類である．セル内部にはさまざまなデータを表現する．フォームの作成は，レイアウトを定義したひな型の各セルにデータを記入して作成する．

(b) フォームの変換：フォームフローモデルでは種々のフォーム変換モジュールを連結し，この経路に沿ってフォームを流す．これによって次々と各種変換を経て，最後に出力フォームへと変換される．このプロセスをフォーム変換モジュールをノードとするフローグ

ラフを用いてモデル化，表現する．このグラフには変換モジュールの出力を次の変換モジュールの入力に結合したフォームの流れを示すリンクと，フォーム変換の開始のタイミングを制御するトリガ信号を伝えるリンクとの 2 種類のリンクが記述される．

(c) トリガ：トリガはフローに対し制御を行う信号である．トリガは変換終了時といった業務状態や，アラームクロックのような時間状態など種々のオブジェクトから発せられる．トリガを受ける側も受けることで各動作の処理を開始する機構が必要である．

2.2 フォームフローモデルのコンポーネント化

本研究では前節で述べたフォームフローモデルを構成するフォーム，トリガ，フォーム変換の各要素をオブジェクト指向に基づき定義された可視的コンポーネントによって定義，構築する．これらのコンポーネント部品を組み合わせてフォームフローシステムを構築する．この際，フォームフローモデルを構築する基盤技術として，(1) 可視的コンポーネントの組合せによるシステム構築，(2) 実行時に動的にコンポーネントの移動，結合，機能合成などが可能である，といった機能が必要となる．このような基盤技術の上に各要素をコンポーネントとして構築し，組み合わせてフォームフローシステムを構築することで 1 章で述べた追加拡張機能のうち A の『GUI を用いて容易に自由なフロー定義が可能』，B の『Drag-and-Drop による簡易にフロー定義が可能』，D の『定型処理に割り込んで再利用可能』の 3 つの機能を拡張できる．上に述べた基盤技術のシステムの 1 つとして，3 章で述べる部品化メディア・アーキテクチャがある．4 章ではこの部品化メディア・アーキテクチャを基盤に用いた，フォームフローモデルのコンポーネント化のフレームワークを示す．

2.3 拡張フォームフローモデル

本研究では，2.1 節で述べた基本フォームフローモデルを 2.2 節で述べた可視的コンポーネントによって構築したものに対し，さらに 1 章で述べた残りの機能を追加拡張した拡張フォームフローモデルを提案する．

1 章で述べた F の『外部オブジェクトとの連係が容易』と，C の『トランザクションシステムとの統合』の機能拡張のためにバーチャル・フォームを提案する．従来のフォームフローシステムでは，外部オブジェクトとの間に個々のインタフェースを用いることによってシステムとの統合を行っていた．今回提案するバーチャル・フォームはオフィスワークの部分だけでなく，プラント制御系やトランザクション処理なども含めた企業内でのあらゆる処理をフォームフローシステム内のフォームへの処理という統一した形式で統合的に管理可能となる．

1 章の E で述べた『既存の部品やシステムをボトムアップに構築可能』の拡張のためにリモート・リファレンスを提案する．従来のフォームフローシステムでは，専用のツールを用いて表示されているアイコンを結ぶ操作でフロー定義を行う．このアイコンと実際の部品との間に明確な関連付けはない．変換モジュールやフォームの 1 つに関連するリモート・リファレンスは，それ自身が 1 つのコンポーネントであるため，変換モジュールやフォーム，フロー定義などのコンポーネント部品と機能連係を行い，これらの部品との間で境界なく統合的に扱うことができる．リモート・リファレンスを用いて統合したシステムにも，さらにリモート・リファレンスを適用して，より大きなシステムをボトムアップに構築できる．

1 章の G で述べた『ツールをネットワークから配布，利用可能』のために任意のコンポーネントをネットワークを介して配布でき，他のコンピュータ上で取得，利用できる機能を用いる．これによりフォームだけでなく，提供された変換モジュールやフロー定義，そのリモート・リファレンスなどを取得，自分の環境に合わせ再編集して用いる．再編集したコンポーネントも再びネットワークへ提供することができる．

2.3.1 バーチャル・フォーム

フォームフローシステムでは複数のフォームが同一データを参照することがある．このときあるフォームのデータ更新を，他のフォームとの間で整合性を保つ必要がある．この解決にデータをデータベースを用いて一括管理するフォームを考える．このフォームはセルのデータがデータベースの 1 レコードの各属性値と連係する．データ参照，更新の際に，データベースにアクセスを行い，データの整合性を保つ．このフォームには検索と更新の 2 種類の機能がある．検索機能は特定のセルへの入力を条件にデータベースにアクセスし，残りのセルを自動的に埋める．結果が複数ある場合はフォーム自身を複製する．更新機能はフォームへの入力データに従い，データベースのレコードを更新する．

フォームの参照対象をプラント制御系や電子メールなど種々の外部オブジェクトとすることで，検索機能はこれら外部オブジェクトの状態監視に，更新機能は外部オブジェクトの制御になる．このように外部オブジェクトを参照して，動的に内容が変わるフォームをバーチャル・フォーム，対して通常のフォームをリアル・フォームと呼ぶ．フォームという形の可視的コン

ポーネントによって Proxy を表現することで，他の可視的コンポーネントとの機能合成を直接操作によって行うことが可能となる．これによってシステムの利用者でも容易に他システムとの連携，組替えを行うことが可能となる．システムの利用者はこの 2 つのフォームの区別を意識せずに利用できる．

2.3.2 リモート・リファレンス

リモート・リファレンスはネットワーク上に分散して存在するツールやシステム，フォームの 1 つへのポインタを内部に保持することでそれらと関連を持つ可視コンポーネントである．リモート・リファレンス自体を 1 つの可視的コンポーネントとして実現することで対応するコンポーネントを参照するのみの機能だけでなく，対応するコンポーネントの持つ情報を参照したり，提供している機能の一部を利用したり，対応コンポーネントとの間でイベント配信を行ったりすることができる．そしてリモート・リファレンスと機能合成した他の可視的コンポーネントからこれらの機能を利用することができ，システム利用者にも容易に分散して存在するシステムを結合することが可能となる．

フロー定義で，異なる変換モジュールに関連するリモート・リファレンス間へのリンクの定義が，変換モジュールどうしのリンクになる．

システム利用者は他から提供されているリモート・リファレンスを入手することで，それを介して，関係付けられているツールやフォームの保持している情報を参照したり，提供されている機能を呼び出して利用，制御が可能である．リモート・リファレンスによって参照するサブフォームフローの詳細を隠蔽し，1 つのコンポーネントと見なすことを可能とする．リモート・リファレンスを適用できる対象はコンポーネント，ツール，サブフォームフローなど任意の階層のものを対象とすることができる．リモート・リファレンスは既存のシステムに対して従来の機能に影響を与えることはない．これによって既存の複数のシステムをサブシステムと見なしたより複雑なシステム提案するボトムアップ構築に適用することができる．

2.3.3 流　　通

2.2 節で述べた機能を持つ基盤技術を用いることで，他のコンピュータから取得した可視的コンポーネントを再編集，再利用することができる．今回利用した部品化メディア・アーキテクチャには，コンポーネント部品による WebBrowser があり，HTML 内に埋め込まれた可視的コンポーネントを取得，利用できる．

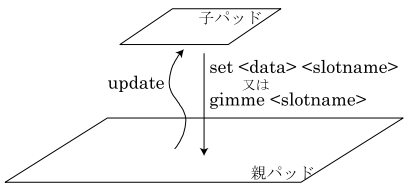


図 1　パッド間のデータ授受
Fig. 1　The message flow between two pads.

3. 部品化メディア・アーキテクチャ

本研究では，2 章で述べた各要素を部品として構築し，これを組み合わせてシステムを構築する．構築の基盤技術には，部品化メディア・アーキテクチャとして利用可能な IntelligentPad システム[8]を用いた．IntelligentPad システムでは，計算機上のオブジェクトすべてをパッド（Pad）として統一した表現形態で扱う．各パッドはスロットと呼ばれる連想配列を持ち，貼り合わせたパッドどうしでスロットを結合することで機能合成を行う．IntelligentPad はパッド間のデータ授受の形式を統一することで任意のパッドどうしを結合可能とする．

図 1 に貼り合わせたパッド間でのメッセージとデータの流れを示す．子パッドの状態更新によって，主スロットのデータを引数に set メッセージを親パッドに送る．逆に親パッド側の状態更新では，まず親パッドから子パッドへ update メッセージが送られ，これを受けた子パッドは gimme メッセージにより親パッドからスロットのデータを受け取る．こうして貼り合わせたパッド間でデータ授受が行われる．このほかに ‘paste’，‘resize’ といったパッドへの操作イベントも geometrical message として子パッドから親パッドに送られる．

IntelligentPad に類似のものとして JavaBeans 対応の RAD ツールがある．これは Beans コンポーネントの組合せでシステムを構築する．構築では Builder を用いて，Beans の配置と，各コンポーネントどうしの機能連係の 2 段階の構築を要する．このため JavaBeans は見栄えと機能が異なる構造も可能である．また構築と実行の環境が区別されており，既存のシステムから部品を再利用といった操作をダイナミックには行えない．一方今回利用した IntelligentPad システムは，各コンポーネント間のメソッドが統一，貼り合わせによる親子関係の間にのみ機能連係が存在する見栄えと機能の統一した構造関係によって，コンポーネント間のジェネリックな連係を直接操作によって定義することが可能である．IntelligentPad では構築と実行

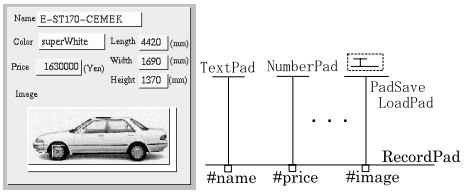


図 2 フォームの例とその構造
Fig. 2 A display hardcopy of a form and its composition structure.

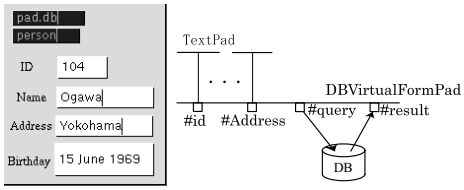


図 3 バーチャル・フォームの例とその構造
Fig. 3 A display hardcopy of a virtual form and its mechanism.

環境の区別がなく，ダイナミックに部品を再利用することが可能である．

IntelligentPad システムを基盤として用いて 2.2 節で述べたコンポーネント化を行いフォームフローの基本部品をパッドによって構築することで，IntelligentPad の利点である動的に可視的コンポーネントを機能合成して可能なシステムを容易に実現することが可能となる．さらに今回提案する 2.3 節の拡張フォームフローの要素部品を構築して拡張フォームフローのフレームワークを示す．

4. 拡張フォームフローのフレームワーク

本章では，各要素部品のフレームワークを述べるために 3 章で述べた部品化メディア・アーキテクチャに基づいて説明する．

4.1 フォームの定義

図 2 にパッドで構成したフォームの例を示す．フォームの台紙は RecordPad である．RecordPad はスロットを自由に追加削除できるパッドである．例では，#name や #image などのスロットが追加してあり，この各スロットに，各セルを表現するパッドを結合してフォームを定義する．

3 章で述べた部品化メディア・アーキテクチャではさまざまなデータ型を表示，利用可能とするプリミティブパッドが存在する．パッド自身も 1 つの表現タイプになる．これはアプリケーション・プログラムの起動ボタンを含ませたり，複雑な形式の表や，内部に別のフォームを埋め込んだりといった，パッドの貼り合わせで表現可能なデータをセルに埋め込み，扱うことができる．図 2 の属性 Image のセルがイメージを直接表示しているのでなくセルに埋め込まれたイメージ表示パッドによって表示を行っている例である．

4.2 バーチャル・フォームの定義

ここでは外部オブジェクトとして ORDB の UniSQL データベース[9]と連係したバーチャル・フォームについて述べる．本研究ではパッドのファイルセーブ形式のバイナリ・ストリング・データをデータベースに格納できるように定義し，これによりパッドもセルに入るデータ型の 1 つとして利用可能になる．

データベースに対応するバーチャル・フォームは，1 枚のフォームがデータベースの 1 レコードを表す．フォームの各セルはレコードの各属性に対応している．**図 3** にデータベースへの検索機能を持つバーチャル・フォームの例を示す．バーチャル・フォームの台紙のパッドが次節の ProxyPad の一種であり，データベースにアクセスする機能を持つ．図の ID のセルに数値が入力されると，ProxyPad がデータベースにアクセスを行い，残りの 3 つのセルを補完する．

表 2 ProxyPad のスロット一覧
Table 2 The slot list of a ProxyPad.

dbFile	アクセスするデータベースファイル名を保持
class	データベース内のアクセスするクラス名を保持
attribute	クラスの属性と型からなるリストを保持
query	SQL 文を保持
result	検索の結果を保持
update	更新機能のためのトリガを受ける

4.2.1 ProxyPad

ProxyPad とはデータベースなどの外部オブジェクトにアクセスできるインタフェース･プログラムの機能を持つプリミティブパッドを指す．ProxyPad は外部オブジェクトへのアクセスに必要な変数や，アクセスの結果をスロットに保持する．ProxyPad として外部オブジェクトを表現することにより，従来の Wrapper の機能に加え，他の任意のパッドと同様に構築時だけでなく実行時においてもユーザが直接操作可能な可視オブジェクトとして扱うことが可能となる．ProxyPad で表現された外部オブジェクトと他のパッドや外部オブジェクトどうしの機能合成がパッドの貼り合わせによって容易に実現できる．こういった機能は従来の Wrapper には付加されていないものである．

図 3 の ProxyPad が持つスロットを**表 2** に示す．スロット #attribute には検索結果であるレコードの各属性値の型変換に用いる，クラス属性とその型の対からなるリストを保持している．このリストはスロット

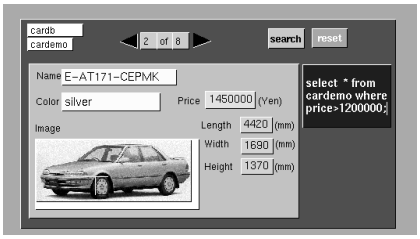


図 4 FormViewer の画面ハードコピー
Fig. 4 A display hardcopy of a FormViewer.

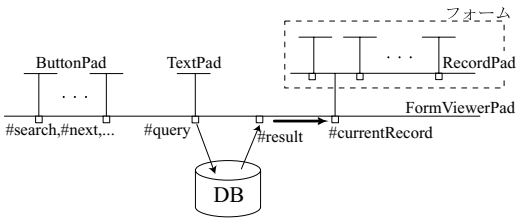


図 5 FormViewer の貼り合わせ構造
Fig. 5 The composition structure of a FormViewer.

#dbFile と#class にデータベースファイル名とクラス名を設定することで ProxyPad がデータベースから属性情報を取得する．そのとき同時に各属性名からなるスロットをパッドに作る．各種表示用パッドをこの各属性名のスロットにつなぐことでフォームを定義する．その後検索のキー属性のセルを指定する．

4.2.2 FormViewer

ProxyPad の異なる利用例としてデータベースからの検索データでリアル・フォームを作成する FormViewer を示す．FormViewer の構成例を**図 4** に，その貼り合わせ構造を**図 5** に示す．この例ではデータベースへのアクセス機能を持つ ProxyPad を台紙に，その上に検索結果を表示するフォーム，SQL 文を定義するパッド，アクセスのための ButtonPad が貼ってある．

この台紙である ProxyPad は検索結果のテーブルを保持するために表 2 のスロットのほかに**表 3** のスロットが必要となる．フォームのパッドはスロット#currentRecord に，SQL 文の定義パッドはスロット#query に，コマンド用の ButtonPad は各コマンド・スロットにそれぞれ結合している．

フォームの台紙の RecordPad は，連想配列をスロット#self に受けると，配列の要素をスロットとして追加する機能を持つ．各セルのパッドをこれらのスロットに結合してレコードの各属性値を表現する．

スロット#search に結合した ButtonPad から true が set されると，ProxyPad はデータベースに検索を行う．結果のレコード集合はスロット#result に格納

表 3 ProxyPad（FormViewer）に必要なスロット
Table 3 The slot list of a ProxyPad (FormViewer).

result	検索結果を配列として保持
currentRecord	result 内の cursor 番目のレコードを保持
cursor	currentRecord に入れるレコード番号を保持
next	cursor スロットの値をインクリメント
previous	cursor スロットの値をデクリメント
reset	フォームのセルの値をクリア
search	検索開始のコマンドスロット

される．この先頭レコードがスロット#currentRecord に格納され，スロット#cursor に 1 が格納される．

スロット#next または#previous に true を set することでスロット#cursor の値が増減し，スロット#currentRecord に格納するレコードも更新される．

FormViewer では，利用前にアクセスするテーブルを指定して，フォームのレイアウト定義を行う．スロット#currentRecord に RecordPad をあらかじめ結合しておいて，データベースファイル名とテーブル名を指定することで，スロット#currentRecord に値が空のレコードが格納され，RecordPad に各属性名をスロット名としたスロットが作成される．この RecordPad 上に各属性を表現するパッド貼り合わせてフォーム定義を行う．

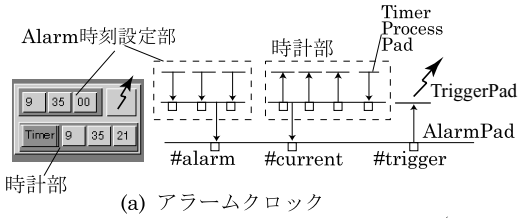
図 4 の例では，各属性の表現として TextPad，NumberPad，PadSaveLoadPad を結合している．PadSaveLoadPad は，パッドのファイル・セーブ形式のバイナリ・ストリング・データをスロットに受け取ると，そこからパッドを構成して自分の子パッドとするパッドである．この例の Image 属性である車のイメージは，画像でなく，パッドのセーブ形式をデータベースに格納したものを取り出し表現している．

検索の結果が表示された RecordPad をコピーすることで独立なリアル・フォームとして利用可能である．

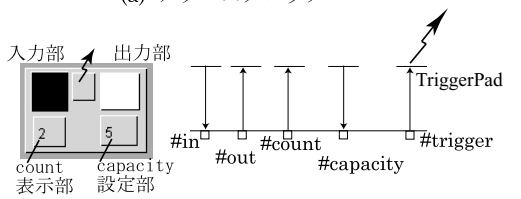
4.3 トリガ

本システムではトリガ部品も他の部品と同様にパッドによって構築する．トリガを発生するために TriggerPad を用いる．TriggerPad は状態更新から update によるトリガイベントを発生する．このトリガイベントを後述のフローベースパッドを用いてトリガを受信する側のパッドへと伝達する．

図 6 にトリガ部品の例を 2 つ示す．(a) はアラームクロックの例である．あらかじめスロット#alarm に予定時刻を設定しておく．TimerProcessPad から現在時刻がスロット#current へ set され，予定時刻に達した時点で AlarmPad の#trigger が true に更新する．これにより TriggerPad にトリガイベントが発生する．(b) は通過フォームの数が設定数に達した場合



(a) アラームクロック



(b) フォームカウンタ

図 6 トリガ発生部品
Fig. 6 Examples of trigger generating components.

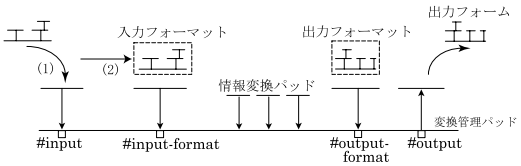


図 7 フォームコンバータの貼り合わせ構造
Fig. 7 The composition structure of a FormConverter.

表 4 情報変換パッドの基本演算
Table 4 The list of basic operations provided as ConversionFormulaPads.

addition	入力配列の合計を計算
summention	累計を計算
multiplication	積を計算
subtraction	配列の第一引数からそれ以外の差を計算
division	第一引数と第二引数の商を計算
maximum	配列の最大値を取得
minimum	配列の最小値を取得
average	配列の平均値を計算
count	入力の回数をカウント
aggregate	入力を保持して配列に追加
separate	入力の配列を分解
array	配列の順序を変更して再構成
unarray	配列の指定番目の要素を取得
dictionary	連想配列を構築
custom	スクリプトによる定義

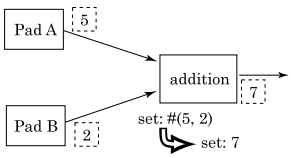


図 8 情報変換パッドへ渡されるデータの流れ
Fig. 8 The data flow among ConversionFormulaPads.

にトリガを発する例である．スロット#in から入ったフォームはそのまま#out から出力される．このとき#count の値がインクリメントされ，#capacity の値に達した場合に TriggerPad にトリガイベントが発生する．

4.4 フォームの変換

フォームの変換部品としてフォームコンバータ（以降コンバータ）を用いる．

コンバータのパッドの貼り合わせ構造を**図 7** に示す．台紙のフォーム変換管理パッドが変換方法の定義を管理している．このパッドはスロットとして#input，#output，#inputFormat，#outputFormat，#relation，#trigger を持つ．この子パッドとして，入力口，出力口のパッド，入力と出力のフォーム形式の定義のためのパッド，変換方法を定義する情報変換パッドを結合する．

#inputFormat，#outputFormat にはフォーム形式を定義するパッドを結合する．この子パッドとして貼られたひな型フォームのポインタが台紙のスロットに set され，変換に使われる．

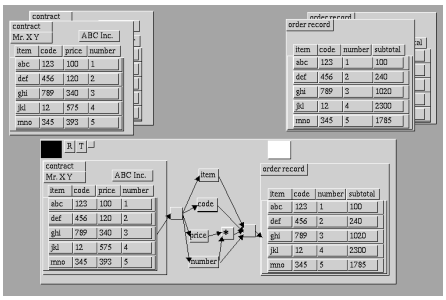
情報変換パッドは個々に単純な演算機能を定義してあり，これを複数枚組み合わせて目的の演算を定義する．現在，実装してある演算機能を**表 4** に示す．

台紙のスロット#relation には，ひな型や情報変換パッド間のデータの流れるリンクが連想配列として格納される．**図 8** を用いて変換時のデータの流れについて述べる．台紙のパッドが PadA，PadB の主スロット値を取得し配列にする．これを “addition” の情報変換パッドに送る．情報変換パッドは配列データを処理した結果（この場合は 2 つの和）を，自分の主スロットへ set する．台紙はこの値を取り出し次の処理へと送る．こうして変換が最後まで進められる．

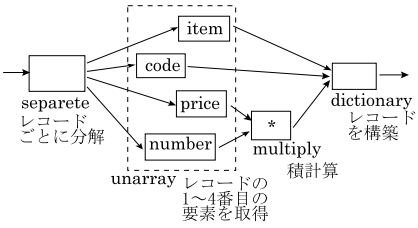
入力口のパッドは，自分に Drag-and-Drop されたパッドを自分のスロットに格納する．逆に出力口のパッドはスロットに set されたパッドのインスタンス・データからパッドを再構成して自分の子パッドとする．これによって台紙のスロット#input にフォームを set し，スロット#output から出力フォームを取り出す．

このように入力，出力フォームの形式と変換方法を定義すると，以降はこのコンバータでのフォーム変換が可能になる．

変換は，(1) 入力口へドロップされたフォームが台紙の#input に set される．(2) ひな型の入力フォームと形式を比較し，セルデータをひな型に転写する．(3) #relation のリンク情報に従い，台紙はリンク元からデータをリンク先へと set していく．この操作を繰り返して，全リンクの受け渡しを行う．(4) 全受け渡し終了時点で出力のひな型には出力フォームが完成して



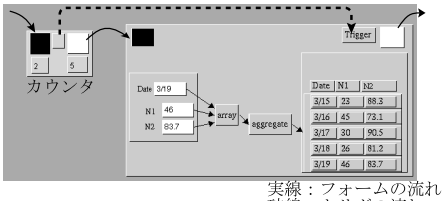
(a) 画面ハードコピー



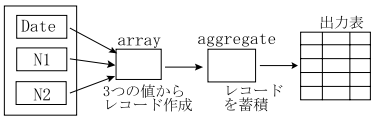
(b) 情報変換パッド部分の定義

図 9 契約書 → 注文書の変換例

Fig. 9 A display hardcopy of conversion from a contract form to an order form.



(a) 画面ハードコピー



(b) 情報変換パッド部分の定義

図 10 日報 → 週報への変換例

Fig. 10 A display hardcopy of conversion from daily reports to a weekly report.

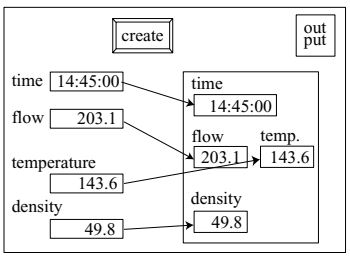


図 11 フォーム・ジェネレータの例

Fig. 11 An example form generator.

いる.(5) このひな型のコピーを#output へと set して，スロット結合を介して出力口にフォームがポップアップする．という手順になる．

出力フォームを#output にコピーするタイミングには,(1) すべての変換終了を契機にコピーと (2) スロット#trigger に結合した，トリガ受信のパッドからトリガを受けることでコピーとの 2 通りがある．

以下にコンバータの例を 2 つ示す．

図 9 に契約書の入力フォームから，注文書のフォームを出力するコンバータの画面ハードコピーを示す．入力と出力のフォームのひな形，フォーム入力口，出力口が 1 枚ずつと 7 枚の情報変換パッドから構成されている．情報変換部分では，一番左側の情報変換パッドが入力フォームのテーブルを行ごとに分解する．次の 4 枚が各行のデータから各属性値を取り出す．このうち price と number の積を計算して，最後に一番右側の情報変換パッドが 4 つの属性値から行データを再構成して出力フォームに集計したテーブルを渡して変換を完了する．

次に**図 10** に示すのは，#trigger への入力によるフォーム出力の例である．これは同一形式の日報フォームを入力とし，コンバータ内部に集計結果を蓄積していく．カウンタからのトリガを#trigger に受けることで週報を出力する．

4.2 節で述べたバーチャル・フォームをコンバータの入出力に利用できる．入力には検索機能を持つバーチャル・フォームを用いる．コンバータが入力されたバーチャル・フォームにフォームの値設定を要求する．バーチャル・フォームは外部オブジェクトにアクセスし，セルを満たす．このフォームデータをコンバータは変換に用いる．出力には，検索と更新の両方が考えられる．検索の目的で用いる場合には，一部のセルをコンバータが満たしたバーチャル・フォームは外部オブジェクトにアクセスし空欄を満たす．更新に用いる場合には，コンバータがすべてのセルを満たし，この情報で外部オブジェクトの状態更新を行う．コンバータはこれらの出力バーチャル・フォームを結果として出力する．

外部オブジェクトからの情報をフォームへ変換したり，逆にフォームのデータを外部オブジェクトへ出力したりするツールも，フォームからフォームへの変換でない特殊なフォームコンバータである．前者をフォーム・ジェネレータ，後者をフォーム・アナイアレータという．**図 11** にフォーム・ジェネレータの例を示す．これはプラント制御系の ProxyPad を用いて ‘create’ が押された時点の状態からなるフォームを作成する．

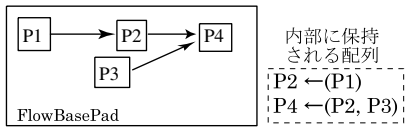


図 12 フローベースパッド上のリンクと内部に保持される状態
Fig. 12 Links on a FlowBasePad and their internal representation.

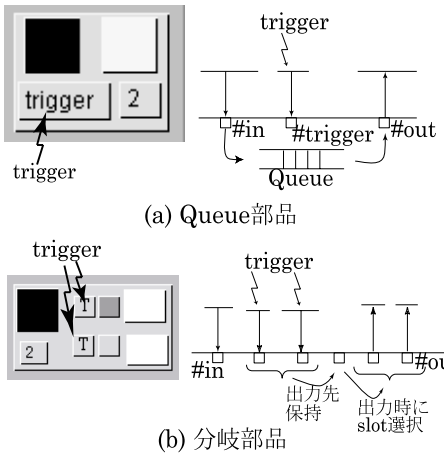


図 13 経路定義の部品の構造
Fig. 13 The mechanism of flow definition components.

4.5 フローの定義

フロー定義部品の台紙にフローベースパッドを用いる．フローベースパッドは自身の上に貼られている各パッド間にリンクを定義し，自身の内部に**図 12** に示すようなリンク元のパッドの PadID をキーに，リンク先の PadID を値にした連想配列で保持する．フローベースパッド上のパッド 2 枚を始点，終点として指定することで，リンク情報が追加登録される．このリンク情報から出力フォームやシグナルを次の入力へと伝達する．リンク元をツールの出力口パッドに，リンク先を次のツールの入力口パッドに指定することで，出力口にポップアップしたフォームを入力口のパッドへとドロップする．トリガの伝達はリンク元に TriggerPad をリンク先にツールに結合した ButtonPad などを指定する．TriggerPad の update イベント発生を検知して ButtonPad にこのイベントを伝達する．ButtonPad はこのツールに ‘click’ イベントを引き起こす．

分岐などの経路の形式を定義する部品もパッドとして定義する．これらの部品をフローベースパッドの子パッドとして利用することでさまざまな流れの形式を定義する．現在，本システムで実現している部品を以下に示す．

(a) Queue：フォーム入力と出力を一対とシグナル入力を持つ．入力フォームを自分の内部に Queue として蓄え，シグナル入力によりフォームを 1 枚出力する．Queue が空でシグナルを受けた場合にはフォームの入力があるとすぐにフォームを出力する．Queue 部品の貼り合わせ構造を**図 13** (a) に示す．

(b) 分岐：分岐部品は 1 つのフォーム入力と複数のフォーム出力，それに対応した複数のシグナル入力を持つ．フォームとシグナルの入力はそれぞれ Queue に蓄えられる．シグナル入力を受けることで対応するフォーム出力先を選択し，フォーム出力を行う．図 13 (b) にこの貼り合わせ構造を示す．

(c) 選択：選択は，分岐とは逆に複数の入力と 1 つの出力，それと出力先決定のシグナル入力を持つ．各フォーム入力とシグナル入力は Queue の機能を持つ．シグナルによってフォーム入力の 1 つを選択する．その入力フォームが空の場合には，フォーム入力があるまでシグナルを Queue に蓄えておく．

(d) 合流：合流も選択と同様に複数の入力と 1 つの出力を持つが，これは入力が行われた順に出力される．

(e) 複製：1 つの入力と複数の出力を持ち，すべての出力に対して入力されたフォームを複製して出力する．

フローベースパッドを用いたフローの定義例を**図 14** に示す．左側のコンバータから出力された積を追加したフォームを次の比較部品に入力．積が 500 以上か否かを判断し，True と False のパッドの片方からトリガを発する．このトリガを受けた分岐部品は右上と右下のコンバータへと，経路を選択する．このようにフローベースパッドで部品の出力と入力を結んで一連の作業の自動化を定義する．

4.6 リモート・リファレンスとボトムアップなシステム構築

前節までに述べた部品を用いて 1 階層の経路を定義可能となる．これにボトムアップなシステム構築を行うためにリモート・リファレンス部品としてメタパッドを導入する．これを用いて分散して存在する部品やフォームフローシステムを組み合わせてより統合されたフォームフローシステムを構築する．

4.6.1 メタパッド

メタパッド（meta）はただ 1 枚の任意のパッド（referent）と参照関係を持ち，単体では機能を持たない．meta と referent は互いの PadAddress を保持して参照相手を認識する．PadAddress はサイトアドレスと PadID の組である．PadAddress を用いて複数のシステム内からパッドを一意に指定できる．meta と referent の間で ‘paste’，‘click’，‘set’ などの特定イベントを転送し，パッド間でこのイベントを共有する．このようにメタパッドは特定のパッドの proxy として

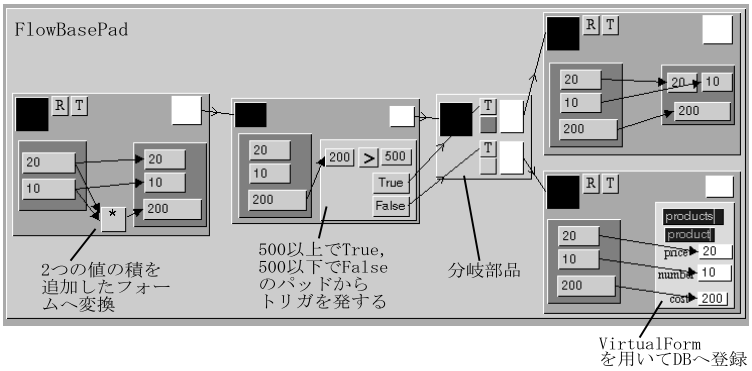


図 14 フロー定義の例
Fig. 14 An example flow definition using a FlowBasePad and Coverters.

動作する．

メタパッドは，対応するイベントによって以下に示す 6 つに分類される．

- ConverterMeta はコンバータの台紙を referent に持つ．コンバータと同じ名前のスロットを持ち，入出力口やトリガ用の ButtonPad を結合できる．スロットへの ‘set’ メッセージを referent と共有することで入力フォームの情報やトリガ，出力結果のフォームなどを互いに参照し合う．
- FormMeta はフォームの台紙である RecordPad を参照する．フォームの代わりにネットワーク内を行き来し，必要に応じてフォームの情報を呼び出す．FormMeta がネットワーク内を行き来することでフォーム本体をやりとりする場合に比べ通信量を抑えられる．
- InputMeta はパッドの入力口となるようなパッドを referent にする．InputMeta にパッドをドロップすると，referent の上に送り貼りつける．
- OutputMeta は InputMeta と逆に referent に貼られたパッドを自分の上に貼る機能を持つ．
- SignalInMeta は ButtonPad の meta となる．通常コンバータのリセットなどツールへの操作は ButtonPad を結合しておきクリックによって行う．SignalInMeta の ‘click’ 操作から ButtonPad に ‘click’ イベントを引き起こす．
- SignalOutMeta は referent のスロットの ‘update’ から meta に ‘click’ イベントを引き起こす．

meta のコピーは，対応する referent もコピーされ，コピーどうしが参照関係を持つ．referent のコピーは独立に referent 単体がコピーされる．複数のサイトから referent に meta を介してアクセスする場合，提供されている meta をコピーして用いるため，各々コピーされた referent へ独立にアクセスが行われる．コンバータのようなツール自身による処理では個々の referent が独立なツールとして働く．データベースなどの外部オブジェクトと連係している場合には，proxyPad からのトランザクション単位で外部オブジェクト側がアクセスのコンカレンシー制御を行う．よって複数の meta からの同時アクセスは不具合を生じない．

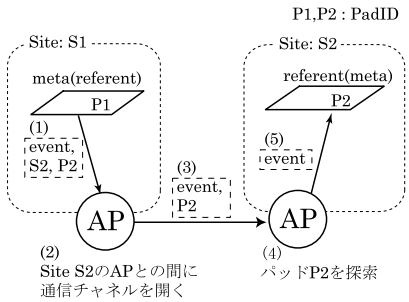


図 15 イベント転送の機構
Fig. 15 An event-transportation mechanism.

meta をネットワークを介して転送した場合には，新しい PadAddress を referent に伝え referent 内部の参照情報を更新する．

meta と referent の間でのイベントデータの転送機構を**図 15** に示す．イベントデータはアクセスポイント（AP）[10]を介して転送される．AP は各 IntelligentPad システムが 1 つずつ持つ通信管理オブジェクトである．AP 間の通信にはソケットを用いる．転送すべきイベント発生の場合，(1) 送信側のパッドはイベントデータと相手パッドの PadAddress からなるパケットを自分のサイトの AP に渡す，(2) AP はパケット内のアドレスから，転送先の AP と通信チャネルを開く，(3) パケットを転送先の AP に送る，(4) パケット内のパッド ID からデータを受けるパッドを探す，(5) イベントをパッドへ渡す，の手順でイベントデータが送られる．

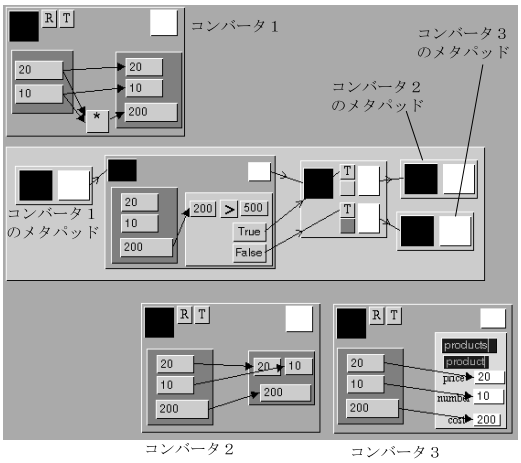


図 16 メタパッドを用いたフロー定義例
Fig. 16 An example flow definition using metaPads.

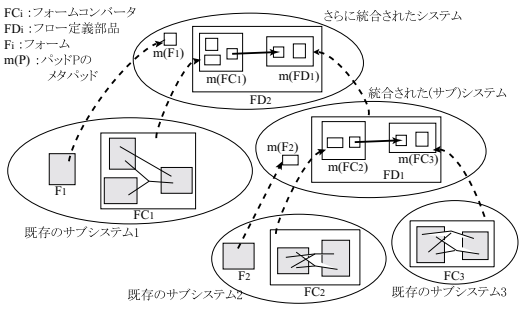


図 17 ボトムアップな構築
Fig. 17 An example bottom-up integration of distributed form converters and subsystems.

コンバータに対して作成した meta を他のサイトに転送することで，そのサイトからこちらのコンバータを利用した作業が可能になる．

メタパッドをフローベースパッドの子パッドとして用いることでネットワーク全体のツールを利用したフロー定義が可能となる．

4.6.2 ボトムアップなシステム統合

図 14 のフロー定義をメタパッドを用いて行った例を**図 16** に示す．このとき各コンバータとフローベースパッドは同じサイトに存在しなくてもよい．またコンバータに対してはメタパッドを作成しただけで何ら変更を加えていないので，コンバータ単体による従来どおりの処理も可能である．

図 17 にメタパッドを用いてボトムアップにフローを構築する例を示す．既存のサブシステム 2 と 3 からメタパッドを生成して 1 階層のシステムを構築している．このシステムからさらにメタパッドを生成してより大きなシステムへとボトムアップにシステムを統合できる．部品のフローベースパッドやコンバータは異なるサイトに存在でき，単体としての業務にも利用可能である．**図 18** にフォームフローシステムの画面ハードコピーを示す．図の上部にメタパッドを用いたフロー定義がある．3 つのメタパッドはそれぞれ中央左と下部のウィンドウに示すフォームコンバータのメタパッドである．このフローで用いる入力フォームと出力フォームの例が中央右のウィンドウに置いてある．

4.6.3 人手による作業の導入・統合

フォームへの処理として認証のように人手が介在する操作もフォームフローシステムに組み込み，統合する．処理を行う人へのフォームの受信箱とその人が処理済みのフォームを送り出すための送信箱を用意し，そのメタパッドを作成することで，人手をフローシステムに組み込むことができる．受信箱のメタパッドへのフォームのドロップにより，担当の受信箱へフォームを送付する．逆に送信箱へフォームを入れることで送信箱のメタパッドからフォームを取り出す．人手をフォームフローシステムに組み込むには，作成したメタパッドをコンバータの場合と同様にフロー定義に組み込むだけでよい．この場合もメタパッドの利用者はその対象がコンバータと人手どちらかを意識する必要がない．

4.7 WWW を用いた流通

この節では部品化メディア・アーキテクチャによる可視的コンポーネントのネットワークを介した流通機構について述べる．部品化メディア・アーキテクチャではコンポーネントによる WebBrowser が用意されている[11]．これは通常の WebBrowser と同様に読み込んだ HTML を解釈して表示する機能を持つ．パッドのセーブファイルを HTML の IMG タグを拡張する形で埋め込んでいる．パッドのセーブファイルが埋め込まれた HTML をこの WebBrowser で読み込んだ場合，セーブファイルを読み込み，自身の上に構築したパッドを貼って表示する．読み込まれたパッドは WebBrowser からはがして，他のパッドと機能合成を行うことができる．**図 19** に WebBrowserPad を用いた流通の例の画面ハードコピーを示す．左側の WebBrowserPad がメタパッドと入出力フォームの埋め込まれたページを表示している．そこから取り出した入力フォームとメタパッドを右側に示す．図に示すように取り出したパッドを自分の環境に合わせて編集して利用することができる．

5. 結　　論

本論文では，従来のフォームフローモデルに不足し

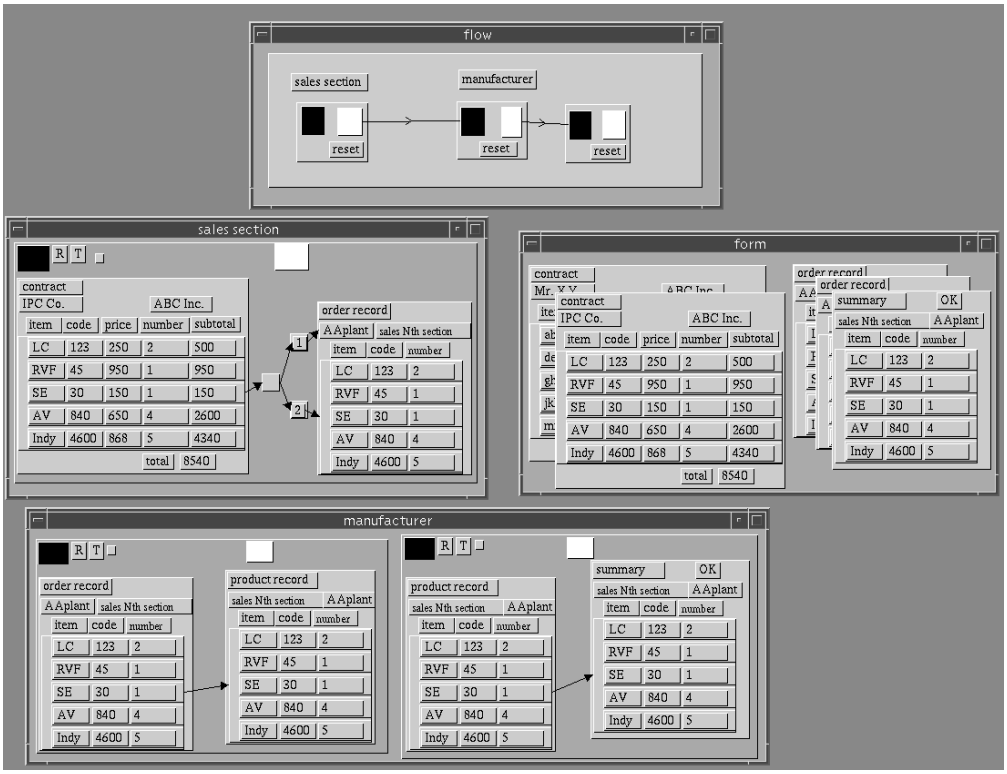


図 18 フォームフローシステムの画面ハードコピー
Fig. 18 A display hardcopy of an example form flow system.

ている機能を追加した拡張フォームフローモデルを提案した．そしてこのモデルを可視的コンポーネントの合成によってシステム構築するために必要となる要素部品とそれらの組合せ規則を定義することでジェネリックなフレームワークを定義した．このフレームワークの実現可能性を示すため，部品化メディア・アーキテクチャを実現している IntelligentPad システムを記述の基盤に用いて，このフレームワークを具体的に開発実現し，システム構築を行った．各部品のフレームワークとそれらを合成したシステムを提示することで実現可能であることを示した．これによりシステムの利用者が GUI 環境で可視的コンポーネントを Drag-and-Drop によって直接操作することでフローを定義し，システムを利用することが可能となる．

外部オブジェクトに対する Proxy 機能を持つバーチャル・フォームで，その外部オブジェクトの状態監視や制御をフォームフローシステムのフォームへの処理で行うことが可能になる．フォームという形の可視的コンポーネントによって Proxy を表現することで，システムの利用者でも直接操作によって容易に他システムとの連携，組替えを行うことが可能となる．その

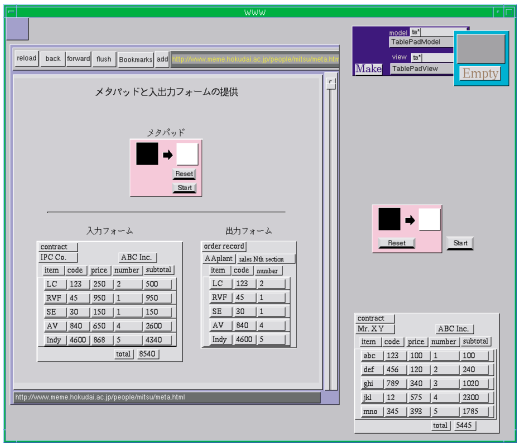


図 19 WebBrowser を利用したメタパッドと入出力フォームの取得画面のハードコピー
Fig. 19 A display hardcopy of an example WebBrowserPad.

際，通常のリアル・フォームと区別することなくシステムで利用可能である．

リモート・リファレンスを用いて既存のツールやフォームなどの部品を参照，制御して利用できる．リ

モート・リファレンスを可視的コンポーネントとすることで参照のみでなく，スロットの提供やイベント配信の機能を持ち，リモートに存在するコンポーネントから対象コンポーネントの機能を容易に利用することが可能となる．これによりネットワーク上に分散して存在する部品には変更を加えずにフロー定義に組み込むことができる．こうして構成した小規模のシステムからさらにリモート・リファレンスを適用してより大きなシステムをボトムアップに構築していくことができる．

これらのフォームや変換モジュール，リモート・リファレンスなどの HTML に埋め込まれたコンポーネントを，WebBrowser を用いて取り出せる．この取得したコンポーネントを自分の環境に合わせて再編集したり，編集したものを HTML に埋め込んで再流通したりできる．

また，人手が介在する操作も，処理を行う人手の受信箱と送信箱へリモート・リファレンスの仕組みを適用させることで，コンバータと同様にフォームフローシステムに導入・統合可能となった．

人手が介在する場合，リモートに送ったフォームの今存在する位置とその状態がどうなっているのかを知る必要が出てくる．この解決のために，位置や状態を知らせる機能を持つコンポーネントをフォームに付加して送る方法を適用したモデルを現在研究，構築している．

参 考 文 献

1) Zloof, M.M.: Office-by-Example: A business language that unifies data and word processing and electronic mail, *IBM Systems Journal*, Vol.21, No.3, pp.272–304 (1982).
2) Yao, S.B., Hevner, A.R., Shi, Z. and Luo, D.: FORMANAGER: An Office Forms Management System, *ACM Trans. Office Information Systems*, Vol.2, No.3, pp.235–262 (1984).
3) Sarin, S.K.: Object-Oriented Workflow Technology in InConcert, *IEEE COMPCON*, pp.446–450 (1996).
4) FileNET Corporation: *Visual WorkFlo*. http://www.filenet.com/images/pubimages/products/visual_workflo.pdf
5) Alonso, G., Agrawal, D., Abbadi, A.E. and Mohan, C.: Functionality and Limitations of Current Workflow Management Systems, submitted to *IEEE Expert* (1997).
6) Kappel, G., Pröll, B., Rausch-Schott, S. and Retschitzegger, W.: $TriGS_{flow}$ Active Object-Oriented Workflow Management, *Proc. 28th Annual Hawaii International Conference on System Sciences. Volume 2: Software Technology*, Los Alamitos, CA, USA, pp.727–736, IEEE Computer Society Press (1995).
7) Alonso, G. and Mohan, C.: *WFMS: The Next Generation of Distributed Processing Tools*, chapter1, pp.35–62, Kluwer Academic Publishers (1997).
8) 長崎 祥，田中 譲：シンセティック・メディアシステム：IntelligentPad，コンピュータソフトウェア，Vol.11, No.1, pp.36–48 (1994).
9) NTT データ通信株式会社：*UniSQL/X User's Manual*.
10) 長崎 祥：シンセティック・メディアとそのグループウェアへの応用，博士論文，北海道大学大学院工学研究科 (1994).
11) Tanaka, Y.: Meme Media and a World-Wide Meme Pool, *Proc. ACM Multimedia 96*, Boston, MA, USA (1996).

(平成 12 年 10 月 12 日受付)
(平成 13 年 9 月 12 日採録)

中川 光紀（正会員）

1993 年北海道大学工学部電気工学科卒業．1995 年同大学院工学研究科電気工学専攻修士課程修了．2001 年同大学院工学研究科電子情報工学専攻博士課程単位修得．現在，K-Plex Inc. に勤務．ソフトウェアの開発に従事．

田中 譲（正会員）

1972 年京都大学電気工学科卒業．1974 年同大学院電子工学専攻修士課程修了．工学博士．1974 年北海道大学工学部助手．1977 年同講師．1985 年同助教授を経て，1990 年同教授，現在に至る．1996 年北海道大学知識メディアラボラトリー長．この間，1985 年 10 月より 1 年間，IBM 社 T.J. ワトソン研究所客員研究員．ソフトウェア科学会，人工知能学会，米国 IEEE 各会員．データベース理論，データベース・マシン，並列処理アーキテクチャ，メディア・アーキテクチャ等の研究に従事．コンピュータ・アーキテクチャ等の著書あり．1994 年に IntelligentPad の開発に関して日経 BP 技術賞大賞受賞．