機能デザインとビジュアルデザインを分離したGUI開発ツール

5L－1

永森光晴　阪口哲男　杉本重雄　田畑孝一
図書館情報大学

1. はじめに

これまでに，GUIプログラミングの複雑さを軽減するためのさまざまなツールキットやGUI構築ツールが開発されてきた．しかしながら，これらのツールには，(1)プロトタイピングがしづらい (2)GUIのカスタマイズがしづらい (3)GUIの再利用がしづらい，などの問題点がある．これらの問題点は，現在のGUIプログラミングモデルにおいて，GUIの論理的構造がプログラムコードの中に埋もれてしまっていることや，応用プログラムの処理の作成とGUIの作成を明確に分離していないことが原因である[3].

本稿では，これらの問題点を解決するために，応用プログラム処理の作成とGUIの作成を分離したGUIプログラミングモデルを提案し，それに基づくGUI構築ツールについて述べる．

本研究では，GUIを応用プログラムの処理とGUIの間の値の受渡しを行なう界面と，GUIの画面表示と利用者に操作方法を提供する界面の２つの界面を持つオブジェクト（ユーザインタフェースオブジェクト(UIO)）であると捉える[1][2]．UIOは，応用プログラムから見たGUIの論理的構造と機能を表す記述と，実行環境で利用されるウィジェットとその属性値を表す記述によって定義される (図１).

2. 機能デザインとビジュアルデザインを分離したGUIプログラミングモデル

現在のモデルでは，ウィジェットの物理的構造の各部分が応用プログラムにおいてどのような役割を持っているのかわからない．また，応用プログラムの処理とコールバック関数が複雑に入り組み，明確に分離されていない．応用プログラムの観点から見ると，GUIの外観や操作方法は，あまり意味を持たないので，GUIプログラミングの初期段階では，応用プログラムの処理を表現するGUIの機能に重点をおき，外観を定めるためのウィジェットの選択や属性値の設定などに煩わされることは望ましくない．すなわち，

・GUIから応用プログラムが適切な値を受け取れるか（入力）
・応用プログラムからGUIに適切な値を渡せるか（出力）

をテストすることのみに注意を向けながらGUIを構成していくことが望ましい．一方，応用プログラムとのコミュニケーションのためにGUIが満たすべき機能に煩わされることなしに，GUIの外観や操作方法を作り上げられることが望ましい．

以上のことから，本研究では，GUIの論理的構造をプログラムコードに反映させ，GUIの開発過程を応用プログラムの処理の観点から見た論理的機能の作成と，外観と操作性（Look-and-Feel）の作成の２つの過程に分離したプログラミングモデルを提案する（図２）．前者は，いわばフレームワークだけからなるGUIのプロトタイプを作成する過程であるので，プロトタイピング過程と呼ぶ．後者は，外観と操作性のみを作成する過程であるので，ビジュアルデザイン過程と呼ぶ．プロトタイピング過程では，応用プログラムの処理と切り離してGUIの論理的機能をシミュレーションし，その機能を検証する．そのため，ビジュアルデザイン過程では，GUIの論理的機能が正しく動作することが既に検証されているので，GUIの論理的構造に従ってGUIのカスタマイズを行なうことが可能である．また，本研究で提案するモデルでは，GUIを効率よく開発し，GUIの再利用を図るためにUIOを蓄積するためのデータベース（UIO-basc）を用意する．

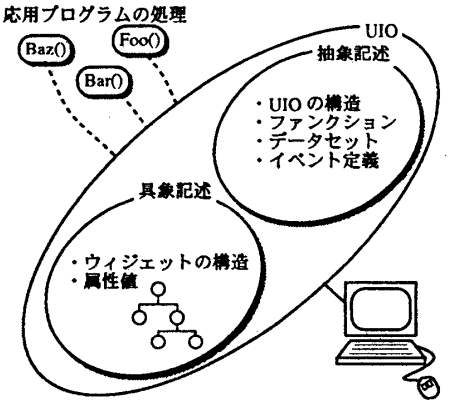

図１　UIOの概念図

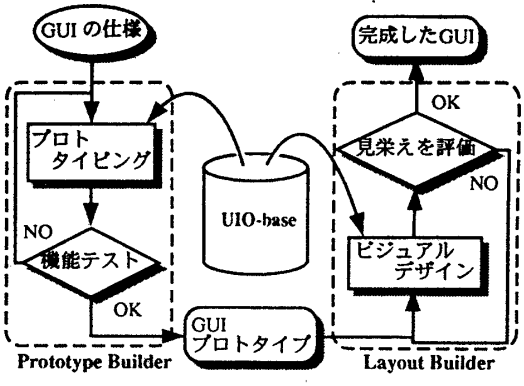

図２　UIOに基づくGUIプログラミングモデル

A Two Phase GUI Development Tool: Separating Framework Prototype Design and Visual Product Design.
Mitsuharu NAGAMORI, Tetsuo SAKAGUCHI, Shigeo SUGIMOTO, Koichi TABATA
University of Library and Information Science

3. UIO に基づく GUI 開発ツール：ASSORT

3.1. ASSORT の構成

現在，UIO と，UIO を利用した GUI プログラミングモデルに基づくユーザインタフェースビルダ ASSORT (Application Semantics Specification Oriented Tool) の開発を進めている．図2に示すように，ASSORT は，プロトタイプビルダ，レイアウトビルダ，UIO-base の3つのツールで構成されている．

プロトタイプビルダは，プロトタイピング過程に用いるツールで，UIO の構造の定義や機能的役割の定義と，その動作をテストする役割を持っている．レイアウトビルダは，ビジュアルデザイン過程に用いるツールで，プロトタイプビルダで作成した機能的に完成している GUI のカスタマイズを行う．UIO-base は，プロトタイプビルダとレイアウトビルダの両方で UIO を再利用するために用いるツールである．

3.2. プロトタイプビルダ

プロトタイプビルダでは，マウスを使って対話的に UIO の構造の作成し，個々の UIO が持つ機能的役割を定義する(図3)．プロトタイピング過程では，ボタンやメニュといったウィジェットの種類や見栄えは重要ではない．そのため，プロトタイプビルダでは，矩形を使って UIO を表現し，それらを組み合わせることによって UIO の構造を作成する．矩形の包含関係は UIO の親子関係を表している．プロトタイプビルダでは，以下の操作を行なう．

(1) 矩形を組み合わせて UIO の階層構造を作成する
(2) 利用者から受け取ることのできるイベントを定義する
(3) 論理的機能定義が正しいかどうか検証する

プログラマは，作成した矩形で表現される UIO に対して実際にマウスやキーボードから入力を行ない，(1)，(2) で定義した論理的機能が正しく動作するかを検証する．

3.3. レイアウトビルダ

レイアウトビルダでは，機能的に完成している UIO の GUI をカスタマイズする (図4)．レイアウトビルダは，プロトタイプビルダが出力した論理的機能が完備している UIO の記述を読み込み，その記述に従って実際に利用者が操作をするためのウィジェットを用いた GUI を作成する．プログラマは，レイアウトビルダが提示した GUI の見栄えが，応用プログラムの仕様やスタイルルールにそぐわない場合，他の Look-and-Feel を持つ GUI に置き換えることができる．また，レイアウトビルダでは，現在の GUI ビルダと同様に，GUI を構成しているウィジェットのレイアウトや配色，表示する内容などの物理的属性をカスタマイズする．

3.4. UIO-Base

作成した UIO は，UIO-base に蓄積し，再利用する．UIO-base は，プロトタイプビルダとレイアウトビルダの両方で利用され，UIO の検索，登録，削除，変更などの機能を提供する．

プロトタイプビルダでは，抽象記述や UIO 構造木を検索キーにして UIO-base に蓄積されている既存の UIO の中から同等の機能を持つ UIO を検索する．レイアウトビルダでは，GUI のカスタマイズのために，同一の抽象記述を持ち，かつ異なる具象記述を持つ UIO を検索する．

4. おわりに

本稿では，UIO の概念と UIO に基づいた GUI プログラミングモデルについて述べた．このモデルでは，応用プログラムの処理の作成と GUI の作成を明確に分離することができた．また，応用プログラムの機能的役割の観点から GUI のカスタマイズが可能であることを示した．今後は，より多くの UIO を UIO-base に蓄積し，UIO-base の検索性能などの評価を行なっていく．

参考文献

[1] 永森，ほか，アプリケーションセマンティクスに基づく GUI のカスタマイズ．第47回情報処理学会全国大会，1994，1D-8.

[2] Nagamori, et al., Customizing User Interfaces based on Application Semantics: An Approach to Reuse Graphical User Interfaces. 47th FID, 1994, pp.301-305.

[3] 永森，ほか，GUI のカスタマイズを指向したユーザインタフェースビルダ．電子情報通信学会ソフトウェアサイエンス研究会，1995，SS94-42.

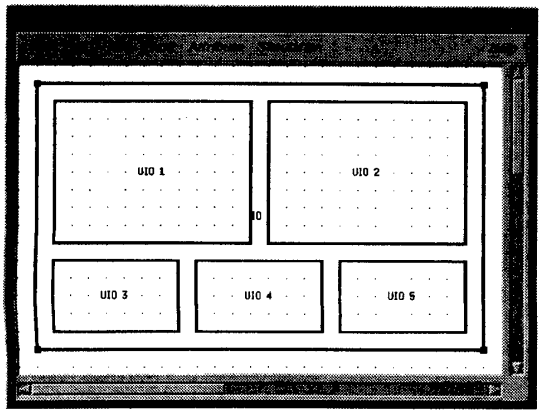

図3　プロトタイプビルダのウィンドウイメージ

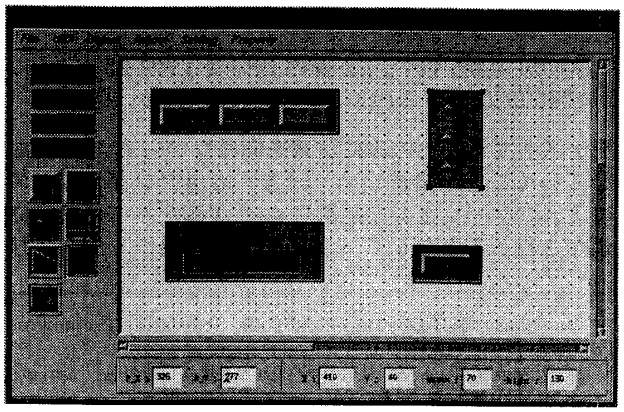

図4　レイアウトビルダのウィンドウイメージ