2T−1

InterViews/Unidrawによる Domain Specific CASEの開発*

小山 徳章・五十嵐 真悟・Janet Gaboury・西村 朋子†
株式会社東芝　システム・ソフトウェア技術研究所‡

1　はじめに

近年、ソフトウェアの生産性向上を目的として、プログラム開発をオブジェクト指向プログラミング(OOP)で行なうケースが増加している。GUIを使用するDomain Specific CASE(以下、DS CASEと略す)開発においても、OOPの適用により開発効率の向上が期待される。

本稿で我々は、DS CASEにOOPを適用する際の設計指針を示し、適用例を報告する。

2　背景と問題解決の手段

DS CASEの開発では、次の2点が開発効率向上のポイントとなる。

1. Domain SpecificなGUI部品、データ構造、データ処理部分を効率良く開発する。
2. 他Domainでも再利用可能なグラフィックエディタの部分を効率良く再利用する。

我々はこのようなDS CASEの開発においてもOOPの適用が有効であると考えている。しかしながら、OOPによるDS CASEの開発には次のような問題がある。

- 既存のGUIライブラリのほとんどはオブジェクト指向言語とのインタフェースがなく、DS CASE全体をOOPする場合OOPのメリットが低下する。
- プログラマが独自のクラスを設計してしまうと、再利用可能なグラフィックエディタ部分においても再利用しにくいクラスができてしまう。

この問題を解決するためには、次のような手法をとれば良い。

- オブジェクト指向で設計され、オブジェクト指向言語で利用できる、よく洗練されたGUIクラスライブラリを利用する。
- クラスライブラリの利用手法をも含めたDS CASEの統一したクラス設計指針を示すことにより、DS CASEの再利用性を向上する。

今回我々は、GUIクラスライブラリとしてInterViews/Unidrawを利用することにした。

本稿で我々は、GUIを使用するDomain Specific CASEの開発にOOPを適用する際の設計指針を示す。

3　InterViews/Unidrawの概略

InterViews[1]はStanford大学で開発されたX Window Graphical Tool Kitであり、C++のクラスライブラリである。InterViewsで定義されているGUI部品群は、外観(appearance)とユーザの入力に対応する動作(behaviour)をもったInteractorと呼ばれるクラスにより構成されている。GUIアプリケーションの外観と動作は、Interactorを階層的に組み合わせて作成する。階層的に組み合わせるための機構として、Sceneと呼ばれるクラスが用意されている。

また、Unidraw[2]はグラフィックエディタを作成するためのクラスライブラリであり、InterViewsのクラスライブラリを組み合わせて実現されている。以下にUnidrawのクラスを2、3概説する。

Component　グラフィック部品を表現するクラスである。グラフィックの外観と性質により、種々の派生クラスが存在する。各々の派生クラスは、グラフィックの性質を表現する具体的なデータ(Graphicクラスのオブジェクト)を蓄えている。

Tool　グラフィック部品の移動、回転などの機能を、ユーザの入力をインタラクティブに受け付けながら実現するクラスである。

Command　Toolクラスとは逆に、ユーザの入力をインタラクティブには受け付けずに、ある機能を実現するクラスである。

InterViews/Unidrawのクラスを継承して作成したクラスを組み合わせて利用すれば、DS CASEをOOPによって開発することができる。

4　DS CASEのクラス設計指針

上述したInterViews/Unidrawを利用してDS CASEを作成するための、クラス設計指針を示す。

1. **基本方針**

   先に述べた問題を解決するため、InterViews/Unidrawをできるだけ効率良く利用してDomain Specific部分を盛り込むことをクラス設計の基本方針とした。

2. **クラスダイアグラム**

   図1は、DS CASEのクラスダイアグラムの概略である。このクラスダイアグラムは、InterViews/Unidrawを調査,分析して得たクラスダイアグラムに、Domain Specific部分を加味して作成した。図中、太線で示したクラスは、InterViews/Unidrawのクラスを継承し、Domain Specificな部分を付加したクラスである。

3. **Domain Specificな部分のクラス設計指針**

   Domain Specificな部分を付加したクラス(図1の太線で示したクラス)について概説する。

   GUI部品　DS CASEでユーザが編集する対象をDS CASEの外部仕様から抽出し、これらをGUI部品クラスとする。

---

*Development of Domain Specific CASE with InterViews/Unidraw
†Noriaki KOYAMA, Masato IGARASHI, Janet GABOURY, Tomoko NISHIMURA
‡Systems & Software Engineering lab.,TOSHIBA Corp.

GUI 部品クラスは Component クラスの派生クラスとし、次の 2 種類の属性を持たせる。

(a) 外観(形状) を表現する Component クラスのオブジェクト。

(b) 部品の振る舞いを表現する Domain Specific データのクラスのオブジェクト。

GUI 部品が複数ある場合は、各々の GUI 部品に対して (a) の属性を基準に、GUI 部品クラスの基底クラスとなる Component クラスを選定する。また必要ならば、(b) の属性をもとに GUI 部品を整理し、そのクラス階層を決定する。

データ　Domain Specific データを表現するデータのクラスを設計する。データのクラスはデータのもつ本質的な意味だけを捉えてクラス設計を行なう。

DS CASE の機能　DS CASE の機能は、Tool クラスと Command クラスに分類して実現する。分類の基準は、ユーザの入力をインタラクティブに受け付けるか否かである。ユーザの入力を受け付ける機能を Tool クラスとし、他方を Command クラスとする。

- Tool クラスで実現する機能は、ユーザの入力中の動作, 入力後の動作に分け、各々をクラスで実現する。前者は Manipulator クラスで実現し、後者を Command クラスで実現する。Tool クラスは、これら 2 つのクラスを操作するように設計する。
- Command クラスは GUI 部品を統括、操作して、ある機能を実現するように設計する。GUI 部品に隠蔽されているデータ操作の処理部分は、個々の GUI 部品に実行させるように設計する。

クラス設計手順を以下に示す。

1. GUI 部品を DS CASE の外部仕様から抽出し、GUI 部品のクラス階層と基底クラスを決定する。
2. Tool クラス, Command クラスを DS CASE の外部仕様から抽出し、必要な Manipulator クラスを設計する。
3. Tool クラス, Command クラスの抽出と並行して DS CASE の外観を設計する。
4. データのクラスを設計する。
5. GUI 部品クラスの private 変数とメソッドを定める。
6. 新たに作成する Command クラスに対応する GUI 部品内部データの処理部分を、個々の GUI 部品メソッドに記述する。

以上の指針および手順を実際に DS CASE 開発に適用した。今回は、例として “シーケンス制御ブロック図エディタ (以下、エディタと略す)” を取り上げ、分析, 設計を行った。このエディタは、シーケンス制御ブロック部品を配置、配線してシーケンス制御ブロック図を作画し、この図から専用の制御ソースプログラムを生成する DS CASE である。

まずシーケンス制御ブロック図を表現する 65 種類の GUI 部品を外部仕様から抽出し、これらの部品を抽象的に表現する基底クラスを設計した。部品の外観は、直線、折れ線、長方形、テキストの組み合わせで表現できたため、部品は Unidraw の GraphicComps クラスの派生クラスにした。

つぎに、部品の “入出力” という性質に注目し、この基底クラスから 5 つの派生クラスを作成し、さらにその派生クラスとして個々の部品クラスを設計した。また、部品に持たせる DS CASE 固有データのためのデータのクラスを作成し、各部品にはそのオブジェクトを list、あるいは array の形で持たせた。

Tool クラスとして “部品間の接続” などの機能を、Command クラスとして “専用言語ソース生成” などの機能を外部仕様より抽出し、新たに作成する Tool クラスに対しては、各々Manipulator クラスおよび Command クラスを作成した。

このようにクラスを設計した結果、InterViews/ Unidraw を効率良く使い、OOP による DS CASE の開発を迅速に行なうことができた。

5　おわりに

本稿では、オブジェクト指向 GUI クラスライブラリを効率良く利用して、GUI を使用する Domain Specific CASE を開発するための、DS CASE のクラス設計指針を示した。

本稿で示した設計指針に従って DS CASE を設計すれば、クラスライブラリを効率よく使うことができ、オブジェクト指向の利点を十分活かした開発が行なえ、DS CASE の開発効率を向上させることができると考えられる。

今後、本稿で示した設計指針を多くの Domain Specific CASE 開発に適用し、より詳細な設計指針を示していきたい。

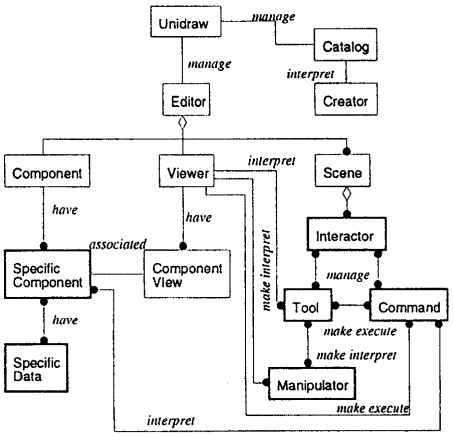

図 1: クラスダイアグラムの概略 (OMT[3]法による)

参考文献

[1] M. A. Linton et.al.: InterViews: A C++ graphical interface toolkit. CSL-TR-88-358, Stanford Univ,Jul.'88

[2] J. M. Vlissides et.al.: Applying object-oriented design to structured graphics. *Proceedings of the 1988 USENIX C++ Conference*, pp.81-94, Oct.'88

[3] J. Rumbaugh et.al.: Object-Oriented Modeling and Design. Prentice Hall, 1991