データベースシステム 94−16
（1993. 7. 22）

オブジェクト指向をベースとした部品の再利用技術について

+ 龍 忠光　+ 泉 寛幸　+ 村川雅彦　+ 市川なおみ
+ 豊田雅信　+ 足立武史　* 戸島哲夫

+ ㈱富士通研究所 情報処理研究部門
* 富士通ネットワークエンジニアリング㈱ 開発部

システムの構築においては、トップダウン的に分析・設計を行いながら一方では構成部品の再利用をボトムアップ的に考えて作業を行っている。前者が「オブジェクト指向設計」であり、後者が「部品化設計」である。この２つの間には大きなギャップがある。このギャップを埋めているのが方式設計者である。この方式設計者の作業を支援し方式設計者と同じ働きをする部品の仕掛けを筆者らは”超部品化技術”と呼んでいる。筆者らは、”超部品化”を行うために、「仮想人間世界」を実現するＯＣＡ［15］をモデルとして導入し、原子部品や複雑な複合部品を考え、これらを超部品化するための研究に取り組んできた。本稿では、この”超部品化”の仕組みと周辺技術について紹介する。

A Reuse Technology of Software-parts Based on Object-oriented Method.

+Tadamitsu Ryu　+Hiroyuki Izumi　+Masahiko Murakawa　+Naomi Ichikawa
+Masanobu Toyota　+Takeshi Adachi　*Tetsuo Tojima

+ INFORMATION PROCESSING DEVISION, FUJITSU LABORATORIES LTD.
* FUJITSU NETWORK ENGINEERING LTD.

On one side, we structure information systems, analyzing and designing with top-down flow and the other side, we consider reusability of software-parts with bottom-up flow. That is called "Object Oriented Analysis" and this is called "Reuseable Design". There is, however, great gap between those two sides, so it is called "System Engineer for Specification Design" who bridges this gap by trial and error.

We planned "The New Technology On Software Parts" that supports him in system design, and exclude trial and error in it. In order to make it, we consider OCA(Object Consideration Analysis) Model that realizes virtual humanity, and we have been enaged in study this technology.

In this paper, we introduce concepts and mechanism in "The New Technology On Software parts".

1．はじめに

現状のシステム開発においては、顧客の業務内容や要求事項を方式設計者が仕様書としてまとめ、プログラマがこの仕様書に基づいてプログラムの作成に取り掛かる手順となっている。ここで方式設計者は、”顧客からの様々な要求攻勢”に会い、一方これを実現させるプログラムの難しさとの板挟みに会い、大変な苦労を強いられることになる。このようにトップダウン的に出てくる顧客の要求が簡単にプログラムに反映出来ない原因は、現在のソフトウェア構造論では、「ファイル構造」，「共通資源」，「タスクレベル」などの設計が複雑になり、実世界のモデル化やその変更が簡単にできないところにある。

筆者らは、モデル化にあたり、日常我々が使っている言葉によるオブジェクト設計を導入することで、この問題を解決することを試みた。

日常我々が使っている言葉の世界、すなわち外延的名辞の世界でモデル化を行う（筆者らはこのモデル化手法を”ＯＣＡ”［15］と呼ぶ）ことにより、従来２～３ヶ月を要していた顧客と方式設計者との打ち合わせ期間は数十日に短縮できると期待している。このための仕掛けとしてその分野（ドメイン）毎に簡単にカテゴリを作ることが出来るカテゴリ作成機能を用意し、この機能を使って”人とのコミュニケーション”を基にした半自動的なプログラム合成方式を採用した。

従来、オブジェクト指向設計で言われている”ＭＶＣ”モデルのうち、最もカテゴリ化（ｸﾗｽﾗｲﾌﾞﾗﾘ化）が遅れているモデル、すなわちＭの部分のカテゴリを作る”カテゴリ作成機能”を準備するものである。筆者らはこれを超部品と呼ぶ。超部品には原子的な部品と複合的な部品があり、これらを組み合わせてどんなドメインでも作り出せる方法論を追求したものである。

筆者らの狙いは、方式設計者を対象としたものでありソフト開発者を対象としたものとは異なり、”人とのコミュニケーション”を重視したものである。部品化の分類としては、「ソースコードを部品化したもの」，「設計要求を部品化したもの」，「設計情報を部品化したもの」などが発表されている［１］［２］が、筆者らの方法は、この分類では「設計要求を部品化したもの」であり、ＰＡＰＳ［２］に似ている。

本稿では、先ずシステム構築に対する現状の問題点を概観し、その後筆者らが提案する”人とのコミュニケーションとは何か”，”ドメインを作る超部品化とは何か”について述べる。

2．ソフトウェア作成における現状の問題点

現行の手続き型プログラミングの場合、効率化を阻害している大きな要因として次の五つが挙げられる。

(1) トリガ条件や動作レベル設定の煩雑さ

現行の手続き型プログラムでは、常にシステム全体の動きを見ながらどういう状態の時にどのプログラムに起動を掛けるかという内部の状態を全て割り出し、それを状態図や状態遷移図としてまとめ、共通して使うフラグやレジスタを注意深く設定しなければならない。

このような設計では、ユーザからの改造要求が来るとほとんど作り変えになってしまう。

(2) ファイル設計の煩雑さ

ファイルの構造の善し悪しはファイルとのアクセス回数に深く係わっており、レスポンスに大きな影響を与えることになる。　また処理プログラムと密接に結びついているため、そのファイルは特定のプログラムでしか使えず、一度作成したものは変更が大変難しいため、ファイル設計の専門家による慎重な設計が要求されている。ファイル設計者は、そのプログラムが使う入力データの種類と性質を全て把握し、それらをどのように加工し、どのように出力するかを知らなければならず大変な工数がここに割かれることになる。

(3) 共通資源の設計の煩雑さ

フラグ，バッファー，ファイルなどの共通資源の使い方は、そのシステムの性能に大きく影

響する。先に述べたファイル設計より深刻であり、そのシステムの生死に係わってくるもので各タスクの起動，終了条件や状態に細心の注意を払いながら設計する必要が有る。

(4) 信頼性確保の難しさ

共通資源などは多数のプログラムから随時参照や更新が掛けられており、条件ヌケなど不具合がある場合、その影響は広い範囲に及んでしまい、場合によっては、システムダウンを引き起こすことになる。

(5) 画面データ作成の煩雑さ

システム設計で一番手のかかるのがこの画面設計である。入力項目，出力項目の設定や多様なデータ（コード系，ドット系，ベクトル系，音系，静止画像，動画像など）があり、また画面上の配置，大きさ，配色，ブリンク処理など操作性や見易さの観点からの考慮事項が多い。画面枚数も簡単に数百枚に達し大変手間の掛かる作業となっている。

これらの不具合点を解決する超部品化の仕掛けについて次に述べる。

3．超部品のコンセプトについて

超部品は、方式設計作業の効率化を狙ったもので、”人とのコミュニケーション”を重視し半自動プログラム合成方式を採用している。このため、操作のビジュアル化を行い、遺伝的アルゴリズムを導入し、画面上での遺伝子操作によるプログラムの自動変更を実現している。ここでは、超部品の狙いや実現のための仕掛けについて述べる。

(1) 動機

方式設計者のコンサルティングのサポートを行う強力なツールとして考え出されたものである。方式設計者は、ユーザとの打合せ時にユーザの話を聞いてシステム設計（トップダウン思考）行い、一方では、それが実現可能かどうかの部品設計（ボトムアップ思考）を行い、この作業を繰り返しながら、トップダウンとボトムアップとのギャップを埋め、見積りが出来る状態にもってゆく。このためのユーザとのすり合わせに約2～3ヶ月程度を要しているが、この作業をサポートし期間を大幅に短縮させることを狙いとしたものが超部品化技術である。

(2) モデル化

システム設計を簡単にするには、実世界をいかに簡単に、忠実にモデルに置き換えるかが重要である。そこで筆者らは、我々が使う人間の言葉が、「概念的認識をする外延的解釈」で行われていることに着目し、外延的解釈を含んだO-id（外延的名辞またはオブジェクトコマンド［7］［8］と呼ぶ）を導入し、この外延的名辞によりオブジェクト設計を行うオブジェクト思考設計法（ＯＣＡ［15］）を新たに考え出した。これにより、通常我々が実世界で行っている事象の捉え方と同じ感覚でオブジェクトモデルを構築出来、オブジェクトの世界に人間の世界を持ち込んだ仮想人間世界ともいえる世界を作り出している。

ＯＣＡでは、上に述べたように実世界の対象（オブジェクト）を外延的名辞の世界と情報隠蔽された内包の世界とに分けて捉える。外延的名辞の世界をオブジェクト世界とよぶ。このオブジェクト世界は、スキーマ論的に分類すると大きくは「静的モデルの世界」，「動的モデルの世界」，「因果関係モデルの世界」の三つに分類される。

これらの概念を図１．に示す。

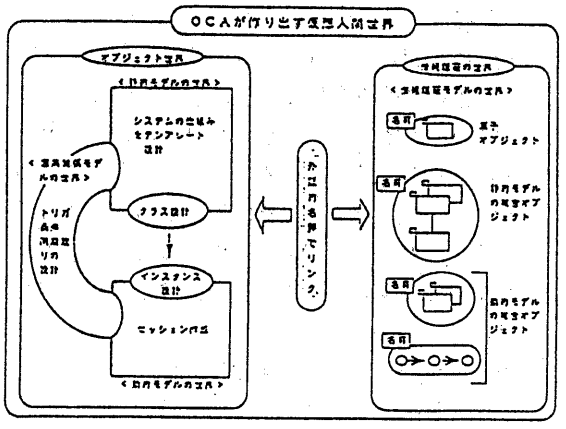

図1．オブジェクト世界と情報隠蔽の世界

① オブジェクト世界における三つのモデルについて述べる。

(a)オブジェクト世界における静的モデルとは、モデル化の対象の”時間を止めても同時に存在する関係”を示すものである。〔15〕

(b)オブジェクト世界における動的モデルとは、モデル化の対象の”時間を動かして存在する関係”を示すものである。〔15〕

(c)オブジェクト世界における因果関係モデルとは、「静的モデル」から展開される「動的モデル」において、時間的関係を示すものである。「静的モデル」，「動的モデル」を構成するオブジェクト世界（仮想人間世界）は、いろいろな法則，規則によって機能する。それは例えば自然法則であったり社会のルールであったりする。動的モデルにおいて定義した機能は、この法則、規則に従って動作しなければならない。通常この法則や規則は、静的モデルで定義するもの（時間軸に無関係なもの）と、動的モデルで定義するもの（時間軸に関係する法則や規則）とがある。後者はインプリメント上は、時間をパラメータにした静的モデルと見ることができる。このような動的モデルにおける時間軸の関係を定義するモデルを「因果関係モデル」と呼ぶ。

このように、外延的名辞によって構成された静的世界，動的世界，因果関係世界はポインタやメッセージで名辞の間を関係付けた、いわゆる「名辞による複合の世界」となっている。

一般に人間の思考は、これらの「複合体」に対してさらに性質を与えたり、名称を付与したりして使われている。一旦、それを定義すると、さらにその名辞によってオブジェクト世界を構成することができる。その際、その複合体の構成子や定義はオブジェクト世界つまり仮想人間世界では、深く認識する必要はなく、外延的解釈で行われる。

② 情報隠蔽された内包の世界における、「情報隠蔽モデル」について述べる。

先に述べた外延的名辞の世界における静的世界，動的世界，因果関係世界から帰結した定義、または原子オブジェクトレベルでの実体，内容全体をモデル化したものでる。

本モデルとオブジェクト世界（仮想人間世界）との唯一のインターフェイスは外延的名辞でとられている。

(3) 超部品化に必要な仕掛け

超部品化は方式設計者が抱える、トップダウンとボトムアップとのギャップを埋めることを狙ったものであり、これを実現させるための手立てとして人とのコミュニケーションを基本にしている。ここでは、部品を使って貰うための「自己ＰＲ機能」，部品の組合せのための「相性合わせ・チェック機能」，最適な部品を早く見つけるための「連想機能」，実行可能なプログラムを効率よく作るための「自動プログラム合成機能」などを提供している。これらの機能について順次説明する。

① 自己ＰＲ機能

他の人が作った”未知の部品”を使うためには、その部品がどのような機能をもつもので、どのように使えばよいかを表現できなければならない。このために部品に持たせるＰＲ項目は

・中に持っている変数のＰＲ
・目的や型のＰＲ
・プログラム仕様のＰＲ（コメント，説明文，管理データ，目的，使い方）など、

であり、オペレータから部品群に要求が来たら各部品は自己ＰＲ機能によって自己ＰＲを行うことにしている。 オブジェクトはメタデータとユーザデータとが組みになって管理されており、オブジェクトの性質をメタデータで表すことにより自己ＰＲを容易にしている。

これらのメタデータはユーザデータから自動的に作られている。

② 組合せのチェック機能

お互いに独立して作られた原子オブジェクト部品や複合オブジェクト部品の組合せは、ハイパー処理機能を使って人とのコミュニケーションにより行われる。組み合わせの為の接続条件

が一致しない場合は、オペレータに通知しオペレータの判断を受けることができる。
これは超部品に必要な直交性を補完するものでである。これにより、シリーズ・ロジックによる隣同志の繋がり，すなわち順次接続のみでなく、離れたところの部品との繋がり（いわゆるナナメ接続）を可能とさせる柔軟性を持たせることができる。順次に結ばれた関係は「直列因果関係」，ナナメに接続された関係は「ナナメ因果関係」と呼ぶ。この機能により、各々の小さな部品は、すべて直交するように相性チェックがなされ、引数合わせが行われてさらに大きな機能を持つ複合オブジェクト部品として作り出される。このようにして作り出された複合オブジェクト部品もまた一つの部品として組合せに使用される。
図２．にその概念を示す。

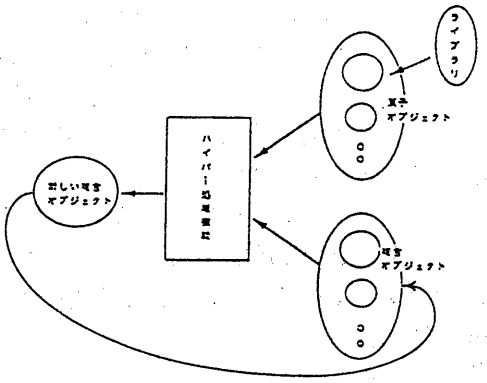

図2. 複合オブジェクトの組み合わせ

③ 部品の連想と創造機能

連想や創造を人間的に捉えると、第六感にピンとくる閃きが基本になっている。 これは、知識の属性と関係が一部変わることにより起こると考えられる。これを部品に当てはめて考えると、その部品を特徴付ける属性に注目し、この属性が一部変わったもの同志の繋がりを掴むことにより同じように連想と創造の機能を持たせることができる。この属性を表すものをキーワードとし、このキーワードに対し文字による

メンバーシップ関数の考えを適用する。ユーザデータからその特徴を表す情報をメタデータとして取り出しこのメタデータからメンバーシップ文字関数として機能するシグニチャが抽出されており、このシグニチャは部品の遺伝子として機能し部品の連想や創造を引き起こす基となっている。この遺伝子による連想と創造の概念を図３．に示す。
ここで、オブジェクトはメタデータとユーザデータで出来ておりメタデータを記号圧縮したものがシグニチャであり遺伝子的な働きをする。この遺伝子である性質は、属性（データ）とメソッド（プログラム）との双方に働くものである。 図３．において、一つがＮＵＬＬになるだけで一つ上のスーパークラスが，またさらに一つＮＵＬＬになるとさらに一つ上のスーパークラスが出来る。これが、遺伝子操作による部品の階層化である。
この操作を逆にしたものが図３．(c) である。図は、遺伝子の一つを別のものと取り替えると”一種の連想”となり、追加してゆくと”一種の創造”となってゆく様子を示している。

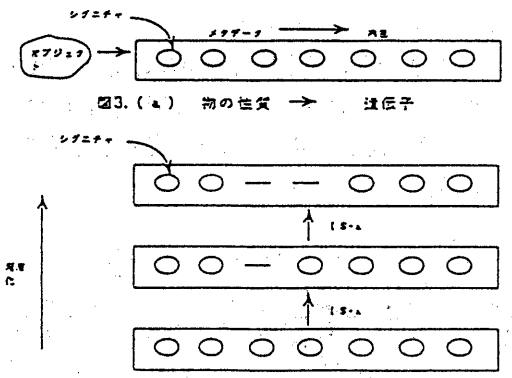

図3.（a） 物の性質 → 遺伝子

図3.（b） 遺伝子操作による部品の階層化

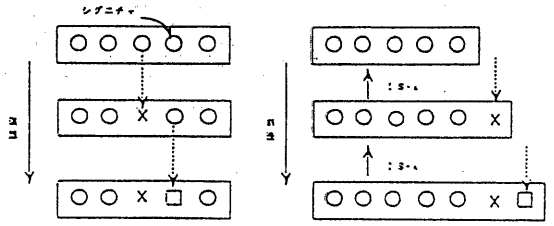

図3.（c） 遺伝子操作による連想と創造

④プログラムの自動合成

これまでに述べてきた①～③により、人とのコミュニケーションを交わしながら作成された部品は、ハイパー処理機能によりプログラムが正常に動作するための要件が付与されている。筆者らは、ハイパー処理機能が持つこのような特性を自動プログラム合成と呼ぶ。この要件について以下に述べる。

(a)部品が定義体であること。

(b)部品がリスト構造になっていること。

仮想人間世界で定義された部品は、手続き型プログラムのソースコードやロードモジュールとして存在しているのではなく、外延的名辞によるリスト構造を持った定義体として存在している。これにより部品の再利用が促進され③に述べた連想や創造が可能となる。

(c)オブジェクト管理による実行制御。

定義体毎にオブジェクト管理部が自動生成され、部品群を実行可能なモジュールに展開し、起動制御を行う。このオブジェクト管理機構は図4．に示すように階層化されている。

(d)カーネル側での動的モデル，因果関係モデルのスケジュール展開。

動的モデルの間の実行制御は、因果関係モデルにより行われる。カーネル部は実行時にこれらの因果関係をスケジュール展開して保持している。

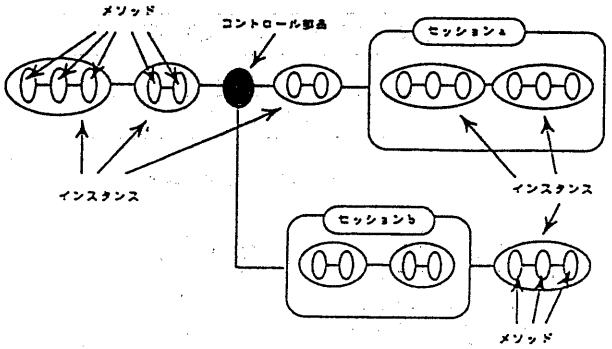

図4. オブジェクト管理の階層構造

4．超部品の構造論

部品を超部品として機能させるためには、原子オブジェクトの構造、複合オブジェクトの構造、遺伝子オブジェクトの構造の3つが重要な役割を持つ。これらの構造論は論理3段形式「入力（WHAT），処理(HOW)，出力（DO）」になっており、これをWHD理論と呼ぶ。

WHD理論の他に、複合オブジェクトについてはハイパー処理機能と因果関係処理が，また遺伝子オブジェクトについてはメタデータ処理，遺伝子処理，ビジュアル化処理が大きな係わりを持っている。

超部品は、完全な独立性，直交性，単純性，抽象性，正規性を有しておりこれを組み合わせて柔軟性が実現されている。これらの5つの特性についてのべる。

「完全独立性」について、

部品自身が処理するデータの授受を統一されたインタフェースのみで行うことを意味し、これを司るのが「コマンドリンク」である。部品間のパラメータや因果関係におけるフラグ・共有資源のデータの入出力は、すべてこのコマンドリンクとのインタフェースだけを意識すればよく、他の部品との複雑なインタフェースは考えなくてよい。従って部品に対して高い独立性を保持できる。

「直交性」について

ハイパー処理機能により部品を選んで，相性診断を行い，引数合わせを行う。この過程で人とのコミュニケーションにより因果関係が作られる。この引数合わせの段階では、オペレータがその値を決めるオーダ値と、処理の結果としてメッセージで渡されるメッセージ値とがある。

「単純性」とは、先に述べたWHD構造のことであり、入力データ、処理、出力データの形をしており処理を設定すると自動的に入力データと出力データが決まり取り出され、自己PRにも利用される。

「抽象性」もWHD構造に関連しており、90%がデータで10%がプログラムであるような

部品として考えだされた。　だから人の手によって９０％のパラメータをセットできることになり、固定的な手続きプログラムの部分は１０％程度に押さえることになり、抽象性，汎用性を格段と向上させている。

「正規性」とは、似た部品が多く出来ないようユニークにするためのもので、このためには一つのオブジェクトがあまり大きなステップ数にならないようにし、約５０～２００ステップに押さえ、外延的名辞でオブジェクトの世界を記述しやすい単位に止めている。

先ずはＷＨＤ理論について述べる。

(1)　ＷＨＤ理論

これは情報隠蔽された部品の構造論であり、単純な機能を持つ「原子オブジェクト」や複合された機能を持つ「複合オブジェクト」に共通な構造を規定するものである。

原子オブジェクトを組み合わせて作った複合オブジェクトにおいても外部から見た形はＷＨＤ構造の形に整えられている。

情報隠蔽された部品の構造は、図５．に示すように論理型３段形式になっている。ＷＨＡＴが入力データ，ＨＯＷが処理，ＤＯが出力データである。

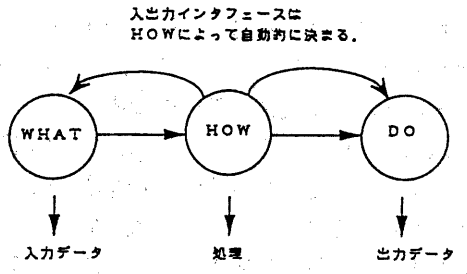

図5.　ＷＨＤの構造

①入力条件（ＷＨＡＴ）について述べる。

入力条件には大きく分けてデフォルト値（オーダ値等）と流動値（メッセージ値等）との２つがある。デフォルト値のセットの仕方には、オペレータが値をセットする場合と定数のセットしてあるバッファーから取ってくる場合とがある。流動値の場合は、シーケンスに並んでいる前の部品の出力データを受ける場合と因果関係で結ばれた部品から受け取る方式とがある。

②ＨＯＷの部分は処理手続きがパラメータ化されており、大きな機能を持ったものでも小さな機能のものでも対応できる。中には大変複雑なオブジェクトを部品として作らなければならない場合も出てくる。この場合、論理型３段形式では対応しきれないことも想定され、これについては４段，５段というのも考えられる。

この原子オブジェクトのＷＨＡＴとＤＯ、つまりＲｅａｄとＷｒｉｔｅをサポートするコマンドリンク部品である。原子オブジェクト部品や複合オブジェクト部品が行うＲｅａｄ・Ｗｒｉｔｅは全てコマンドリンクにより行われる。これが直交性を成り立たせるための仕掛けとなっている。このコマンドリンクの導入により、ユーザはユーザデータ処理におけるフロー作成に必要なO-id をセットするだけで済み、複雑な引数合わせをする必要がなくなる。

このコマンドリンクの働きを図６．に示す。

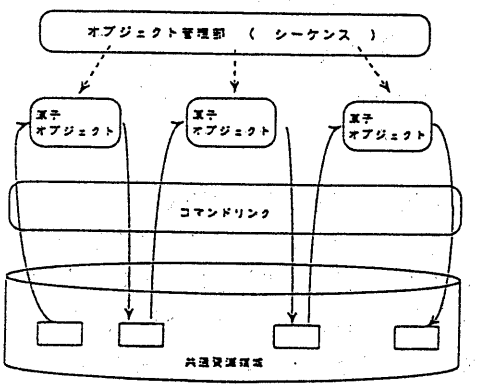

図6.　コマンドリンクの動き

③出力条件（ＤＯ）については、「バッファ値」と「メソッド値」の場合がある。「バッファ値」は実際に利用するもので入力値と同じく、フラグ、数字、文字、コマンド、ON/OFF 値などがある。「メソッド値」は部品単体が決めるものではなくフローの問題であるので複合オブジェクトの構造論のところで述べる。

ここに述べたWHD構造のタイプは、図7．に示すように5種類に分類できる。
図7．において、破線枠内の構造に注目し上からそれぞれ、「比較型」，「合流型」，「分流型」，「End型」，「Start型」の名称を付与している。
通常システムを構成する上では、原子オブジェクトは、基本的な機能を持つもので、その数は少なくいろいろな複合機能の構成要素となるものである。

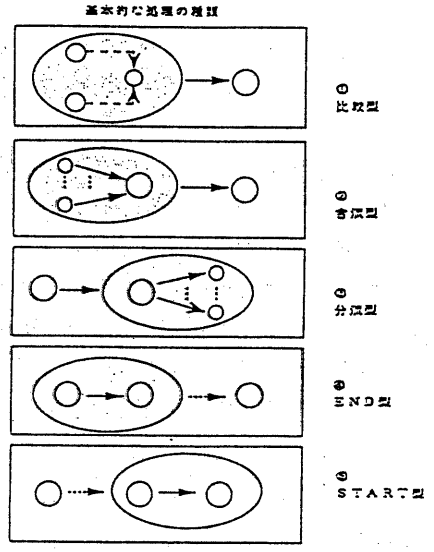

図7. WHD構造のタイプ

(2)複合オブジェクトの構造論

複合オブジェクトはリスト構造で遺伝子で構成されこの遺伝子の最小単位が遺伝子オブジェクトである。複合オブジェクトは、図8．(a)に示すように原子オブジェクトを組合せることで作成する。原子オブジェクトをフロー制御で繋いでゆき、クラス，セッション，などのオブジェクトを作る。このフロー制御を行う部品がコントロール部品である。コントロール部品には、IF−THEN，WHILE−FORの他にNEXT，GOTO，COME−FROM，START，ENDなどがある。また、同期取りに使う因果関係部品は、ナナメ接続のための部品である。

図8．(b)は、コントロール部品により複数の原子オブジェクトを組み合わせてプログラムを作成状況を示している。　STARTから始まりコントロール部品によりCとEの分岐が起こりDとFが合流していく様子を示している。ここで→　はNEXTで表している。

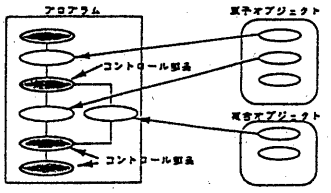

図8.(a)　ハイパー処理における複合オブジェクト作成

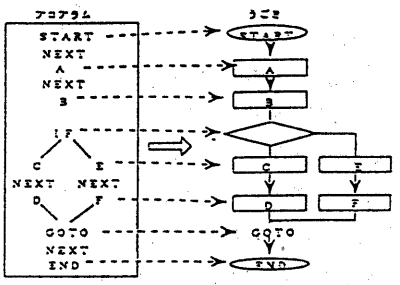

図8.(b)　ハイパー処理におけるフロー制御

(3)　遺伝子オブジェクトの構造論

遺伝子オブジェクトの構造を図9．に示す。この遺伝子オブジェクトは、ハイパー処理機能によりオブジェクトの部品ファイルのメタデータとしてセットされこのメタデータがシグニチャ（遺伝子）に変換され遺伝的に機能する。これを遺伝子進化（順リンク）と呼ぶ。この遺伝子はビジュアル化されハイパー処理機能により活用される。このビジュアル化は、画面ドリブン方式となっており、図9．においてハイパー処理機能からビジュアル化機能へ起動をかけ、その後は画面ドリブン方式でハイパー処理を進める。遺伝子データは、ハイパー処理でメタデータが作られた後で画面とは別に作られる。一方、シグニチャを修正するとメタデータの方が変わるようにも出来る。これを遺伝子操作（逆リンク）と呼ぶ。これにより要求仕様からプログラムの実体を作り出す手立てを与えている。

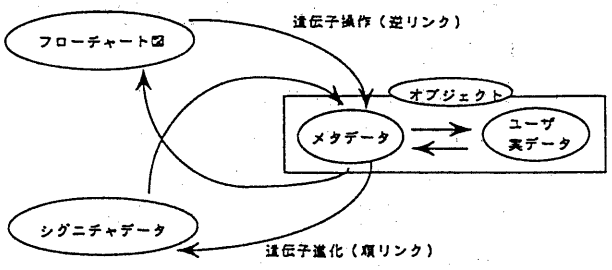

図9. 遺伝子オブジェクトの構造論

5．具体的な部品とシステム設計

(1) 超部品化手法から見たカーネル部とアプリケーション部

人とコミュニケーションしながら部品を選択し、カスタマイズを行いながらシステムを作り上げてゆく全体の流れを図10. に示す。

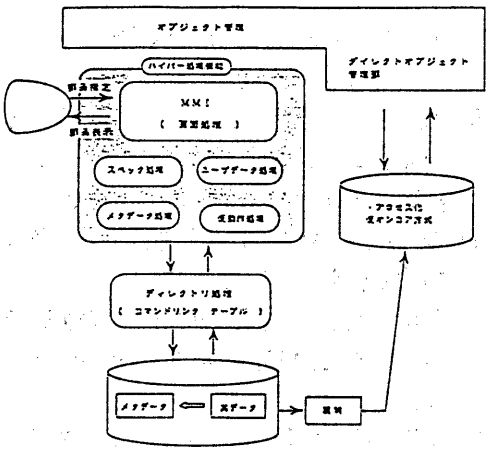

図10.オブジェクト思考設計全体の仕掛け

図において、ハイパー処理機能により使えそうな部品が選択表示される。属性指定により、クラスオブジェクト部品を、またスキーマ設定によりインスタンス・オブジェクト部品を生成することができる。部品表示には、部品のメタデータである名称やコメントの出力をする目次表示、さらに部品の内容を表わすスキーマや属性の表示，クラス属性，インスタンス定数の表示がある。

”部品組合せ”すなわちオブジェクトを作るには、複合クラス作成として属性追加，変更，削除，複合インスタンス作成として、スキーマ追加，変更，削除機能が用意されている。

全体の動きとしてはこの他に、画面作成を簡素化するユーザ画面作成機能や作ったシステムを正常に動くかどうか一時的に動かしてみる仮動作機能などがある。

(2) 部品の種類

表1．に示すように、「一般部品」，「フロー制御部品」，「データのリード／ライト部品」，「同期取り部品」等に分類できる。このうち「一般部品」以外は、カーネル部として提供されるのでユーザは「一般部品」のみを作ればよい。またこの「一般部品」についても基本的なものは、ライブラリとして提供されるのでこれらを組み合わせて使えばよく、ユーザの負担を大幅に軽減できる。

超部品の種類	具体例
一般部品	表示処理（タイムアウト，　） 変換，タイマー 画面制御（Mコントローラ） AND，OR， 検索，比較，合成分解など
フロー制御部品	IF，WHILE，イベント発生 GOTO，COMEFROM
データのリード/ライト	領域確保，名称登録，read/write 定義域チェック，時系列DB，RB
同期取り部品	同期条件セット，制約セット

表1 超部品の例

(3) 自然なシステム設計

ＯＣＡの導入により、我々は、システム設計を行う場合、仮想人間世界ともいえるオブジェクト世界の中で、モデルの概念設計を行い、ハイパー処理機能のビジュアルな画面で最適な部品の連想・検索や部品間の相性合わせを可能としている。さらに、シグニチャの遺伝子の性質を最大限に利用したスペック指向［14］を導入すれば、あたかもそのドメインの専門家の知識を利用でき、またより人間っぽい人工淘汰的な進化や徳を持った部品やプログラムの実現が期待できる。。

6．まとめと今後の課題について

(1) まとめ

以上、部品の再利用を中心に筆者らがシステム開発の在り方について問題として認識している点を述べ、この問題に対する筆者らの考える解決策を述べてきた。 筆者らの問題認識は、いかに人間の世界に近い感覚でモデル設計ができるかと言う点であり、この点について外延的名辞によるオブジェクト世界（仮想人間世界）での設計を実現させるＯＣＡのモデル化を提案してきた。ここでは、コミュニケーションを重視した半自動プログラム合成方式を採用しており、より自然なシステム設計が行える方式を目指していることに注目願いたい。

(2)今後の検討

「因果関係における優先順位の取扱」，「現在のＧＵＩをもっと使いやすくするためのＧＵＩのパラメータ化」，「ディレクトリ処理における時系列ＤＢ，ＲＤＢ，ＯＯＤＢの作り込みやシグニチャ判断，エノバージョン判断などの新しい機能の折り込みとこれによるメモリサイズの膨張抑制対策」，「負荷をかけずに、シミュレーションレベルでのコンパイルをなくす方法」などの検討が必要である。

今後、スペック指向を進めてゆき、

・人とのコミュニケーションを自動ラーニングすること
・人とのコミュニケーションのコモンセンスの規則（プロダクションルール）をいれること

等を行い、相性診断・引数合わせを自動的に行わせるメカニズムを組み込み、より自然なシステム構築手法としたい。

なお、今回当該研究の開発の機会を与えて頂いた大規社長に感謝致します。

参考文献

〔1〕古宮，原田 「部品合成手法による自動プログラミング」 自動ﾌﾟﾛｸﾞﾗﾐﾝｸﾞﾊﾝﾄﾞﾌﾞｯｸ ｵｰﾑ社 P121-140 1989.12

〔2〕古宮 「部品合成によるﾌﾟﾛｸﾞﾗﾑ 自動合成ｼｽﾃﾑＰＡＰＳ」，情報処理学会研究報告 87-SF-21 1987.6

〔3〕E. Yourdon, and L. Constantine, Structur-ed Design, Yourdon Press, 1979.

〔4〕S. シュレィアー，S. J. メラー著，本位田，山口訳，オブジェクト指向システム分析，啓学出版，1990.

〔5〕P. Coad, and E. Yourdon, Object-Oriented Analysis, Yourdon Press, 1990.

〔6〕J. ランボー他，羽生田監訳，オブジェクト指向方法論ＯＭＴ，トッパン，1992.

〔7〕龍，佐藤，高原「分散処理における新データモデルの提案」 SITA '90

〔8〕龍，筧，青江 「分散システムにおけるファイル構造の提案」情報処理学会ﾃﾞｰﾀﾍﾞｰｽｼｽﾃﾑ 1990.7

〔9〕村川，龍「ｵﾌﾞｼﾞｪｸﾄｾﾝｻｰﾓﾃﾞﾙ におけるﾒﾀﾃﾞｰﾀ ｱｰｷﾃｸﾁｬ の提案」情報処理学会ｱﾄﾞﾊﾞﾝｽﾄﾃﾞｰﾀﾍﾞｰｽｼﾝﾎﾟｼﾞｭｰﾑ 1991.7

〔10〕龍，若林、村川「オブジェクトの捉え方とオブジェクトセンサーモデル」 情報処理学会ｱﾄﾞﾊﾞﾝｽﾄﾃﾞｰﾀﾍﾞｰｽｼﾝﾎﾟｼﾞｭｰﾑ 1991.7

〔11〕龍，戸島，村川，豊田「自己組織化通信方式の提案」 SITA '92

〔12〕豊田，足立，龍「複合ｵﾌﾞｼﾞｪｸﾄを表すｾﾏﾝﾃｨｯｸ ID 方式」情報処理学会ﾃﾞｰﾀﾍﾞｰｽ研究会 1992.11

〔13〕市川，龍，戸島 「セマンティックIDによるメッセージパッシング方式の提案」処理学会ｱﾄﾞﾊﾞﾝｽﾄﾃﾞｰﾀﾍﾞｰｽｼﾝﾎﾟｼﾞｭｰﾑ 1992.12

〔14〕村川，龍，戸島，豊田「オブジェクト指向設計を支援するハイパー処理機能の提案」 情報処理シンポジューム 1993.1

〔15〕泉，龍，村川「リアルタイム性を追求したオブジェクト思考設計法（ＯＣＡ）」 1993.5.

〔16〕龍，村川，豊田，足立，戸島 「オブジェクト指向設計法における分岐制約処理」 1993.5