7F-4 アイコンプログラムの作成・実行環境の開発

岩田誠司* 吉本岩生* 平川正人** 田中稔** 市川忠男**
*広島大学大学院 **広島大学工学部

1．はじめに

計算機の普及に伴って、非専門家にも使いやすい計算機システムの重要性が高まってきている。このようなシステムを構築するにあたっては、ユーザインタフェースに視覚情報を用いる方法が有効であると考えられる。

この一つのアプローチとして、アイコンと呼ばれる絵シンボルを用いるものがある。アイコンはまた、それに結びつけられた機能やデータを表しており、アイコンを画面上で指示することによって、該当する機能やデータをアクセスすることができる。従来のコマンド方式に基づくユーザインタフェースと比べて、ユーザとの親和性に優れていると言える。

アイコンの応用は単にコマンドの視覚化にとどまらず、プログラミング環境においても有効に機能しうる。Pict[1]などにその具体例を見ることができる。そこでは、2次元ディスプレイ上でのアイコンの選択・配置ならびにアイコン間の接続関係の規定を行なうことがプログラミングである。このようなプログラミング環境を実現するにあたっては、アイコンの管理、操作、解釈を行なう機能が必要となる。

本研究では、アイコンを管理、操作、解釈し、アイコンプログラムの作成・実行を支援する環境を開発した。

2．アイコンプログラム

以下に、手順をおって、プログラムの作成過程についての説明を行なう。

(1)ユーザはマウスを用いてアイコンメニュ(画面右)の中から必要なアイコンを選択し、それをプログラミング領域(画面左上)の所望の位置に移動・配置する。(図1)

(2)システムは配置されたアイコンが実行可能であれば直ちにそれを実行し、実行結果を表示する。ここで、実行結果もアイコンとして管理される。(図2)

(3)ユーザは(2)で得られた結果を参照し、再び(1)の操作を行って必要なアイコンを選択・配置するとともに、アイコン間の接続関係(データの流れ)を規定する。システムは規定された接続関係に従ってアイコン間を矢印で連結し、実行結果を画面上に表示する。(図3)

(4)上記の(1)から(3)の操作を繰り返し行なう。

以上のように、ユーザはシステムと視覚的なやりとりを繰り返し行なうことにより、容易にプログラムを作成することができる。このようにして作成されたプログラムをアイコンプログラムと呼ぶ。作成されたアイコンプログラムを1つのアイコンとして定義することができる。ユーザは新しく定義されたアイコンを用いて、より大きなアイコンプログラムを作成することができる。

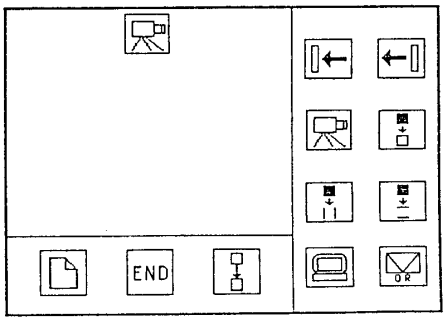

図1．画像入力アイコンの配置

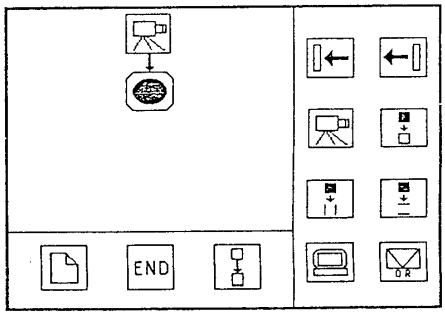

図2．画像入力アイコンの実行結果表示

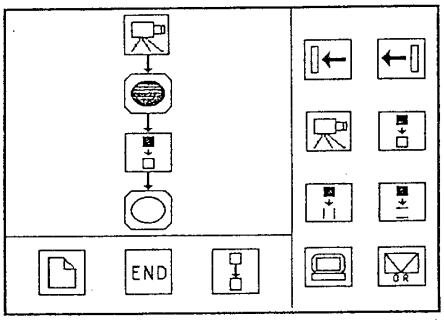

図3．エッジ検出アイコンの接続とその実行結果表示

本システムが提供するアイコンプログラミング環境は以下の特徴を持っている。

- アイコンを用いて画面で直接プログラムの作成を行なうことができる
- 記述されたプログラムはデータの流れを視覚化しており、処理の流れが理解しやすい
- 結果を見ながら、対話的にプログラムを作成していくことができる

Development of Iconic Programming Environment
Seiji IWATA, Iwao YOSHIMOTO, Masahito HIRAKAWA, Minoru TANAKA, and Tadao ICHIKAWA
HIROSHIMA UNIVERSITY

3．アイコン

アイコンは絵シンボルを表わしていると共に、それから連想されるオブジェクトを示している。絵シンボルの決定にあたっては、ユーザがそれを見て直観的にオブジェクトの内容を理解できるようにする必要がある。

アイコンはシステム内ではディスクリプタによって管理されている。アイコンの解釈、実行、あるいはアイコンに対する操作(移動、挿入、消去等)はすべてディスクリプタの情報に基づいて行なわれる。

本システムではアイコンを、それが表しているオブジェクトの型によって、以下の７種類の型に分類している。

(1) データアイコン
文字、数値、図形等の実際のデータを表現するアイコンである。プログラミングの途中で逐次得られる実行結果もデータアイコンとして定義される。

(2) データタイプアイコン
データタイプを表現するアイコンである。データタイプは、一般にいう整数型、文字列型などの物理的なデータ構造の違いだけではなく、データの意味(概念)によっても詳細に区別される。また、プログラム作成過程において、具体的な入力データが与えられなかった場合には、システムはアイコンの実行結果として、該当するデータタイプのアイコンを出力する。

(3) プリミティブアイコン
システムにあらかじめ用意されている最も基本的なオブジェクト(機能)を表現するアイコンである。オブジェクトの内容は従来のプログラミング言語で記述される。

(4) パネルアイコン
従来のファイルシステムにおけるディレクトリの概念と同様に、あらかじめ定義されているアイコンの集合を、ユーザがわかりやすいように１つのまとまりとして表現するアイコンである。アイコンメニュはこのアイコンによって管理されている。

(5) プログラムアイコン
アイコンプログラムを表現するアイコンである。

(6) コントロールアイコン
プログラムの制御の流れを表現するためのアイコンである。例えば、入力、出力、ループ、条件分岐などを表現するものがある。

(7) コマンドアイコン
アイコンプログラムの実行、アイコンの挿入・削除などの、システムコマンドを表現するアイコンである。

4．システム構成

システム構成を 図4 に示す。システムは Programming Execution Manager、Icon Interpreter、 Icon Image Editor、User Interface、Primitive Binder の５つのモジュールから構成される。また Library として、Primitive / Data File、Icon Image File、Icon Descriptor File の３つのファイルを持つ。

まず、Libraryとして提供されている３つのファイルについての説明を行う。

(1)Primitive / Data File
応用プログラム開発の基本要素となるプリミティブアイコンのオブジェクトならびにデータアイコンの示すデータを蓄えているファイルである。

(2)Icon Image File
アイコンとして表示される絵シンボルを蓄えているファイルである。

(3)Icon Descriptor File
アイコンに関する情報（ディスクリプタ）を蓄えているファイルである。

次に、5つのモジュールについて説明する。

(1)ＰＥＭ (Programming Execution Manager)
PEM は本システムの中心的役割を担うモジュールであり、他のモジュール(PBD, IIE, IIP)の動作はPEMによって管理されている。

(2)ＩＩＰ(Icon InterPreter)
アイコンプログラムの解釈・実行を行なうモジュールである。

(3)ＩＩＥ(Icon Image Editor)
アイコンとして画面上に表示される絵シンボルを作成するためのグラフィックエディタである。

(4)ＰＢＤ(Primitive BinDer)
従来のプログラミング言語で記述されているプログラムと絵シンボルを結び付けてプリミティブアイコンを定義するためのツールである。

(5)ＵＩＦ(User InterFace)
システムとユーザとの間のインタフェース管理モジュールである。

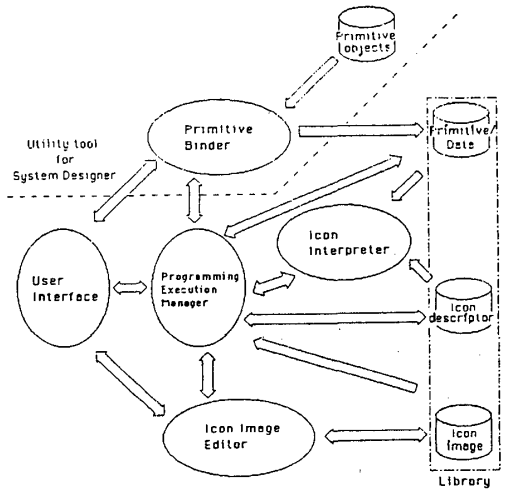

図４．システム構成

5．おわりに

本研究では、アイコンプログラムの作成・実行環境の開発を行なった。ディスプレイ上でアイコンを配置・接続してプログラムを作成し、それを直接解釈・実行させることにより、視覚的なプログラミングが可能になった。

システムは、ホストコンピュータ VAX 11/750 にワークステーション VAX station 100 を接続した構成の上にＣ言語を用いて構築されている。

参考文献

[1] E.P.Glinert and S.L.Tanimoto, "Pict : An Interactive Graphical Programming Environment," IEEE Computer, Vol.17, No.11, pp.7-25(1984).