ヒューマンインタフェース 80－8
音声言語情報処理 23－8
（1998. 10. 17）

ラピッドプロトタイピングツールMuseのCASEへの適用

神尾広幸、雨宮美香、松浦博、新田恒雄
（株）東芝 マルチメディア技術研究所

あらまし
本稿では、マルチモーダルインタフェースの作成と評価を迅速に行うラピッドプロトタイピングツールMuse(Multimodal User-intefcace-design Support Editor) と、それを金融機関向け自動取引装置 (ATM: Automatic Teller Machine) のCASE (Computer Aided Software Engineering) 環境に適用した事例について紹介する。Museを用いたマルチモーダルインタフェースの構築は、画面設計、動作設計という2つのステップを通して行われる。画面設計のステップでは、UI-objectという部品を画面上に配置する。これによりUIの外観を設計していく。一方、動作設計のステップでは、UI-object間にリンクを設定する。リンクはリンク元のUI-objectで発生されるイベントとリンク先に送信するメッセージの対応関係を保持するオブジェクトであり、ユーザからの入力操作に対応するUIの動作は、すべてリンクによって記述される。Museによって設計されたUIのデータは、動作仕様記述ファイルとリソースファイルに変換して出力される。前者はUIの動作に関する情報を日本語のスクリプト言語で表現したものであり、後者は画面上に配置されるUI-objectの情報をMicrosoft Visual C++ のリソース形式で記述したものである。ATMのソフトウェア開発環境では、動作仕様記述ファイルからVisual C++のソースファイルを作成し、リソースファイルと合わせてコンパイルすることによって、ATM上で動作するプログラムを作成する。

Appling Muse, a Rapid Prototyping Tool to a CASE environment for developing Automatic Teller Machines

Hiroyuki KAMIO, Mika AMAMIYA,
Hiroshi MATSU'URA and Tsuneo NITTA
Multimedia Engineering Laboratory, Toshiba Corporation

Abstract
This paper describes about a Multimodal User-interface design Support Editor (Muse), and an instance of appling a Muse to a CASE(Computer Aided Software Engineering) environment for developing ATMs(Automatic Teller Machines). A developing UIs by Muse consists of 2 steps, or appearance designing step and behavior designing step. Under the appearance designing step, a developer designs the appearance by putting UI-objects on a screen. Under the behavior designing step, a developer set the link between UI-objects. The link is an object which preserves the correspondence of UI-object's event to a destination's message. Muse generates scenario files and resource files. The scenario file is described the UI's behavior which designed by links. The resource file is expressed the UI-objects' properties in the form of Microsoft Visual C++ resource. The GUI-generator which is the tool of ATM development environment translates the scenario files into Visual C++ source code. And the ATM's application produced with compiling the source code with resource files.

1 はじめに

金融機関において、自動取引装置はなくてはならないものである。当初は紙幣引出し専用の CD(Cash Dispenser) 機の導入から始まり、現在では紙幣・硬貨の入出金や現金振込、通帳記帳など様々なサービスを提供する ATM(Automatic Teller Machine) が広く普及している。また、"金融の自由化" に伴い業界内の競争もますます激しさを増しており、ATM は単なる合理化のツールとしてではなく、金融機関と顧客とを結ぶ重要なチャネルとしての役割を担いつつある。このため、ATM で行われるサービスの種類は年々増大し、ATM アプリケーションの仕様は頻繁に変更されている。

当社では、これらの仕様変更に迅速に対応し、かつ信頼性の高いアプリケーションを構築するため、ATM 向け CASE(Computer Aided Software Engineering) 環境を構築した [1]。

一方、筆者らは ATM に代表される社会情報システムのユーザインタフェース (UI) を容易に構築し、その評価・改良を迅速に行うラピッドプロトタイピングツール Muse(Multimodal User-interface design Support Editor)[2] を開発してきた。Muse では UI の構築作業のすべてを GUI 環境で行うため、プログラミングに不慣れな人でも簡単に UI を設計することができる。このため、営業担当者やシステムエンジニア (SE) が、顧客となる金融機関において ATM の仕様を打ち合せる際に利用されてきた。

今回は、ATM 上で動作するアプリケーションの開発効率化を図り、また SE が Muse を用いて作成した仕様の再利用促進を目的に、ATM-CASE への適用を試みる。また、将来的には金融機関に Muse を提供し、顧客自身がアプリケーションの開発を行うことができる EUC(End User Computing) 環境の構築を目指している。

本稿では、ATM-CASE の適用製品である自動取引装置 CW-500 の紹介と (2.)、ラピッドプロトタイピングツール Muse の説明 (3.)、

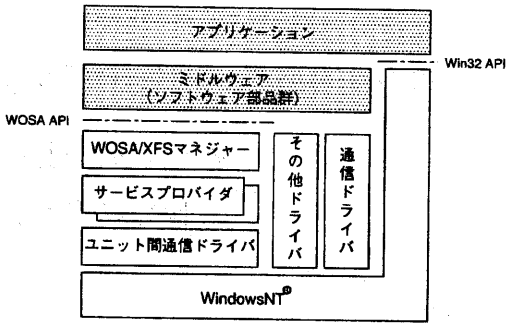

図 1: CW-500 のソフトウェア構造

Muse の ATM-CASE への適用 (4.) について報告する。

2 新型 ATM CW-500

金融機関を訪れる人の 80%は ATM の利用が目的であるといわれている [3]。このように利用者に最も身近な存在である ATM は、"より快適に" 利用できるものでなくてはならない。このため、ネットワーク機能やマルチメディア機能などを利用した高性能な UI を備えることが望まれている。

当社では、オープンプラットホームの採用と最新のマルチメディア技術の取り込みを積極的に行い、快適な操作性と新しいサービス提供を可能にした新型 ATM "CW-500 シリーズ" (以下 CW-500 と略記) を開発した [4]。CW-500 では汎用の入出力デバイスやソフトウェアコンポーネントを有効に利用するため、オープンプラットホーム上にシステムを構築している。制御部には IBM[1]PC/AT 互換ボードを採用し、OS として WindowsNT® [2]を搭載している。また、金融システムにおけるマルチベンダ環境でのアプリケーションプログラム開発効率化のためのアーキテクチャ WOSA/XFS(Windows® Open Services Ar-

[1]IBM は International Business Machines 社の商標です。

[2]Windows®, WindowsNT®は Microsoft 社の商標です。

chitecture eXtension for Financial Services) を採用している。CW-500 のソフトウェア構造を図1に示す。

3 ラピッドプロトタイピングツール Muse

Muse はマルチモーダルインタフェースの設計とプロトタイピングを迅速に行うことを目的とした、ラピッドプロトタイピングツールである (図2)。

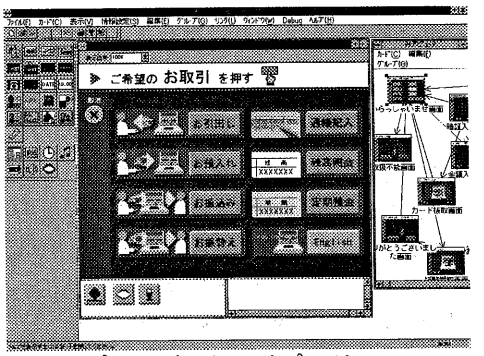

図 2: ラピッドプロトタイピングツール Muse

Muse には、画面や対話シナリオを設計する UI 設計機能と、設計した UI を Visual Basic®[3] (以下 VB と略記) のプログラムコードに変換し、動作確認するプロトタイピング機能を備えている。このため、アプリケーション開発の初期段階で UI 仕様の評価・改良を行い、優れた UI を迅速に開発することができる。本章では、UI 設計機能とプロトタイピング機能について説明する。

3.1 UI 設計機能

Muse を用いた UI の設計は、画面イメージの設計と動作シナリオの設計の2つのステップによって行われる。

画面イメージの設計ステップでは、画面の背景部を表すオブジェクトであるカード上にボタンやテキスト等の UI-object を配置することにより、画面の外観を決定していく。UI-object は固有のプロパティを保有しており、この値の設定も設計ステップで行う (図3)。また、Muse には音声認識部品、音声合成部品、指書き文字認識部品等のマルチモーダル入出力を担当する UI-object も用意されており、それらの UI-object をカード上に配置することにより、マルチモーダル入出力の機能を UI に追加することができる。

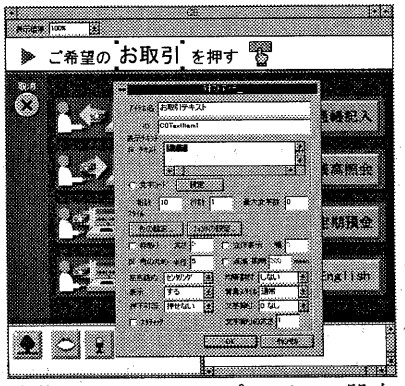

図 3: UI-object のプロパティ設定

動作シナリオの設計ステップでは、カード上の UI-object から他の UI-object やカードへリンクを設定していく。リンクとは、リンク元の UI-object が受けるイベントと、リンク先に送信するメッセージとの対応関係を保持するオブジェクトであり、そのリンクが有効となる条件も同様に保持している。図4は、Muse で金額を入力するカードを設計している例である。このカードには入力用の数字ボタンと、入力された金額を表示する "金額表示箱" という UI-object が配置されており、数字ボタンが押される度に金額表示箱に金額が代入されていく。このような動作シナリオは、各数字ボタンが "押された" イベントを受信すると、金額表示箱に "一文字追加" メッセージを送信する、というリンクによって表現される。

[3] Visual Basic® は Microsoft 社の商標です。

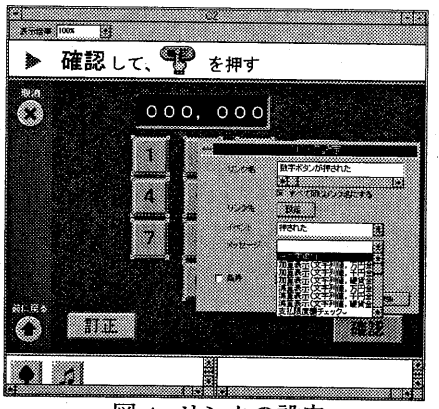

図 4: リンクの設定

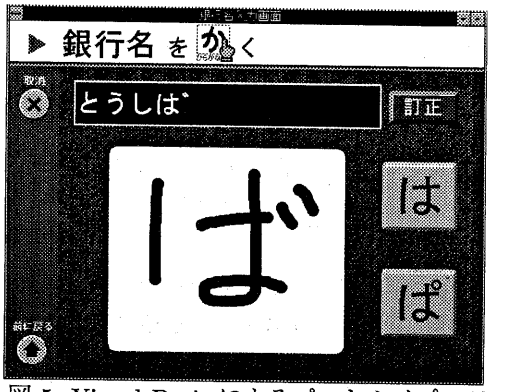

図 5: Visual Basic によるプロトタイピング

3.2 プロトタイピング機能

設計された UI は、Muse によって VB のプログラムコードに変換される。このコードを VB 上で動作させることによってプロトタイピングを行う。Muse のデータは表 1 の対応関係に基づいて VB のコードへ変換される。Muse 上で用いられる UI-object には各々対応したカスタムコントロール（OCX）が用意されており、Muse は Form 上にこれらの OCX が配置されたコードを生成する。また、UI-object 間に設定されているリンクについては、リンク元のイベントを OCX が発生するイベントとして、リンク先に送るメッセージを OCX が持つメソッドとして、コード内に記述する。この変換作業により、Muse のデータから VB のコードが生成される。図 5 に、VB によるプロトタイピングの様子を示す。

表 1: Muse データと VB データの対応関係

Muse	Visual Basic®
カード	Form
UI-object	OCX
イベント	OCX のイベント
メッセージ	OCX のメソッド

4 ATM-CASE への適用

前章で説明した通り、Muse は UI の設計を迅速に行うラピッドプロトタイピングツールである。ATM-CASE への適用においても、Muse は CW-500 の UI 開発を担当するツールとして利用されている。本章では、CW-500 の ATM 用フレームワークの構造と、Muse と ATM-CASE の接続について説明する。

4.1 ATM 用フレームワーク

CW-500 の開発では、オブジェクト指向技術を用いて ATM 用フレームワークを構築し、これに基づいてアプリケーションの設計を行っている。ATM 用フレームワークは、アプリケーション内部を役割に応じて 4 つのカテゴリに分割している。この 4 つのカテゴリと役割を以下に示す。

- **サービスカテゴリ**
  取引種別毎の制御
- **ユニットカテゴリ**
  紙幣・硬貨・通帳ユニット等の制御
- **媒体カテゴリ**
  紙幣・硬貨・カード等の媒体情報の制御
- **GUI カテゴリ**
  接客部の GUI 制御

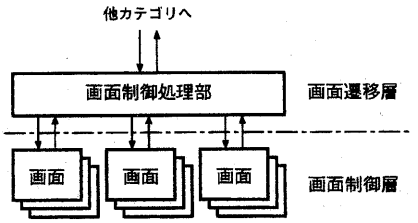

図 6: GUI カテゴリの内部構造

これらのカテゴリの中で、Muse は GUI カテゴリの設計に用いられている。

GUI カテゴリは、画面制御層と画面遷移層という 2 つの層に分割された構造になっている（図 6）。画面制御層は画面内の動作を制御するライブラリの集合であり、画面制御層内には画面の数だけライブラリが存在する。このライブラリは、画面の表示や画面内の処理について実装されたものである。また、画面遷移層ではこのライブラリのロード・アンロードを制御することにより、画面の切替えを行う。

4.2 Muse と ATM-CASE の接続

Muse で作成した UI のデータは、画面制御層のライブラリ作成に用いられている。

まず、Muse で設計された画面イメージは、Visual C++® [4]（以下 VC++と略記）のリソースファイル形式に変換して出力される。3.2 で述べた VB への接続と同様に、カード上に配置された UI-object は、各々に対応した OCX としてリソースファイル内に記述される。

一方、Muse で設計された動作シナリオは、図 7に示す画面制御仕様記述ファイルに変換して出力される。仕様記述ファイルには日本語で処理の流れが記述されており、ATM-CASE ではこの仕様記述を GUI-Generator というツールによって VC++のソースコードに変換する。このソースコードとリソースファイルをコンパイルすることによって、画面制御層のライブラリを作成する。

[4]Visual C++® は Microsoft 社の商標です。

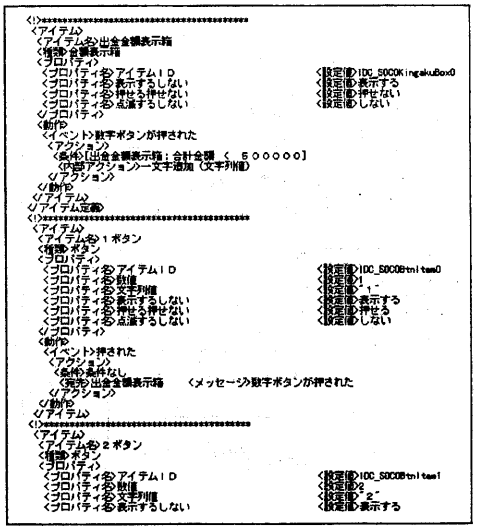

図 7: Muse から出力された画面制御仕様記述ファイル

4.3 ATM-CASE 適用の効果

Muse を ATM-CASE に適用したことにより、CW-500 の画面制御層開発の工程は図 8 のようになった。画面制御層の設計者は、画面毎の処理を分析し、Muse を用いて画面の設計を行い、GUI-Generator でソースコードを生成し、VC++でコンパイルする、という手順で画面制御のライブラリを開発する。

Muse の ATM-CASE 適用によって、以下のような効果が現われている。

1. 画面設計の簡易化

画面は、Muse を用いることで GUI 操作によって設計できるため、従来と比較して簡易化されている。このため、設計に要する期間を短縮できた。

2. 仕様変更の容易化

画面制御層では 1 画面毎にライブラリを用意しているため、新たなサービスを追加する等の仕様変更が発生した場合でも、Muse を

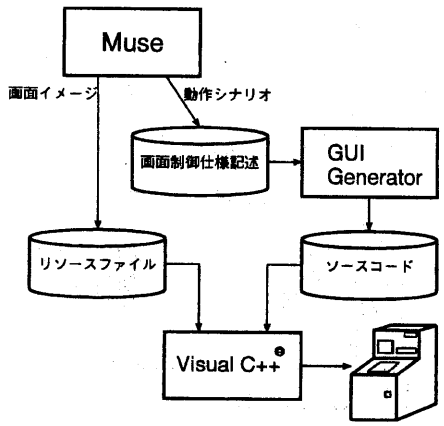

図 8: 画面制御層開発の工程

用いて新たな画面を作成し、画面制御層に追加することで迅速に対応することができた。

3. 仕様の理解性向上

Muse では動作仕様はすべてリンクによって記述されているため、UI-object 間のリンク関係を把握することで画面内の動作の理解性を向上できた。また、Muse から出力される画面制御仕様記述ファイルは日本語で表現されているため、画面内の動作全体を把握する際にも理解しやすくなっている。

4. 不具合の減少

ATM-CASE 環境により、アプリケーション開発者は、直接プログラミングを行わずに Muse のデータからソースコードを自動生成できるようになっている。このため不具合が大幅に減少した。

5 まとめ

ラピッドプロトタイピングツール Muse を、自動取引装置 CW-500 の開発環境である ATM-CASE に適用した。これにより、CW-500 アプリケーション開発における大幅な効率化が図られた。

今回 Muse を CW-500 開発環境に適用できた理由としては、ATM 用フレームワークの構造に依るところが大きい。ATM 用フレームワークでは GUI カテゴリを他のカテゴリから分離し、独立性を保っている。また GUI カテゴリ内でも、画面制御層を画面遷移層と分離することにより、画面間の独立性を保っている。このため、画面単位で UI の設計を行う Muse との親和性が高く、適用が可能であったと思われる。また、画面制御層のライブラリ群においても、再利用性を考慮して OCX による処理の部品化が図られている。これは UI-object 間のリンクによって処理を記述する Muse の考え方と一致している。この ATM 用フレームワークのように、オブジェクト指向分析によって適度に分割された構造は、仕様変更に強い柔軟なシステムの作成を可能にする。今後はこのフレームワークを応用し、他製品の開発環境の整備と Muse の適用を進めていきたい。

参考文献

[1] “次期現金自動取引装置向け CASE”, 東芝レビュー, Vol.52, No.3, pp.30 (1997).

[2] H.Kamio, et al. “Muse, a Rapid Prototyping Tool”, Design of Computing Systems: Social and Ergonomic Considerations, 21B, pp.569-572 (1997).

[3] “金融機関におけるインタフェースデザインの総合アプローチ”, 富士通ジャーナル, Vol.19, No.4, pp.26 (1993).

[4] 舩津ほか, “最新のソフトウェア技術を活用した次期 ATM CW500”, 東芝レビュー, Vol.52, No.10, pp.59-62 (1997).