3階層アプリケーション開発環境 －HYPERPRODUCE II－ クライアント開発

2J－3

森山令子 菅野幹人 上田尚純 大江信宏
三菱電機株式会社 情報通信システム開発センター

1.はじめに

HYPERPRODUCE II（以下HP II）は、3階層基幹業務アプリケーションを効率よく開発する環境を提供するミドルウェアである。本報告では、HP IIの機能の一部であり、クライアント上で動作するGUIアプリケーション（以下APP）を生成するClient Program Wizard（以下CPW）について報告する。

2.特長

図-1にCPWの機能概要を示す。

① 画面自動レイアウト機能　画面の表示イメージを作成する機能。Visual Basic*1(以下VB)で読み取り可能な画面情報ファイルを生成する。画面設計の省力化を目的とする。

② 画面ロジック生成機能　画面上の部品のイベントに対して動作を設定する機能。動作の定義は外部ファイルで行う。定義した外部ファイルは他APPで流用することが可能である。設定した動作はVBのロジックとして生成される。

*1 米国マイクロソフト社の登録商標

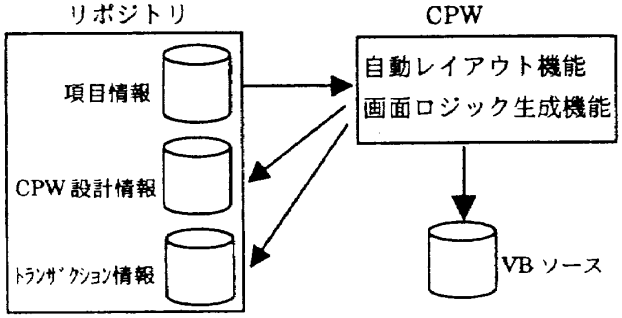

図-1 CPWの機能概要

Developing system for three tier business Applications
－HYPERPRODUCE II for Client Part－
Ryoko Moriyama　Information and Communication systems development center
Mitsubishi Electric Corporation

3.実現方式

① 自動レイアウト機能

CPWでは、リポジトリ上の項目情報から画面で必要な全ての項目を選択し、選択した各項目に画面作成に必要な属性を付加することにより自動レイアウトを行う。項目の選択時に、まとめて配置したい項目で1つのグループを作成し、複数のグループで1枚の画面を構成する。CPWで各項目に追加される属性には、個別に設定されるフォントなどの属性と、グループごとに設定される属性がある。

業務用APPの画面には図-2のような3つのカテゴリがあり、カテゴリ情報はグループの属性となる。

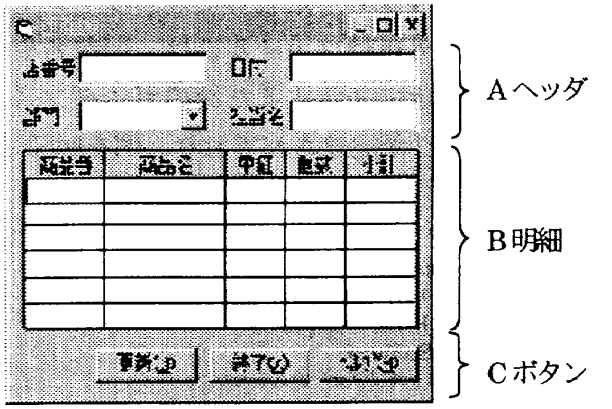

図-2 業務用APPの画面例

A ヘッダ　入出力可能な単項目が並んでいる部分。隣の見出しラベルとは対で扱う。

B 明細　表形式の入出力部品。複数の項目で1つの表を形成し、各項目が列に対応する。

C ボタン　複数のボタンが並んでいる部分。

グループのカテゴリである程度枠が決まり、さらに各項目の属性を設定して詳細を決める。ヘッダグループの場合にはコントロールの種別や見出しラベルの有無等の指定が必要であり、

その他見栄えのための色やフォントの指定も必要となる。

画面はグループの並び順で配置される。グループの順番やグループ内項目の順番は CPW 上で自由に変更できる。以上の情報より、レイアウト確認画面にて、画面レイアウトイメージを表示する。この時、カテゴリがボタンの場合には大きさを均等にするなど、グループのカテゴリを考慮したレイアウトを作成する。レイアウト確認画面では、グループごとにコントロールの縦・横並びを変更したり、全体の間隔を調整する。設定した内容は随時レイアウト確認画面で表示できる。

② 画面ロジック設計

画面ロジック設計では、ビジネスロジック APP の呼び出し処理や画面の初期化処理など GUI APP として必要な処理を項目情報に追加する。これには業務用 APP のために必要な関数を決められたフォーマットで定義した外付けファイルを使用する。外付けファイルには、関数名や引数情報、CPW 上で表示するためのガイダンス、実際のロジックが記述されており、例を図-3 に示す。

```
Version A00
Division:コントロール用
DivInformation:コントロール専用外付けファイル
Title:フォーカスセット（文字列選択なし）
ProcedureName:cpwSetFucus
Information:コントロールにフォーカスをセット
ProcefureMode:Sub
ParamNum:1
Param1:CONTROL;フォーカスをセットするコントロールを指定
Inline:0
Code:
Public Sub cpwSetFosuc(ctlControl As Control)
  'パラメータに指定されたコントロールにフォーカスをセット
  ctlControl.SetFucus
End Sub
```

図－3 外付けファイルの例

CPW では外付けファイルに定義されている関数を、項目情報としてコントロールのイベントに割り付ける作業を行う。1つのイベントに対して複数関数の割り付けが可能である。また、割り付けられた関数の実行部分は VB のソースファイルとして生成される。

1つの外付けファイルに対し複数関数を定義したり、複数の外付けファイルを扱うことができるため、一度作成した外付けファイルを別 APP に流用することが可能である。

4.効果

図-4 に CPW 使用時・不使用時の開発時間測定結果を示す。項目数が 20 個、40 個、60 個のそれぞれの場合について測定を行った。1項目が複数コントロールに展開されることがあるため作成した画面上のコントロール数は CPW で扱う項目数より多くなっている。ロジックは終了処理と画面初期化処理のみとする。

項目数*2	CPW 不使用		CPW 使用		生産性		作成
	A:画面レイアウト	B:画面ロジック	C:画面レイアウト	D:画面ロジック	C/A	D/B	コントロール数
20	23	8	9	2	0.39	0.25	44
40	50	15	18	2	0.36	0.13	69
60	105	24	34	2	0.32	0.08	130

*2 ( )内はトランザクション項目数(上り個数,下り個数)

図-4 CPW 使用・不使用時の開発時間(単位:min)

図-4 において C/A, D/B を CPW を使用した場合の生産性とみなすことができる。開発時間比であるので数値の小さいほど生産性が高いことになる。画面生成における生産性（C/A）は項目数が多いほど効果的である。これは項目数が多いとコントロールの位置揃えの手間が大きくなるからである。CPW 不使用時のロジックの記述は項目数に応じて多くなるが、CPW 使用時は関数の呼び出しを設定するだけであるのでほとんど変わらない。

項目数が多く複雑な画面において、CPW の使用効果が相対的に高まるといえる。

5.まとめ

今後は、グループのカテゴリの種類を増やすことによう画面バリエーションを充実化など、開発ツールとして強化させたい。