Vol. 35 No. 5　　情　報　処　理　　May 1994

解　説

オブジェクト指向分析・設計

6. GUI クラスライブラリ構築への適用†

井　上　健††

1. はじめに

OMT 法を適用して，大規模な GUI ライブラリの構築を行った経験を報告する．

OMT 法は GE 中央研究所（以下 GECRD）のランボーらによる提案であるが，GECRD では，この手法にもとづいて，開発支援を行うソフトウェアツール OMTool を開発した．当社では OMTool の開発初期より評価およびエンハンス提案を行い，C++ 対応版の開発にも参加した．さらにそれと同時進行の形で，グラフィックエディタ開発支援ツールキットである GIST (Graphical Interactive Systems Toolkit) の開発を行った．

ここでは，オブジェクト指向および OMT 法を GIST 開発に適用した報告と，OMTool のような CASE ツールの利用に関する議論を行う．

2. C++ OMTool

オブジェクト指向 CASE ツールのひとつである OMTool は，OMT 法によるシステム開発を支援するもので，オブジェクトモデル，動的モデル，機能モデルの三つを対話的にスクリーン上で構築することができる．OMTool によって設計されるクラス階層図の例を**図-1** に示す．

このツールのひとつの特徴は，作成したオブジェクトモデルから C++ のソースコードを出力することである．クラス定義，メソッドのインタフェースなどに変更があった場合，先に生成したソースを読み込んでから再出力するため，OMTool 上での変更を実装途中のソースに自動反映することができる．また，emacs エディタとの通信機能ももち，マウスによってクラスボックス内のメソッドを選択してから所定のコマンドを発すると，エディタ内でそのメソッド定義のところへ自動的にカーソルが飛ぶ．これはソースのサイズが大きくなってきたときに便利な機能である．

3. GIST の紹介

GIST は GUI アプリケーション構築用の C++ クラスライブラリであり，応用分野としては，幾何図形を使ったインタフェースビルダ，チャートエディタを始め，シミュレータやテスタ，その他種々のグラフィカルでインタラクティブなアプリケーションがあげられる．

ここでは，OMT 法の適用を報告するための例として，構成の面から GIST を紹介する．機能的な特徴については参考文献 2)を参考にされたい．

GIST のクラスの全体像を**図-2** に示すが，構成の特徴は以下のとおり．

- Subject-View-Image モデル

アプリケーションが使用するデータを Subject，表現を Image とし，それを View クラスが連結する方式をとった．これによりデータを多様な形式で多様な出力に一貫した形式を保って表示できる．

- 図形オブジェクトの構造化による表示管理

Image クラスでグラフィック木を管理し，この木を使って表示オブジェクトの表示/隠蔽などが効率よく確実に行える．

- 図形間の意味的な関連が記述可能

図形オブジェクトどうしが連結している場合，ひとつの図形の変化に対して自動的に他オブジェクトへの影響を計算し，管理することができる．これは，図形オブジェクトのイメージを，Pad や Joint というオブジェクトを使って連結する方法をとることによって実現している．

- EIL (Event Interpretation Language) の導入

† A Report of OOA/OOD for a GUI Class Library Construction by Takeshi INOUE (Yokogawa Electric Corporation Open Systems Laboratory).

†† 横河電機(株)オープンシステム研究所

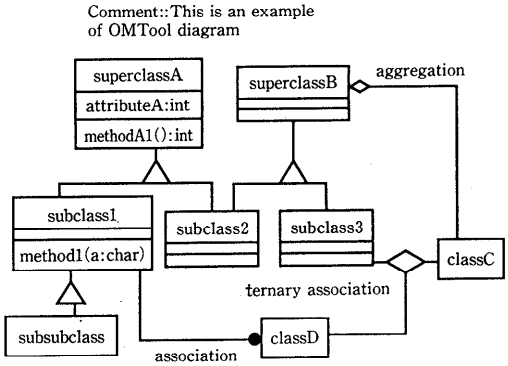


図-1　OMTool によるオブジェクトモデル表示例

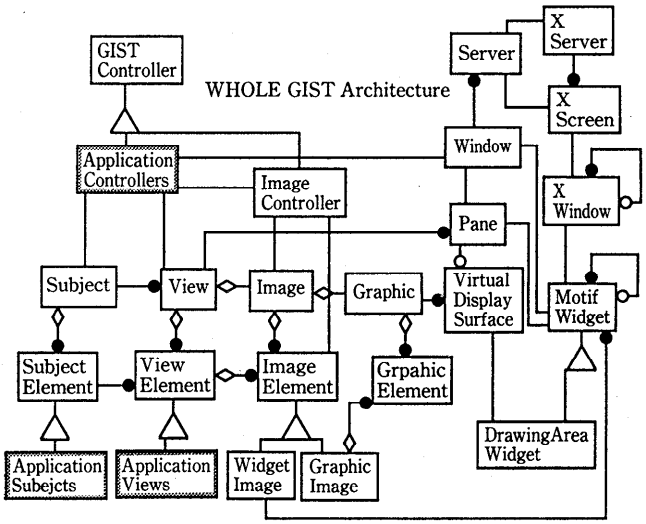


図-2　GIST の全体構成

ユーザインタフェースやアプリケーションの振舞いは，イベントをトリガとする状態遷移を考え，EIL 言語を用いて記述できる．

- X ウィンドウ/Motif とのインタフェース

GIST のクラス群を出力メディア独立の形式で実現し，X ウィンドウや Motif とのインタフェースをもつクラスをそれらとは別に用意して表示する．

GIST の規模はクラス数が約 200，ソースが約 10 万行である．4 人のエンジニアにより約 1 年半の期間で作り上げた．

図-2 において，暗色のクラスは GIST のユーザが開発する部分である．GIST を利用して GUI アプリケーションを構築する場合は，以下のようになる．

1. SubjectElement のサブクラスを作って，ドメインで扱うデータを格納する．
2. ViewElement のサブクラスを作って，データの図形表現を定義する．
3. Controller のサブクラスを作って，アプリケーション内外からの刺激に対する振舞いを規定する．

GIST は，刺激に対して GIST イベントとよばれるイベントオブジェクトを発生させるメカニズムをもっており，GIST イベントが Controller オブジェクトによって解釈され，Controller の状態に応じた処理を行う．この処理を定義するために，さきに述べた EIL 言語が用いられる．

4. OMT 法を用いた GIST の開発過程

要求の分析から設計まで約半年を要した．ここでは，オブジェクト指向による分析/設計の過程での特徴的なことがら，評価に重点をおきながら述べる．

分析と設計はどちらも同じメンバで作業したため，明確な分離，区別は意識しなかった．オブジェクトとオブジェクト間の関係の大まかな決定までが分析であり，そのあとの，各オブジェクトの詳細設計にはいるところから設計フェーズになるのであるが，この境界はあいまいで，行きつ戻りつ考え直す作業が頻繁におきた．このような作業は実装の段階も含んで行われている．これがオブジェクト指向による開発の大きな特徴のひとつである．

4.1 開発の進め方

設計は以下の方式で行われた．まずプロジェクトリーダのオフィスに 4 人が集まり，オブジェクトモデルやシナリオを話し合ったり，黒板を使ってデータの流れなどを検討した．オブジェクトモデルが検討されてゆく過程でリーダがワークステーション上で OMTool を使ってクラスの階層や定義を決めてゆき，クラス間の関係やクラス内部の定義もどんどん付加していった．

われわれの分析，設計の流れをおおまかにまとめると以下のようになる．

1.「まず」何をオブジェクトにするのか，主役をつとめるいくつかのオブジェクトを考え，クラスボックスを作る．

2. 主役たちの関連，結び付きを考え，OMT 法にもとづく表記法による線で結ぶ．(関係するオブジェクトの多重度 (multiplicity) はまだ考えない)

3. 主役たちを管理/操作するオブジェクトが必要になったら適宜定義する．

4. 関連そのものに単純でない役割があると分かった時点で，関連オブジェクトまたは，独立したオブジェクトに昇格させ，それらを主役たちと関連づける．

5. 各オブジェクト自身について考え，属性，操作，サブクラス化などを定義する．この間に随時，関係の multiplicity などもつめて定義してゆく．

GIST の分析/設計は図-2 の，Subject, View, Image, Graphic のレイヤを先に行い，大まかなレイヤ分けをした後に，Graphic レイヤの図形表示のための詳細な設計が始まった．このレイヤが決まると，順々に Image, View, Subject と詳細を議論し，表示のためのインタフェースオブジェクトを決め，それからコントローラでこれらのオブジェクトを扱うしくみを議論した．GIST の設計に関する限り，表示オブジェクトから考え，それから，それにつながるオブジェクトを考え，最後に振舞いのコントロール，という順に設計することが，能率的に考えやすかった．

4.2 OMT 法による開発の特徴

頭の中にあるオブジェクトの概念をオブジェクトモデルで表現することは，われわれにとって非常に自然で有効に行える作業であった．OMT 法であげている関連 (association) はわれわれにとってより意味深く，初期の段階で定義されることによって，設計が手際よく行われたことを強調したい．これは，オブジェクトの属性（メンバ）として他のオブジェクトへのポインタをもたせるのではなく，関連として線で二つのオブジェクトを結ぶほうがより実世界のオブジェクトの関係を表現し，さらに図になったときに分かりやすいという効果による．OMT 法でいう汎化 (generalization)，集約 (aggregation) は設計が進むにつれて多用したが，設計の流れの印象から述べれば，集約を定義したあとに汎化を考えつめてゆくのが有効な方法に感じられた．

関連と集約は明確に区別しにくい場面もあったが，まず関連として定義し，のちに二つのオブジェクトの包含関係や主従関係を再度考えながら集約関係になおすことによって，定義していった．

GIST の開発はほとんどオブジェクトモデルのみを使用し，動的モデル，機能モデルなどは適宜モデルの基本的な考えを利用することにとどまった．われわれの場合，オブジェクトモデルが決まってから必要に応じてそれらのモデルに近いものを使い分けることが自然にできていた場合もあった．すなわち，場合場合によって振舞いの異なるオブジェクトに対しては状態遷移を考えながら進めると，「異なる状態」というものを構造化された枠組でとらえることができたし，Subject-View-Image の実現は適宜データフローモデルやイベントトレース図などを使いながら議論を行った．

OMT モデルの適用がしにくかったものの一つとしては，図形間の関連を保持するためのしくみの考察があげられる．GIST では図形どうしを連結するために，Pad および Joint とよばれるオブジェクトを使用しており[2]，ひとつの図形が移動しようとしたとき，その図形に付着している Pad にその動きを伝え，それが Joint オブジェクトを介して他の図形オブジェクトの Pad に動きを伝えて計算するしくみになっている．このしくみの設計では，同種クラスに属する多数のオブジェクトどうしの通信を，そのまま OMT 法のモデルやイベントトレース図では表現しにくく，簡略化を適宜行って，動きや流れを考えた．

5. GUI へのオブジェクト指向の適用について

GIST の開発を通して得た経験をもとに，GUI 構築における，OOA/OOD, CASE ツールなどの有用性，適合性などについて考える．また，GIST を GUI のライブラリとして提供し，GUI アプリケーションを開発した経験に基づく考察も行う．

5.1 オブジェクト指向分析／設計

GUI とオブジェクト指向プログラミングの親和性については，その有用性が広く認められており，ここで議論しない．われわれのプロジェクトで OOA/OOD を利用することで非常に役に立っ

たことの一つは，「自分の考えを人に伝えやすい」ということであった．

開発者全員がOMT法というモデル化の記法を熟知し，その表現法を媒体として頭の中にある，クラスの階層，オブジェクト間の関係，役割を議論しあうことは効率的であったと思う．オブジェクト指向というパラダイムだけでなく，思考をうまく表現する方法があるということは，自分自身の考えを進めやすいばかりでなく，グループ開発の上でも有用である．

もうひとつの特徴は，分析から設計，実装と開発フェーズが進んでも，フェーズ間のずれや壁がないいわゆるシームレスなことである．分析したものを詳細化しながら設計し，そこでできたものをそのまま実装したのがGISTであり，下位フェーズの要求を上位に反映させることなども容易であった．フェース間の情報のずれも起きにくい．しかしながらこれは，開発履歴をしっかりとっていないと，初期の分析結果を調べたり，戻って考えたりできにくくなることもある．過去の分析/設計履歴を保存するしくみが支援CASEツールには必要であろうし，明確なフェーズ分けは，プロジェクト管理者が意識的に行う必要がある．

オブジェクト指向の分析/設計法の存在意義というのは，それを用いることにより，効率的に信頼性の高いシステムを生産することであると考えられる．この存在意義を認識した上で，開発するシステムにどう手法を生かすかを考えてゆくことが重要な点である．各種の手法ではモデルの考え方，作り方のノウハウまでことこまかに解説されているが，実際のプロジェクトへの完全な適用は難しく，**4.2** に触れたように，GISTでもモデルを適用しにくい場面がいくつかあった．このような場合には，問題に即してモデルをカスタマイズするか，適応可能な手法を探すなどの試みが必要になる．

また，GISTの場合は，その実現手法に関して実験的，あるいは研究的要素を含んでおり，OMTモデルの記述に関してあまりに忠実に手法に従うと，工数がかかることはもちろんだが，自由な発想をときに妨げる場合もあるのではないかと感じていた．自由なプロトタイピングにより模索しながら考えるような開発の場合にも，モデルを使って検討することは必要であるが，まず方法論を利用するポリシーを考えた上で開発を進めるべきであろう．

5.2 オブジェクト指向 CASE ツール

今回われわれが利用したOMToolは，OMT法のすべてのモデルを完全にサポートしていない，プロジェクト管理のためには貧弱，グループウェア/ネットワークサポートができていない，など，まだまだ未完成の部分も多いが，ひとつの大きなソフトウェアを作るために大きく役だった．オブジェクトモデルを考えそれを表現するには，紙と鉛筆よりも，あるいは汎用の図形エディタよりも，思考を妨げられず修正/変更しやすい，クラスの管理がしやすいなど，利点は多い．

また，OMT法などのオブジェクト指向の分析/設計のための方法論は，上流フェーズにおける有用性が取り上げられることが多いが，われわれは，C++ ソースを生成する機能にも多大な恩恵を受けている．画面上のオブジェクトモデルで，見たいメソッド名を選択してコマンドを選ぶと，エディタ内でそのメソッドが表示されるのは，非常に便利で，ソースの変更も行いやすかった．また，あるクラスのオブジェクトを利用する場合に，ソースを見に行くのではなく，オブジェクトモデル図を見ることによって，メソッドのインタフェースが分かるため，クラスの利用者にも有効である．OMToolではヘッダファイルにある情報がすべてオブジェクトモデル図上にあるため，それに慣れてしまうとC++ のヘッダファイルを見る機会も少なくなった．実際，GISTはOMToolとともに管理され，OMToolとともに社内ユーザに提供している．

OMToolを例に話しを進めてきたが，一般的にこのような実装/保守フェーズまでの本格的な利用に耐え得るCASEツールに要求される機能を考えると，以下のようになる．

- 手法に準拠しており，その方法論を学ぶこと/学んだあとの実践共に最適
- レスポンスよく，なじみやすいユーザインタフェース
- 日本語処理は分析/設計作業には必須
- 自由なモジュール分け，モジュール統合
- プログラミング言語への厳密な対応
- ソースコードの自動生成，モデル上の変更のソースへの自動反映

- ヴァージョン管理，履歴機能
- ネットワーク対応，排他制御などによるグループウェアサポート
- 既存ソースからオブジェクトモデルなどを自動作画する，いわゆるリバースエンジニアリング機能

また，OMT 法の機能モデルや動的モデルなども忠実に表現できるツールでは，仕様記述と実装の一致という意味から，実装にもそれらの結果が反映できることが望ましい．

5.3 GUI ライブラリとしての GIST の利用

当社のシステム技術部では，GIST を利用してプラント制御システムのための，プラント監視画面構築用インタフェースビルダ 3 種類を開発した．これらはやはり 100 以上のクラスからなるシステムであり，この経過も「GUI への適用」という観点から簡単に紹介する．ビルダの開発者は，オブジェクト指向や C++ は初めてのエンジニアばかりであり，GIST 開発者からのコンサルティングが必要であったが，

- OMTool の利用により，オブジェクト指向の概念に馴染みやすい
- コンサルティングの際に OMTool を利用してオブジェクトモデルを説明すると，GIST のクラス階層や各オブジェクトの役割が伝えやすいなどの点で OMT 法および OMTool の利用が有効であった．

GIST ライブラリを利用するには，ライブラリが用意するオブジェクトの挙動をある程度理解しないといけない．われわれが GIST 利用者にソースファイル以外で提供したものは，文章で書かれたドキュメントとオブジェクトモデル図であり，決して十分なものとはいえなかった．挙動のシナリオが伝えられるような動的モデルに相当するものを用意することは，分析/設計のためだけでなく，オブジェクトの保守/再利用の観点からも必要であると感じている．

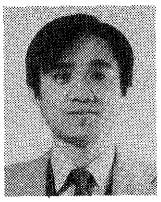
6. ま と め

GIST および，GIST を利用したアプリケーション開発の経験から，OMT 法および OMT 準拠の CASE ツールの評価を報告した．今後 GIST を使ったアプリケーションには，OMT 法や CASE ツールをさらに多用することになるので，より深く方法論およびツールを評価して，改良してゆきたい．

参 考 文 献

1) ランボー, J. 他：オブジェクト指向方法論 OMT. トッパン (1992).
2) 井上，土田，村田，渡辺：グラフィカルエディタ作成用ツールキット GIST，情報処理学会第 47 回全国大会 5-125, 127, 129 (1993).
3) ビンソン，L. J.，ウィナー，R. S.：Smalltalk：オブジェクト指向プログラミング．トッパン (1990).
4) Linton, M. A. et al.: Composing User Interface with InterViews: Stanford University より FTP で入手 (1988).
5) Coad, P. and Yourdon, E.: Object-Oriented Analysis—second edition: Prentice Hall (1991).
6) 井上，土田，村田，西岡：C++ OMTool の大規模ソフトウェア開発への適用事例，情報処理学会研究報告 93-SF-92, pp. 27-32 (1993).

(平成 6 年 3 月 2 日受付)



井上　健

1958 年生．1981 年東京工業大学電気電子工学科卒業．1983 年同大学院総合理工学研究科修士課程修了．工学修士．1983 年横河電機 (株) 入社．1989年 MIT の Project Athena に参加し，ユーザインタフェースの研究に従事．1990 年より 2 年間ゼネラルエレクトリック社中央研究所に出向し，GUI ライブラリの構築に携わる．現在，横河電機オープンシステム研究所主任研究員．日本ソフトウェア科学会会員．