使いやすいソフトウェアと作りやすいソフトウェア
——オブジェクト指向概念とその応用——

解　説

正員 中所武司 (株)日立製作所システム開発研究所

キーワード：オブジェクト指向，ソフトウェア生産技術，言語，ユーザインタフェース，知識表現，データベース，オブジェクト管理

1. まえがき

激動する今の時代を象徴するキーワードとして，グローバル化とパーソナル化をあげることができる。グローバル化は，経済活動の24時間化と現地化という意味を超えて，あらゆる分野で進行している。東からは日米構造協議による日本的経済機構の排除，西からは欧州統合とソビエトのペレストロイカに代表される東欧諸国の急激な社会変革，南からは難民流出による国境の無意味化，という形でグローバル化の波が押し寄せている。これらはすべて社会システムのオープン化と標準化を促す動きである。一方，パーソナル化は物のレベルで進行している。家庭におけるテレビや自動車の一人1台化，オフィスにおける電話やコンピュータの一人1台化は，これらの物の使い勝手や感性を重視した個性化を促している。

このような「システムのグローバル化」と「物のパーソナル化」の動きは，計算機ソフトウェアの世界も例外ではない。種々の応用ソフトがどんなハードの上でも使えるようにするためのオープンソフトウェアアーキテクチャが重要視され，基本ソフトウェアのグローバル化を目指した標準化活動が活発に展開されている。一方，計算機のパーソナル化の代表であるパソコン，ワークステーションの分野では，使い勝手の良いグラフィカルユーザインタフェースの実現が必須になっている。

このような背景のもとに，最近，オブジェクト指向概念[1]がソフトウェアの種々の分野に応用され始めている。先に述べたグラフィカルユーザインタフェースや基本ソフトウェアのほかに，マルチメディアデータベースやネットワークの分野への応用が始まっている。プログラミング言語やソフトウェア工学あるいは知識工学の分野では既に以前から実用化が進んでいる。

本解説では，オブジェクト指向概念を「使いやすいソフトウェア」と「作りやすいソフトウェア」を促すソフト生産技術の観点から位置付け，その応用について述べる。

2. オブジェクト指向概念とは

2・1 概　要

概念的対象物をそのままプログラムの基本単位として表現できるオブジェクト指向概念は以下のような特徴（図1）を有し，情報処理や知識処理関連の多くの技術分野で広く利用されている[2]。

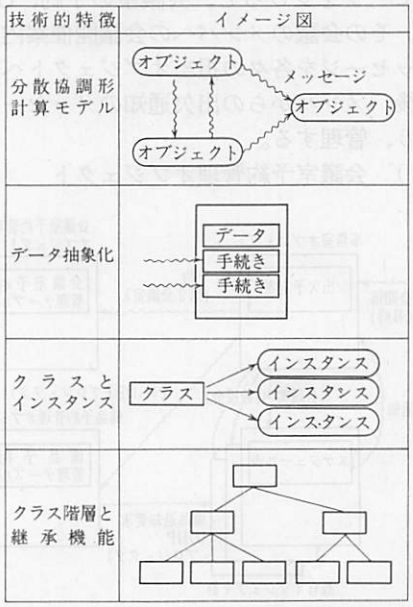


図1　オブジェクト指向概念の特徴

（1） 分散協調形計算モデル　　複数個の自律的機能をもつオブジェクトが互いにメッセージをやり取りしながら協調して問題解決にあたる。

（2） データ抽象化機能　　個々のオブジェクトはデータと手続き（メソッド）を一体化（カプセル化）した自律的機能モジュールとして定義され，外部からのメッセージを受け取ると手続きが起動される。データに対しては外部から直接アクセスはできない。

（3） クラスからのインスタンス生成機能

データの値が未定のオブジェクトを型(クラス)として定義し，機能が同じでデータの値だけが異なる複数個のオブジェクト（インスタンス）を効率良く生成できる。

（4） クラスの階層化と継承機能　　上位クラスのもつ手続きを下位クラスの手続きとして継承できるようなクラス階層構造により，少しづつ機能の異なるオブジェクトを効率良く作成できる。

以下では，これらの特徴について具体的に例を用いて説明する[3]。

2・2 分散協調形計算モデル

今，大学での会議開催の事務処理を分散協調形計算モデルとして表現した例を図2に示す。以下の4種類の自律的機能をもつオブジェクトが導入され，互いにメッセージをやり取りしながら協調して問題解決にあたっている。

（1） 事務室オブジェクト　　会議開催の要求メッセージを受けると，会議室の予約，OHPの予約，その会議のメンバへの会議開催案内のためのメッセージを各々の担当オブジェクトへ送る。その後，メンバからの出欠通知のメッセージを受け取り，管理する。

（2） 会議室予約管理オブジェクト　　会議室予約の要求メッセージを受け取り，予約管理をする。

（3） 備品予約管理オブジェクト　　備品予約の要求メッセージを受け取り，予約管理する。貸出期限をすぎて未返却の備品の返却要求メッセージを貸出先へ送る。

（4） 教官オブジェクト　　会議開催案内のメッセージを受け取り，出欠通知のメッセージを送る。

このようにオブジェクト指向プログラミングでは，実世界の問題解決方法に近い計算モデルを用いて自然な表現が可能である。特に，タスクやモジュールをプログラム構成要素としてモデル化をする従来方式のように，プログラム設計の最初の段階から処理制御の順序や共通データの導入を考える必要がないので，「どのように」(how) ではなく，「何を」(what) 主体の表現ができる。

2・3 データ抽象化

先の図2の会議室予約管理オブジェクトの記述方法について考える。従来の手続き型言語を用いたプログラミングでは，図3に示すように予約管理テーブルを共通データとして，プログラムの任意の場所からアクセス可能とする方式が一般的である。

これに対し，オブジェクト指向プログラミングでは，図4に示すようにデータとその手続き群を一体化したオブジェクトとして定義する。これらの手続きはメソッドと呼ばれ，外部から見たオブジェクトの機能は，このメソッドの集合で表現さ

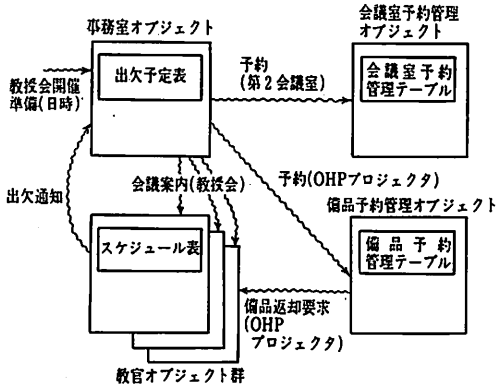


図2 会議開催事務処理の分散協調形計算モデル

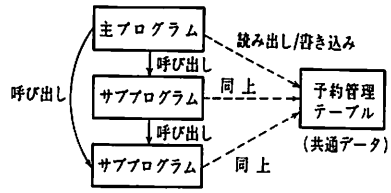


図3 共通データアクセス方式を用いた従来の処理モデル

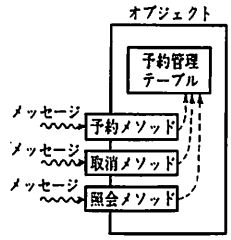


図4 予約管理機能を実現するオブジェクト

れる。データそのものは外部からは隠ぺいされ，直接アクセスすることはできない。図2の例では，会議室予約管理オブジェクトの会議室予約管理テーブルのほかに，備品予約管理オブジェクトの備品予約管理テーブル，事務局オブジェクトの出欠予定表，教官オブジェクトのスケジュール表などが外部から隠ぺいされている。

このようなプログラミング方式では，オブジェクト内のデータの変更やメソッドの処理手順の変更が生じても，メソッドの機能が不変ならば，まわりのオブジェクトは変更しなくてすむため，プログラムの保守性が高くなる。

2・4 インスタンス生成

図2の会議室予約管理オブジェクトと備品予約管理オブジェクトは，予約管理テーブルの内容が異なる以外は図4のような同じ構造のオブジェクトと考えて良い。このように，メソッドが同じで，データの内容だけが異なるオブジェクトを個別に作成する必要はなく，図5に示すように，その型となるオブジェクトを複写して，インスタンスオブジェクトを生成すれば良い。この型となるものをクラスオブジェクトと呼ぶ。

2・5 クラス階層と継承機能

図4の予約管理オブジェクトは，予約，取消，照会の機能をもつが，今，新たに優先予約機能を

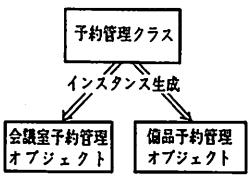


図5 クラスオブジェクトからのインスタンスオブジェクト生成の例

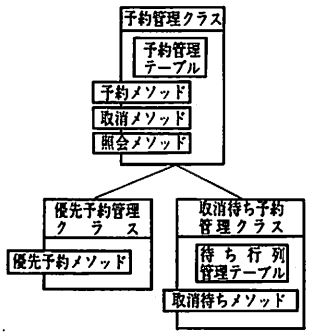


図6 継承機能をもつクラス階層の例

もつ優先予約管理オブジェクトを導入する場合を考えよう。三つの機能をもつ予約管理オブジェクトとは別に，四つの機能をもつ優先予約管理オブジェクトを定義してもよいが，面倒である。そこで，その代りに，図6に示すように，予約管理オブジェクトの下にその子供のクラスとして優先予約管理オブジェクトを位置付け，親クラスがもっていない優先予約機能だけを定義する。そして，他の三つの予約，取消，照会機能は親クラスから継承する。図6のもう一つの例の取消待ち予約管理オブジェクトについても同様である。

3. 簡単操作と簡単プログラミング

このようなオブジェクト指向概念の特徴は，ソフトウェア生産技術の観点から以下のような利点をもつ。

3・1 利用者のための操作性と統一性

パーソナルコンピュータやワークステーションの高機能化と低廉化に伴い，オフィス業務やエンジニアリング業務への計算機利用が急速に進んでおり，計算機の利用者の大半は，従来のような情報処理の専門家ではなくなってきている。このような意味での計算機のパーソナル化に対応して，パーソナルコンピュータやワークステーションのソフトウェアは，誰でも簡単に使えるユーザインタフェースを提供することが必須になっている。

オブジェクト指向概念は，以下のようなユーザインタフェースの操作性と統一性を実現する。

（1） 画面上にみえている各々のオブジェクトがもっている幾つかの機能の一つを選ぶという操作，すなわち，オブジェクトにメッセージを送るという単一的操作で簡単に計算機と対話が可能である。

（2） 種々の応用ソフトウェアのユーザインタフェースを共通部品オブジェクトで構築することにより，異なる応用ソフトウェア間でユーザインタフェースを統一できる。

例えば，図4の予約管理オブジェクトを画面上で選択すると，予約，取消，照会の選択メニューが表示され，その中から照会を選ぶと予約状況が表示される，というようなシステムを簡単に作れる。

3・2 開発者のための部品化，再利用

ソフトウェアを簡単に作る方法として部品化，再利用技術は古くから研究されてきたが，これまで必ずしも実用技術として成功したとはいえな

い。その原因としては，従来のプログラミング方法に基づく手続き的モジュールやサブルーチンでは部品としての機能の独立性が弱く，汎用部品ライブラリーの構築が容易ではなかったことが考えられる。

これに対し，オブジェクト指向概念は次のような点で部品化，再利用に適している。

（1） データ抽象化機能により，独立性の強い，汎用的な機能部品を作りやすい。

（2） 階層化機能と継承機能により，部品ライブラリーを構築しやすい。

（3） インスタンス生成機能により，部品のカストマイズがしやすい。

（4） 分散協調形モデルに基づいてプログラム設計を行うことにより，部品（オブジェクト）の抽出，利用が容易である。

典型的な例としては，Smalltalk-80（Xerox社）やObjective-C（Stepstone社)が提供するクラスライブラリーがある。Smalltalk-80では約5,000のメソッドを含む250のクラス，Objective-Cでは約300のメソッドを含む20のクラスが定義されている。特にObjective-Cでは，これらのクラスをハードウェアにおけるIC部品のように使うという意味で“ソフトウェアIC”と呼ぶ[4]。

3・3 保守者のための移行性と拡張性

情報化社会のインフラストラクチャとして，複雑化，大規模化している応用ソフトウェアに関して，ソフトウェア生産性の低さによる開発コストの増大，開発要員不足によるバックログの増大などの問題が発生している。そのため，このような高価な応用ソフトウェアをできるだけ長い期間使用するための拡張性と，できるだけ広範囲で使用するための移行性が重要な課題である。

すなわち，昔，ハードウェアが非常に高価であった時代には，

「一つのハードの上で沢山のソフトが動く」

ことが重要だった。しかし，ハードウェアが急激に低廉化する一方で，ソフトウェアの開発コストが膨大になっている現在では，

「一つのソフトが沢山のハードの上で動く」

ことが重要になってきた。そのために，最近では，ソフトウェアのアーキテクチャを以下のような階層構成にして，できるだけ標準的に作る努力がなされている。

（1） ハードウェア，基本ソフトウェア，応用ソフトウェアを完全な階層構造にする。

（2） 各階層を部品の集合で構成し，統一的なインタフェースとする。

オブジェクト指向概念はこのようなソフトウェア構成に適しており，以下のように拡張性と移行性を実現している。

① 階層化機能と継承機能により，機能追加が容易である。

② オブジェクトの機能独立性が強く，外部インタフェースと内部構造が完全に分離しているため，機能変更や他機種への移行が容易である。

4. オブジェクト指向概念の適用例

4・1 ユーザインタフェースおよびその構築ツール

（1） ユーザインタフェース　コンピュータのパーソナル化に対応して，だれでも簡単に使えるソフトウェアを実現するために，ユーザインタフェースは非常に重要になっている。オブジェクト指向概念は，使いやすいユーザインタフェースのための三つの特徴を実現する。

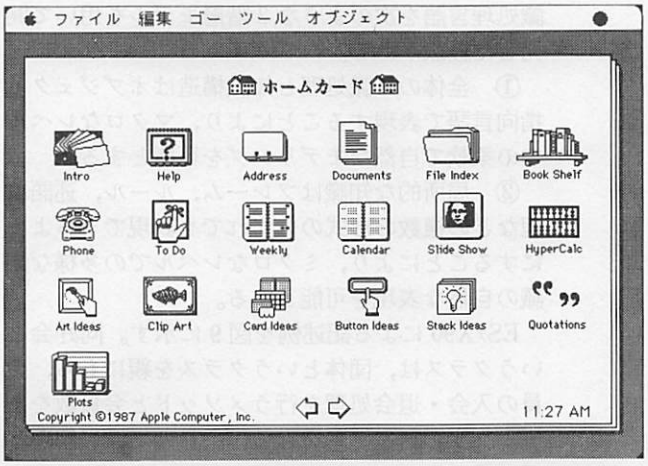
（i） メタファ（暗喩）　オフィスシステムにおける書類，フォルダ，キャビネ，あるいは制御システムにおけるパラメータ設定盤などに対応するアイコンをビジュアルオブジェクトとして画面に表示し，操作の対象をわかりやすくする。

（ii） 直接操作　このビジュアルオブジェクトにデータやメソッド（操作手続き）をもたせ，画面上で「操作対象のビジュアルオブジェクトを指定し，必要なメソッドを選ぶ」ので，操作が直感的でわかりやすい。

（iii） 統一性　画面上の表示物をすべてオブジェクトとして作成し，オブジェクトにメッセージを送るという単一操作で簡単にコンピュータと対話ができる。

このようなグラフィカルユーザインタフェース（GUI）を実現した先駆的システムとして，Apple社のハイパメディアシステムHyperCardがある。最近では，HP社のNewWave，OSFのMotif，UIのOpenLook，Next社のNextStepなどがある[5]。図7にHyperCard（日本語版）の初期画面を示す。画面上のアイコンの一つを選択すると，それに対応するスタックと呼ばれるドキュメントが使用可能になる。

（2） ユーザインタフェース構築ツール　個々の応用ソフトウェアのユーザインタフェース



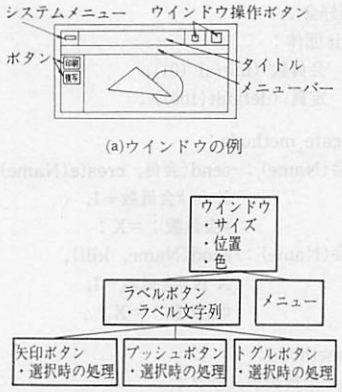
〔アップルコンピュータジャパン（(c) 1987）の許可を得て掲載〕

図 7　HyperCard（日本語版）の初期画面

は，各々の中で一貫性があるだけでなく，異なる応用ソフトウェアの間でも統一がとれている方が利用者の使い勝手が良い。そのためには，開発者にとってもユーザインタフェースが作りやすくなっていることが重要である。そこで，先にあげたOSF/Motif や NextStep では，ユーザインタフェース構築ツールを提供している。

OSF/Motif では，スクロールバーやボタンなどのウィンドウの構成部品を widget と呼ぶオブジェクトとして定義した部品を提供している。その使用例を図 8 に示す。図の(a)は Motif のウインドウの例である。図の(b)は，矢印ボタン，プッシュボタン，トグルボタンの 3 種類のボタンのクラスがウインドウおよびその構成要素に関するクラス階層の中で定義されていることを示す。ウインドウクラスは，位置，サイズ，色のデータと，それを変更する手続きをもつ。ウインドウクラスのサブクラスは，これらのデータと手続きを継承するため，親クラスがもっていないデータと手続きのみを定義すればよい。そして，例えば，「印刷」を指示するプッシュボタンを作る場合，プッシュボタンクラスのインスタンスを生成し，以下のデータに値を設定するだけでよい。

インスタンス名：“印刷ボタン”
サイズ　　：100 * 50
位置　　：20，60
色　　：青
ラベル文字列　：“印刷”
選択時の処理　：プリント



(a)ウインドウの例

(b)クラス階層に基づくウインドウ構成要素の定義

```
XtSetArg ( args [0], XmNarrowDirection ,
    XmARROW__RIGHT);
arrow=  XmCreateArrowButton (parent,
    "Arrow 1", args, 1 ) ;
XtAddCallback ( arrow, XmNactivateCallback,
    click__proc, NULL ) ;
XtManageWidget ( arrow );
```

(c)矢印ボタンの生成プログラム

〔OSF（(c) 1989）の許可を得て掲載〕

図 8　OSF/Motif によるユーザインタフェース記述例

図の(c)は矢印ボタンのクラスからインスタンスを生成するプログラム[6]の例である。1 番目の文は，矢印ボタンのインスタンス生成のための引数設定である。第一引数は引数の設定エリア，第二引数は矢印の向きを指定すること，第三引数は矢印の向きが右であることを指定している。2 番目の文は，矢印ボタンのインスタンス生成関数の呼び出しである。第一引数は画面上で親となるウインドウ構成要素，第二引数は生成されるインスタンスの名前，第三引数は引数の設定エリア，第四引数は引数の数を指定している。3 番目の文は，このインスタンスがマウスで選択されたときに呼びだされる手続きが click_proc であることを指定している。4 番目の文は，このインスタンスの表示をウインドウマネージャに依頼している。

NextStep のインタフェースビルダは，画面上のビジュアルオブジェクトのセットの中から必要なものをコピーして，位置や大きさを自由に変えたり，関係するものを線で結ぶという方法で，ユ

```
class 同好会；
  inherit 団体；
  slots  会員数 (initial (0)),
         定員 (default(100))；

  predicate_methods；
    入会(Name)：-send(会員, create(Name)),
                 X is #会員数+1,
                 #会員数：=X；
    退会(Name)：-send(Name, kill),
                 X is #会員数−1,
                 #会員数：=X；

  rule_methods；
    rules 会員数チェック；
      if #会員数=0 then write('会員未登録です')；
      if #会員数>=#定員 then write('満員です')；

end.
```

図9　オブジェクト指向ベースのマルチパラダイム型知識表現言語 ES/X 90 の例

ーザに見せる画面を直接作成できる。複雑な計算処理が必要になったときは，部分的に Objective-C を用いてプログラムを記述する。NextStep 自身や部品も Objective-C で記述されている。

4・2　知識表現

現在，実用化が進んでいるエキスパートシステムなどの知識ベースシステムでは，知識をどのような形で記憶し，利用するかが重要な課題である。そのための知識表現言語としては，基本言語としての関数型言語 Lisp や論理型言語 Prolog のほかに，プロダクションシステムのルール，フレーム，意味ネットワークなどが代表的である[2]。

この中で，フレームは以下の特徴を有し，知識の構造化や知識間の関係の記述が容易なため，多くのエキスパートシステムで利用されている。

（1）　実世界の概念に対応する枠組みをフレームとして定義する。

（2）　フレーム間に階層構造を導入し，下位のフレームは上位のフレームの属性を継承する。

これらのフレームの特徴は，先に述べたオブジェクト指向概念との類似性が強いため，実際には，両者はあまり厳密な区別はせずに使われることが多い。オブジェクト指向知識表現を含むエキスパートシステム構築ツールとして，KEE (IntelliCorp 社)，LOOPS (Xerox 社) などがある。

筆者らもエキスパートシステム構築ツール ES/X 90 (Expert System Tool for 90's) [7]の知識処理言語を次のような2階層モデルを用いて開発した経験がある。

①　全体の知識処理と知識構造はオブジェクト指向言語で表現することにより，マクロなレベルでの柔軟で自然なモデリングを可能とする。

②　局所的な知識はフレーム，ルール，述語論理などの複数の形式のいずれでも表現できるようにすることにより，ミクロなレベルでの多様な知識の自然な表現を可能とする。

ES/X90 による記述例を図9に示す。同好会というクラスは，団体というクラスを親にもち，会員の入会・退会処理を行うメソッドと会員数をチェックするメソッドをもち，会員の増減を管理している。

4・3　データベース

データベースは，これまで主に数値や長さの限られた文字列を扱ってきたが，最近，文章（テキスト）データ，図形データ，イメージデータなど，データ構造が複雑なものを扱う必要が生じてきた。例えば，テキストと表，図，グラフなどが混在する文書を扱うオフィスシステム，更に画像，作図なども含む設計図や設計仕様書を扱う CAD/CAM システムをはじめとして，ソフトウェア開発支援環境の CASE システム，印刷分野，医療分野，教育分野など，コンピュータ化の進展に伴い，多くの分野でそのニーズが高まっている。

このようなマルチメディアデータを扱うデータベースのための基本技術（データモデル）としてオブジェクト指向概念が利用されている。従来のリレーショナルデータベースなどと異なるオブジェクト指向データベースの特徴は以下のようなものである[9]。

（1）　実世界の物や概念をオブジェクトとして統一的に表現できる。情報としてのデータの形態（メディア）が異なっていてもよい。またデータとしては全く同じものでも実体が複数あれば複数のオブジェクトを作成できる。

（2）　複数のオブジェクトからなる複合オブジェクトも一つのオブジェクトとして統一的に操作できる。

（3）　データと手続きのカプセル化機能により，データの構造に依存した操作手続きをデータに埋め込むことにより，アプリケーションソフトウェアを簡単な構造にでき，拡張，保守が容易になる。

（4） クラスの階層化と継承機能により，知識表現に用いた場合と同じように，データベースの知的検索が可能となる。

既に，Prolog, C, Smalltalk などの特定の言語をベースにしたものが商品化されているほか，研究レベルのものとしては，ORION（MCC），MANDRILL（日立）[9]などがある。

4・4 オブジェクト管理システム

オブジェクト指向ベースのアプリケーションソフトウェアを次々と開発していくとき，アプリケーション間で共通のオブジェクト管理の処理をアプリケーションから分離，独立して共有できれば，個々のアプリケーションの開発効率や保守，拡張性が良くなるし，異機種間の移行性も向上する。

このようなシステム構成の例として，HP 社の NewWave がある。NewWave では，テキスト，グラフ，スプレッドシート，ボイスオブジェクトなどを統合して，マルチメディアのドキュメントを簡単に作ることができる。図 10 に NewWave の複合オブジェクトの例[10]を示す。図中の線はオブジェクト間の関係を表わす。シンプルリンクは単純な親子関係を表わす。ビジュアルリンクは子オブジェクトが親オブジェクトと表示や印刷を共にすることを表わす。この例では，チャートオブジェクトのグラフがドキュメントオブジェクトの構成要素として取り込まれており，双方のオブジェクトのグラフ表示は常に同じになる。データパッシングリンクは，子オブジェクトから親オブジェクトへデータ値が渡されることを表わす。この例では，スプレッドシートのオブジェクトから渡された数値データに基づいて，チャートオブジェクトの中に対応するグラフが作成される。スプレッドシートの値が変ればグラフも自動的に変更される。

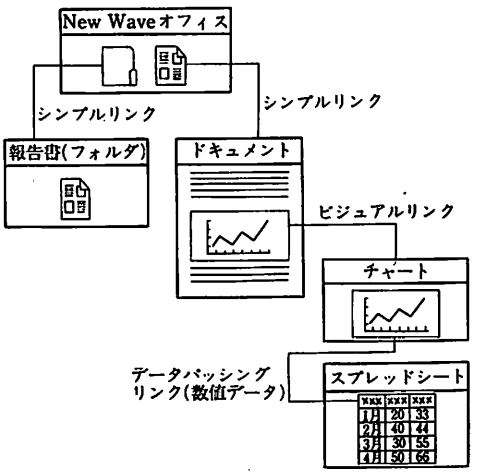


図 10 NewWave における文書データのオブジェクト構成例〔Hewlett-Packard 社（© 1989）の許可を得て掲載〕

このような機能を実現するために，NewWave の OMF（Object Management Facility）は以下のオブジェクト管理操作機能をもつ。

（1） 文書，図，表，音声データなどの基本的なユーザオブジェクトの管理

（2） 文書のホルダのように複数のユーザオブジェクトをまとめて扱うコンテナオブジェクトの管理

（3） グラフや図を含んだ文書のように異種のユーザオブジェクトを含んだ複合オブジェクトの管理

（4） 電卓やキャビネットなどのオフィスツールの管理

（5） オブジェクト間にリンクをはるハイパーメディア機能

5. 発展過程

オブジェクト指向概念とその関連技術の発展は，概ね次の 3 期に分けて考えることができる。

5・1 基本概念創成期（1960 年代後半から 1970 年代半ば）

1960 年代後半にオブジェクト指向概念の重要な特徴であるデータ抽象化，クラスからのインスタンス生成，クラスの階層と継承，などの機能をもったシミュレーション言語 Simula 67[11]がノルウェー計算センタで開発された。1970 年ごろ，MIT の C. Hewitt がメッセージパッシングを主体とした計算モデルである Actor モデル[12]を提案した。1974 年には，B. Liskov が本格的な抽象データ型機能をもつ言語 CLU[13]を開発した。

5・2 言語開発期（1970 年代後半から 1980 年代半ば）

オブジェクト指向言語の代表例として Xerox 社の Smalltalk がある。1970 年ごろから人間の創造的活動を計算機が支援する方法を研究する過程で開発され，1980 年には Smalltalk-80[14]として実用化された。その後，計算機のパーソナル化の動向と相まって，オブジェクト指向への関心が急速に高まり，種々の言語が開発された。そのほ

とんどは，既存の言語をベースにオブジェクト指向機能を導入したハイブリッド型あるいはマルチパラダイム型と呼ばれる言語である。例えば，Prologをベースにした ESP（ICOT），S-LONLI（日立）[15]，Lispをベースにした Tao（NTT），ABCL（東工大），CLOS（ANSI標準案），Cをベースにした C++（ATT），Objective-C などがある。

一方，知識工学の分野でも知識表現にオブジェクト指向概念を取り入れたエキスパートシステム構築ツールとして，KEE，LOOPS などがある。

5・3 応用拡大期（1980年代後半）

最近，オブジェクト指向概念の応用技術分野が急速に拡大している。その代表的なものとして，ソフトウェアの使い勝手の良さを決めるグラフィカルユーザインタフェースとその構築ツール，マルチメディアデータを扱うためのデータベース，システムの分散化に対応するオペレーティングシステムやネットワークなどの分野がある。更に，それらの開発支援のための設計技法[16]や部品化再利用などのソフトウェア生産技術，あるいは，アプリケーションソフトウェアの共通プラットフォームとしてのオブジェクト管理システムの開発などが積極的に行われており，オブジェクト指向ベースのソフトウェア開発環境が充実しはじめている。

6. あとがき

最近の情報化社会の進展の中で計算機システムがますます重要になってきており，「使い勝手の良いソフトウェアを簡単に作る」技術が求められている。その一つの有力な技術として，オブジェクト指向概念が種々のソフトウェア技術に応用され始めているが，まだ実用実績は少ない。オブジェクト指向アーキテクチャを採用した応用ソフトウェアを開発するためには，ソフトウェア生産技術の観点から，オブジェクトおよびメソッドの抽出方法などの設計技法，プログラミング言語とその最適化処理機能を備えたコンパイラ，クラス階層の管理や部品検索などを支援する開発環境，およびユーザインタフェース管理，オブジェクト管理，オブジェクト指向データベースなどの実行環境の充実が必要である。特に，言語，ユーザインタフェース構築ツール，オブジェクト管理システムについては，標準化動向についても注目していく必要がある。

本解説論文の全体のまとめ方について三留和幸氏に貴重なご意見をいただいた。データベースの動向に関しては佐藤和洋氏，ユーザインタフェースの構築法については里山元章氏にご協力をいただいた。（平成2年2月28日受付）

文 献

（1） 米澤：「オブジェクト指向計算の現状と展望」情報処理 **29**, 4, 290（昭63-4）

（2） 中所・芳賀：「オブジェクト指向型言語と論理型言語の融合方式に関する考察」オブジェクト指向（鈴木則久編）p. 133（昭60）共立出版

（3） 中所：「人工知能言語」人工知能（江尻，中野，中所共著）p. 82（昭63）

（4） 坂本：「オブジェクト指向型言語 Objective-C」bit, **18**, 12, 1555（昭61-12）

（5） 田口，他：「コンピュータ新時代を開くオブジェクト指向の潮流」日経コンピュータ 1989年5月22日号，p. 64（平1-5）

（6） "OSF/Motif Features and functionality Track One Course Notes"（1989）OSF

（7） 中所，他：「マルチパラダイム型知識処理言語における対立概念の融合方式」人工知能学会誌 **4**, 1, 77（昭64-1）

（8） 増永：「マルチメディアデータベース総論」情報処理 **28**, 6, 671（昭62）

（9） 山本，他：「マルチメディアデータベースにおけるデータベース言語 MANDRILL/QUEST について」電子情報通信学会データ工学研究会資料 DE 88-20（昭63）

（10） P. S. Showman：“An Object-Based User Interface for the HP NewWave Environment” *HEWLETT-PACKERD JOURNAL* **40**, 4, 9（1989-4）

（11） O. Dahl & C. A. Hoare：“Hierarchical Program Structures” Structured Programming（1972）Academic Press

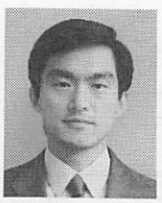
（12） C. Hewitt & H. Baker：“Laws for Communicating Parallel Processes” *Proc. IFIP'77*, p. 987（1977）

（13） B. Liskov & S. Zills：“Programming with Abstract Data Types” *ACM SIGPLAN Notices* **9**, 4, 50（1974-4）

（14） A. Goldberg & D. Robson：“Smalltalk-80 the Language and its Implementation”（1983）Addison Wesley

（15） T. Chusho & H. Haga：“A Multilingual Modular Programming System for Describing knowledge Information Processing Systems” The 10th World Computer Congress IFIP'86, p. 903（1986-9）

（16） G. Booch：“Object-Oriented Development” *IEEE Trans. Software Engineering* **SE-12**, 2, 211（1986-2）



中所武司

昭和21年5月11日生。44年5月東京大学工学部電子工学科卒業。46年3月同大学院修士課程修了。同年(株)日立製作所入社，現在，システム開発研究所勤務，東京工業大学非常勤講師。58年情報処理学会論文賞，61年大河内記念技術賞受賞。工学博士。ソフトウェア工学と知識工学の研究に従事。著書（共著）：「プログラミングツール」，「人工知能」（昭晃堂）。