オブジェクト指向とPowerAEの比較

－有用性についての一論考－

井上　弘行

倉敷芸術科学大学図書館

（1999年9月30日　受理）

序　文

かつて，手続き型言語の全盛期時代，世にICが出始めた頃のことである。技術の水準が低かったり，機械そのものが高価であったりで，システムの稼働に向けられる投資効果には，きわめて厳しいものがあった。そこでは，多くの人材投入と共にプロジェクトが組まれ，綿密な計画のもとに任務は遂行された。受注の多くは，新規単発型が多く未経験分野が大半であった。よって，完成品に至るまでには，勢い頻繁なテスト試行の繰り返しとなった。それは，直接原価に響き，システム従事者は余計神経質になったものである。

ところが，いまや計算機は，かつてのテレビや冷蔵庫の普及のように，多くの家庭で見られる姿となってきた。いつか辿った高級商品と同じく，計算機の大衆化である。そこには，稼働に向けた緻密な分析，真摯な追求心は影を潜め，慎重さを通り越している。

筆者は，いま出来合の図書館システムを相手に草むしりを進めている。杵柄を頼りに，ユーザ思案の取りまとめである。パッケージという世界と，DBというファイルのような器と，UNIXやWINDOWSにみられるOSの進歩から，その様相の変化は，有為転変のごとしである。作業の進行を眺めていると，アプローチ方法の変容には目を見張らされる。SEには，瀬戸内三橋の果たす役割と何か似た共通点が求められる。だが，パッケージとして基本は揃っていると言ってしまえば，それは，「驕り」となる。現実に学生サービスへの向上を謳うなら，何をいわんやかではないかと思われる。「柔軟かつ協調」的発想と常に相手を慮る精神を求めてやまない。人の感性も，システムの完成も，十分なコミュニケーションと信頼を礎に成就する。ラウンドトリップ式の漸進は，初心者には受けても，概してエネルギー消耗は激しく，疲労感を残すだけである。ウォーターフォールとは対照的である。

生産・消費サイクルが近年非常に短くなった。システムしかり，プログラムしかりである。そこで，昼間は図書館システムの電子機械化に輔成しており，深い洞察，究明は無理である。が，上に述べたようなことに鑑みて些少とも階梯になればと，システム開発支援としてのPowerAE（Application　Environment：以下P. AEと略記）について一考を投じてみたい。

言葉というものは，空間的，時間的に知識を伝搬し記憶の形成を演ずる。しかしである。出来合のものは，その意図する分野のことにしか通用しない。それも，すべてが叶うかといえば，嘘である。何某かへの焦燥感は免れない。これをどうするかが問題である。

地球上に自然言語の種類は，何百，何千もあるといわれる。P. AEが，我々の日常語を理解するとは言わない。計算機の人間理解は，明らかに知識概念を形象し，対象とする分野によっては成果もみられる。ビジネスシステムは，科学という世界とはややもすれば距離をおかれ一線を画されてきた。けれども，近代日本を支えてきた「産官学」は，いま新たな次元を生み出そうとしている。P. AEの開花を期待する所以である。

オブジェクト指向は，特定の言語を指すものではない。以下，その観点からP. AEを捉え，有用性・利点などについて一考察を加える。

1．概　要

1970年代，第3世代言語が隆盛を迎えている頃，オブジェクト指向の基本概念は盛んに研究されていた。機能分割法から，構造化手法，そしてCASEまで各種の分析と構築の手法は，次々と世上に誕生している。一方，対応するシステムの規模，機能要求は拡大の一途を辿る。そんな中で今世紀最後の10年が始まるとき，バブルの崩壊と共に事務処理システムの開発環境は，大きく変わり始めた。汎用機による基幹システムは分散型に移行し，PCの飛躍的性能の向上は，開発支援ツールのPowerAEを誕生させた。

まずは，その特徴を集約してみると，次のような観点で開発環境の昇華に努めて来ているのではないかと観測される。

- プログラムの論理思考からのユーザ解放。
- 個としての徹底した部品化概念の導入。
- 開発環境としてのワークベンチの充実と豊富な関数群。
- ドキュメント性，並びにメンテナンス性の向上，および，再試行性の重視。
- 同時分散開発を可能とし，しかもセキュリティを堅持。

情報処理システムと言えば，専門家集団の特許であった。ロジックの組立は，初心者には計算機を縁遠くする元凶であった。もはや，これらは比ではない。ツールの定義要求に応じるのみである。計算機との紐帯を何百倍，何千倍と人に近づけたとも言える。さらに，簡単な機能は，システムの基本処理が担い大きな魅力に映る。ディレクションには，日常感覚さながらの簡易言語が用意されている。評価の列挙には，暇がない。こうした理念に裏付けられたP. AEを，近年のオブジェクト指向を代表するキーワードと比較する。

Object k. w.	PowerAE
オブジェクト指向	イベント駆動方式である
カプセル化	項目にメソッドを施し，発火はイベントにおかれる
クラス	ファイルやタプルと同等にみなすと，項目間の関連強度の問題である

継承	上下の階層はなく，テーブルの与え方でプロセスレコードに反映
多相性	予めの埋め込みにより，オブジェクト指向性は強くない
メッセージ通信	メッセージ・パッシングは弱く，イベント中心でプロセス－プロセス起動も許す

オブジェクト指向とは，データ指向である。同時に，プログラミング指向でみる限りは，P.AEをオブジェクト指向と呼ぶには距離がある。記述仕様が異なれば，双方を俎上に乗せるのは難しい。対比に触れながら，P.AEにその片鱗を探っていく。

図1に，P. AEのワークベンチを見せる。ツールアイコンは，数多く揃っている。

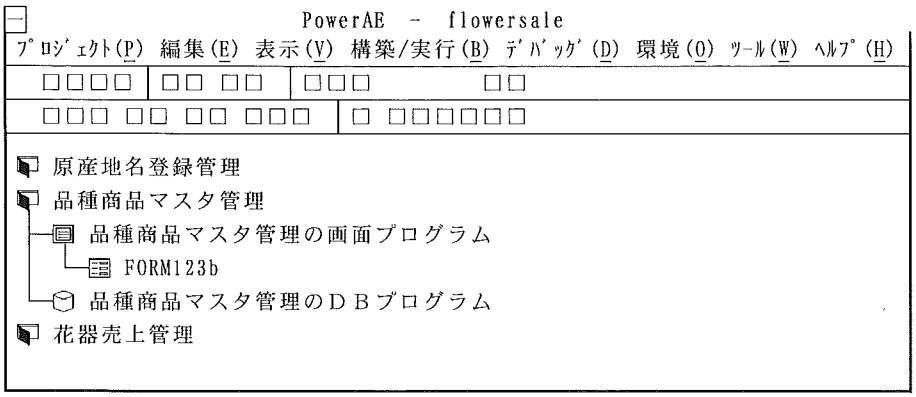

図1　ワークベンチ

2．プロセス

PowerAEは，プロジェクトを有する。その中味は，プロセスである。従来のプログラムと位置づけてもよい。処理単位と捉えてもよく，図2は，その構成である。

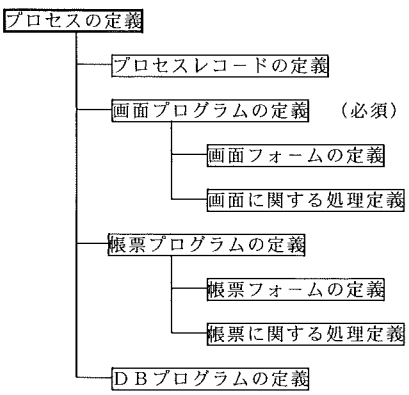

図2　プロセスの構成

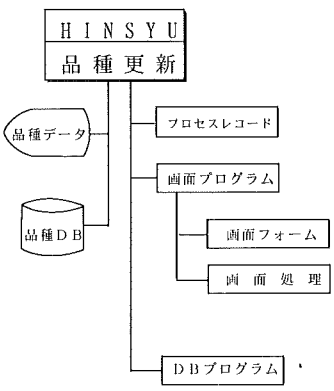

図3　プロセスの一例

プロセスを語るとき，フォームの自動生成とも深く繋がる重要な核にプロセスレコードがある。物理記憶を司る単位様のものではない。一つのプロセスが関与するファイルから，必要なものを抽出してでき上がる項目集団で，論理レコードに相応する。複数のファイルから見通すこと，あるいは，作業用エリア的なものも含めて参照可能としている。コボルのFILE SECTIONとWORKING-STORAGE SECTIONを統合したようなものである。

P.AEで処理を組み立てるには，対象業務を分析し，何処に，どんなプロセスが有効であるかを見極めることが大事である。プロセスの形態決定は，システム設計者の責務であり，運用時の良否に直結する根幹をなしてくる。図3にプロセスの一例を挙げる。

3．辞　書

ファイル項目の定義や処理の定義を簡潔にするために，データ辞書が導入される。最も基本となるものが，ドメイン辞書といわれるものである。項目には桁数とか文字種の属性が要求される。これを明らかにするところのものである。個々の形で与えることも可能ではあるが，辞書に統一的に登録しておけば，必要時にして引用で済ませられる。非常に効率よい定義が行える。また，修正が必要になった場合，辞書のみを変更すればよく，メンテナンスにおける精度低下の防止と所要タイム軽減に結びつく。

辞書類には，ドメイン辞書の他に項目辞書，部品辞書，および業務関数辞書がある。これらの辞書類は，必要に応じて設定する。当然のこととして，辞書類は，他の各定義に先行する。逆の要求は，特命扱いとなる。

ドメイン辞書　→項目辞書定義
ドメイン辞書・項目辞書　→ファイル定義
部品辞書・業務関数辞書　→処理定義

これが効果の作用で，業務関数に関しては，従来の手続き型言語の知識が必要である。

4．データ記憶

P.AEでは，データ記憶の取り扱いとしてDBを使う。物理レベルのシリンダ，トラックをはじめとして，エンティティやインスタンスのような概念は必要としない。DB操作周辺のDDL / SQLやスキーマなどの知識も，一切問われない。定義には，「ファイル」なる用語が顔を覗かせるが，体にまとう衣服程度の知識でよい。索引順ファイルとかブロッキング，クラスタというようなデータの分割・集約を求めるものではない。そこには，データ構造や操作といった機械的側面は，もはや隠蔽の世界とされている。一種のカプセル化への道である。ただ，性能面を問うなら，データの配置に関して緻密性が求められる。

オブジェクト指向でいうクラスの概念を表すファイルは，データの記憶を決める。これには，定義としてドメインや項目辞書の力を借りる。ここで重要なのは，データの一意化を決定するキー項目である。主キー，候補キー，二次キー，参照キーの4種類がある。

ファイルの構築において，神経質になることはない。日常の手にする紙ファイルと同等な感覚でよい。一つにまとまる集団を対象に据えればよい。P.AEは，そこまで機械的属性の世界から解放してくれる。

ある事柄が決まれば，それに基づいて他のものも決まるという関係は，しばしば遭遇する。クラスの相続関係に近いもので，親子関係，子孫関係である。これは，ファイルの自動参照を実現する。キーの指定と並行して，file 1 →file 2，file 2 →file 3というように，各参照先のチェイニングを与える。後は，主ファイルのデータ引用の度に，プロセスは関係する先々のデータ値を取り込んでくる。

ファイルは，プロセスからみて読み書きする直接の対象ではない。必要にして，要求を与えるに過ぎない。それは，指定アイテムのみを取り込む形をとる。ユーザからみると全項目を要求した時においてのみファイルのレコードと一致する。

プロセスの作動結果についても，心配することはない。DBの状態は，いつでも簡単に手に入る。ただ，表示させて答えを確認するだけならそれでよい。用があれば，一時的に記憶値の変更も可能である。テストフェーズからすると，非常に便利と言える。セキュリティ面からすると，両刃の剣か。ファイルの定義を図4にみせる。

ファイル定義－品種ファイル

ﾌｧｲﾙ(F)　編集(E)　表示(V)　ﾌｧｲﾙ項目(I)　ｷｰ(K)　ＤＢ表定義(D)　ｳｨﾝﾄﾞｳ(W)　ﾍﾙﾌﾟ(H)

品種ファイル

	項目名	項目英名	ドメイン名	型	符号	ALL	DEC	DB	
1	品種コード	HINCODE	整数５桁型	外部十進	符号なし	5	0	Y	
2	登録日	ENTRY	西暦日付型	外部十進	符号なし	8	0	Y	
3	品種名称	HINMEI	品名型	日本語	－	20	－	Y	
4	原産地	HOME	地名型	日本語	－	15	－	Y	
5	成長サイズ	SIZE	実数Ｉ型	２進	符号なし	3	1	Y	

	キー名	ｷｰ種別	構成項目	参照ｷｰ	参照先ﾌｧｲﾙ名	参照先ｷｰ名
1	品種鍵	主	品種コード	×	－	－
2	品種鍵の原産地	－	原産地	○	原産地名ﾌｧｲﾙ	原産地名鍵

図4　ファイル定義　　ALL：全体桁　DEC：小数桁

5．処　理

オブジェクト指向での表現を借りるとメソッドということになるのであろうが，P.AEには，そういう用語はない。また，メッセージ（用向き）は，セレクタを介して要求を依頼するが，そこまでの進化も見られない。メッセージ・パッシングではなくイベント内で起きた具象を伝えるに過ぎない。形はクラスに与える手続きに相当するものである。

処理の定義は，動作を起こすイベントをベースに行う。それぞれのイベントは，次のような処理定義におかれる。

画面処理定義／画面項目処理定義
帳票処理定義／帳票項目処理定義
DB 処理定義／関連項目処理定義

こうした処理定義は，多様な形で展開し，その場に適した方法を選べる。客を待たせていたりで，急ぐときなどは，返ってどれでやろうかと迷うくらいである。

プロセスの各プログラムから，プロセスレコードの定義から，フォーム定義画面から，さらにはフォーム属性画面からといったような具合である。図５に，画面の場合のイベントを見せる。ユーザは，プログラムロジックの組立に追われる必要はない。しかし，これらのイベントが起こす動作・機能は知っておくと，よりきめ細かいサービスを期待できる。

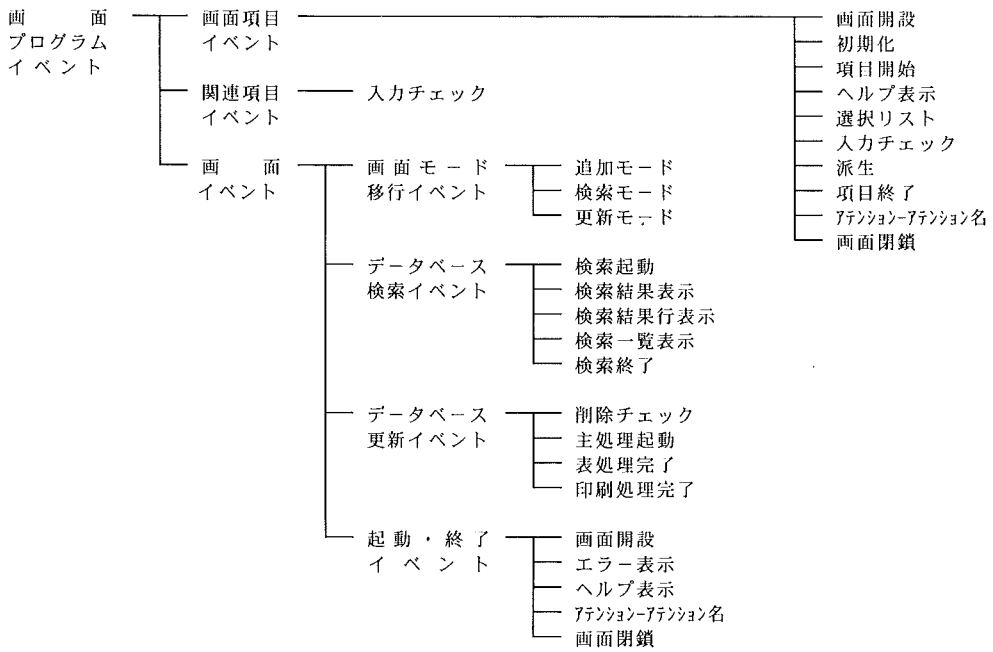

図５　画面プログラム用イベント

処理について，もう少し詳しく覗いてみる。すなわち，各処理の定義は，目的の項目，したがってオブジェクト流にはクラスの属性に対し，処理要求のタイミングとしてのイベント，代入，処理，条件の４点がセットである。

代入は，処理結果の受入を明示するものである。処理は，後に示す簡易言語を使って記述し，また関数も指定できる。条件については，複数個の条件フレームが用意されている。各フレーム間には論理積が成り立ち，AND 条件の記述は比較的容易である。

これに対しOR条件は，一つの条件フレーム内で一気に書き上げなくてはならず，この記述は大変である。

例を挙げてみる。第1四半期を「月」だけでもって示したいなら，次のような感じである。

例1　途中部分文字列(年月日，4，2)＝"04"　OR　途中部分文字列(年月日，4，2)＝"05"　OR
　　途中部分文字列(年月日，4，2)＝"06"

これは，また，次のように表現しても結果は同じである。

例2　途中部分文字列(年月日，4，2)＞＝"04"　AND　途中部分文字列（年月日，4，2)＜＝"06"
　　（注：ANDは，フレーム分割可）

これらを見て一目瞭然，その記述には少々の骨が折れるという次第である。記述が長いというだけでなく，全角と半角，かな，数字などのキー操作が大変なのである。

これは親疎性の問題であるが，ユーザは命令を憶える必要はない。特にコントロールブレイクとサマリ，ファイル・マッチング等について，いわゆるプログラマをいたる所で泣かせて来た「ロジックの呪縛」から解き放たれる世界が，そこには開かれようとしている。この点は，今後の成長が期待されるツールの大きな魅力といえる。

6．簡易言語

PowerAEのイベントに与える処理（メソッド）は，簡易言語で記述する。簡易言語は，文字通り簡易であって処理式と共に条件式にも適用する。日本語表現であるから，日常的感覚で要求を出せる。これまでのような難解な記号をベースにしたものではなく，書き方を憶える学習ステップは，全く不要である。

言語の展開では，制約条項は極力少ない方がよい。この後，式の記述形式をみる。そこでの項目名は，利用者が与える。このとき，先頭文字に「制－」，「前－」，「標－」などは，特殊な意味をもち勝手に付加はされない。いずれもCOBOL言語に連携するもので，それぞれ連結時の全域名，レコード参照の前値，関数呼び出しの領域となっている。

条件式：各イベントに対して指定する条件を与えるもの
処理式：条件が満足された場合に行う処理を与えるもの

式の表現は明確に定められており，その規則に従う必要がある。どういう形をとるか，以下にその記述形式[1]の一部を見せる。その前に，定義記号を簡略に示す。

〈 〉　簡易言語の式を構成する。要素の表現では，この記号自身は示さない。
：：＝　左辺は，この記号の右辺で定義される。
｜　　またはの意。
（ ）　この括弧対は，明示通り与える。

```
条件式  〈条件式〉        ：：＝〈論理式〉                                                    (1)
処理式  〈処理式〉        ：：＝〈論理式〉｜〈算術式〉｜〈文字式〉｜〈項目指定〉
        〈論理式〉        ：：＝〈論理積式〉   ｜〈論理式〉 論理和演算子〈論理積式〉
        〈論理積式〉      ：：＝〈単純論理式〉｜〈論理積式〉論理積演算子〈単純論理式〉
        〈単純論理式〉    ：：＝〈比較式〉｜論理否定演算子〈単純論理式〉
        〈比較式〉        ：：＝〈算術式〉比較演算子〈算術式〉
                              ｜〈文字式〉比較演算子〈文字式〉
                              ｜〈関数呼出し〉｜〈論理式〉
        〈文字式〉        ：：＝〈文字列定数〉｜〈関数呼出し〉
        〈関数呼出し〉    ：：＝〈関数名〉(〈関数引数並び〉) ｜〈関数名〉()
        〈関数名〉        ：：＝〈業務関数名〉｜〈サービス関数名〉
        〈関数引数並び〉  ：：＝〈関数引数〉   ｜〈関数引数〉,〈関数引数並び〉
        〈関数引数〉      ：：＝〈項目名〉     ｜〈数値定数〉｜〈文字列定数〉｜〈関数呼出し〉
        〈項目名〉        ：：＝〈識別名〉
        〈数値定数〉      ：：＝〈整数定数〉｜〈ゼロ定数〉｜〈固定小数点定数〉｜〈浮動小数点定数〉
        〈文字列定数〉    ：：＝〈半角文字定数〉｜〈全角文字定数〉｜〈16進定数〉
```

この記述形式によれば，関数引数並びに関数呼出しをおいて関数の入れ子を許している。最後におかれている半角文字定数は，半角のダブルクォートで囲まれた1024文字以下の半角文字列である。また，比較演算子は，＝｜NOT＝｜〈＝｜〉＝｜〈｜〉の6種類で，論理演算子は，OR｜AND｜NOTの3種類である。

上の処理式において，〈算術式〉に続くものは省略する。ただ，次のものは示しておいてもよかろう。〈項目指定〉の記述に現れるものである。

```
〈項目指定〉    ：：＝  @｜〈項目名〉｜〈項目名〉[〈参照キー名〉]
```

上記中，@は項目自身を意味する。また，その最後のものは，他のファイルの項目を参照する手段を与えるもので，

```
項目名［参照キー名］
```

のような書き方となる。夕ご飯の調理中における，いま，「味噌がいる」，「醤油がいる」の類である。誠に都合よく出来ている。

7．言語比較

最後に，4GLのP.AEが，従来の手続き型言語に比し，どれくらい有効性を発揮するかを見てみたい。

本論の始めにも触れたように，処理の論理フローを組み上げる必要性は消滅している。これは，フローチャートの作成と命令への置換を削減して捉えることが可能ということを意味する。プログラムの作成工程における人材投入比率を，図6のように推定すると，約

3～4割に激減させられると見込める。さらに，テストチェックは，その値に変動はなしと仮定したが，デバッグツールの発達からすると，実際には同等以下に押さえられる筈で，そうなると，益々その値は小さくなるといえる。

作業＼種別	COBOL		P.AE
仕様書の読解	2	→	1
流れ図作成	7	→	0
コーディング	6	→	2
打ち込み	1	→	−
テストラン	4	→	4
計(想定)	20	→	7

図6　作業パワー（単位h）

```
縦計算.
    IF 条件 THEN MOVE 0     TO KOSU
            ELSE ADD  SURYO TO KOSU
    END-IF.
    IF KOSU NOT  =  0 PERFORM  PRINT
    END-IF.
    ～
頁替え.
    ACCEPT  SYS-DATE FROM DATE.
```

図7　命令コード

あるプログラムでの測定によると能力差による変動は否めないが，経験値として活用して差し支えないと思う。このような事が，どうして可能なのか。図7と8を対比してみて戴きたい。図7による記述方式では，流れ図に沿った命令を丁寧に書き上げることを余儀なくされる。図8の処理定義一覧が見せる光景は，必要な処理を書いた文書である。命令ではない。この違いが，それだけの大きな差異，ひいては有効性を示すのだと言える。

項目処理定義一覧

ファイル(F)　編集(E)　表示(V)　AA／BR連携(A)　ヘルプ(H)

項目名	イベント名	代入	処理		条件
品種合計	縦計算明細	=	@ + 数量		注文検索条件
	制御頭書き-商品コード		=	0	
	制御脚書き-商品コード		?	印刷抑止設定()	@ = 0
システム日付	ページ頭書き	=	日付獲得()		

図8　項目処理定義

このことは，言語の優劣を表現形式の差異だけで結論づけるものではない。P.AEは，ビジネスシステムを指向したものである。開発効率が最重要視される。それは，目的とする主眼の相違である。オブジェクト指向の代表である，C++のようなものは依然として手続き色は残存している。仕組みを強く意識した言語である。その点，P.AEは，動作と結果は，一体をなしており処理の型依存は強い。その分，マクロより小さいものへの有効手段は乏しいにしても，「開発」面では高く評価されてよい。

おわりに

PowerAEは，汎用機の世界においては，バックログの由来からシステム危機が叫ばれて誕生してきた。その回避を巡って故に，過去の遺産を継承する必要があり，COBOLとの連携も実現している。命令の一群としてではあるが，カプセル化も現実味を帯びて来ている。そうした興味あるものについても触れて見たかった。が，紙数の制約上から機能を探る話題に留まった。

今後，実用の世界に目を向けるならば，データの分割記憶とテーブル／クラスとの関連，あるいは処理速度（レスポンシビリティ）との関係は，…といった性能効率からのアプローチは欠かせない。システムの能力測定には，様々な条件を設定して試行に頼る以外に策はなく，時間の要する作業である。いまは，直接の教育・研究分野にない筆者には，時間的要素は頗る清貧で，そうした面への追究に踏み込めないのは，残念としか言いようがない。

非研究分野からの仕業にして賜ることも憚りながら，高批には大方の寛厳に願いたい。

最後に，システムの利用者に人間味あふれる「やさしい」システム・環境をいつものモットーとして，この機会を頂戴できましたことに感謝する。

引用・参考文献

1）「PowerAE 開発手引書」，V1.0, FUJITSU
2）妻木俊彦・岩田裕道，「オブジェクト指向モデリング」，日刊工業新聞社，1999年
3）J.G. ヒューズ，石丸知之訳，「オブジェクト指向データベース」，サイエンス社，1998年
4）對馬靖人，「ソフトウェア部品」実用的で効果的な再利用システム，㈱丸山学芸図書，1998年
5）井上弘行，「PowerAE」－新しいシステム開発の世界－，富士通応用研究，1997年
6）小泉澄・渡辺昭，「情報システム開発総論」，㈱楽，1996年
7）P.コード／E.ヨードン，中山秀樹他訳，「オブジェクト指向分析（OOA）」，㈱トッパン，1996年
8）戸田忠良・菅沼清吉，「ソフトウェア部品革命」，生産性出版，1996年
9）土居範久，「オブジェクト指向のおはなし」，日本規格協会，1995年
10）井上弘行，「COBOL 基礎文法」，工学図書㈱，1995年
11）内山悟・小林博英，「わかりやすいデータベース設計技法」，㈱ソフト・リサーチ・センター，1995年
12）Grady Booch，「Object-Oriented Analysis And Design with Applications」，2 nd ed., PP38－40，1994
13）㈶日本情報処理開発協会，「ファイルとデータベース」，中央情報教育研究所，1994年
14）A.S.Fisher，黒田純一郎・中島誠訳，「CASE ソフトウェア開発にはCASEツールを使って」，共立出版㈱，1990年
15）チャールズ・ベルリッツ，中村保男訳，「ベルリッツの世界言葉百科」，頁7－12，新潮社，1983年

補遺：本文中，実装の話はしなかった。もし，興味の湧く方は参考文献5（進呈）をみられたい。

Comparing PowerAE with Object-oriented models

Hiroyuki INOUE

Library,

Kurashiki University of Science and the Arts,

2640 Nishinoura, Tsurajima-cho, Kurashiki-shi, Okayama 712-8505, Japan

(Received September 30, 1999)

PowerAE, probably a compound of Power and Application Environment, is one of 4 GL. It's not always an object-oriented tool. There are some differences between PowerAE and the tool which will be shown by making a comparison between them.

Until 1991, it had been said that the world of computers would fall into a crisis in programming. In light of such strong needs, PowerAE was developed to create new systems and to solve backlogs. It's very useful for the development of business systems.

Its main features are as follows :

1. **Liberation from program logic.**
2. **Developing parts of modules.**
3. **Availability of the workbench, which is easy to use.**
4. **Capacity of documents, and maintainability of processes.**

In this paper, subjects are a process depending on the project, the four basic dictionaries including domain, the DB for data storage and the definition of directions. In addition, it's described that a simplified language in Japanese is used for giving orders. As well, a relationship between Method and Event is briefly touched on.

Most importantly, it has been confirmed that PowerAE has a capacity to reduce man hours by about 70%, as compared with coding in the procedure language of COBOL. Consequently, we are assured that this tool can be widely used to shorten development terms in structuring new systems.