ソフトウェア工学 79－4
（1991. 6. 25）

オブジェクトマネジメントシステムにおける時差解析

西岡健自　平田陽一郎　渡邊多恵子

横河電機株式会社　研究開発4部

オブジェクトマネジメントシステム（OMS）は情報の側面でソフトウェア開発支援環境を統合する基盤である．しかし，統合によって，どのような効果を享受できるのか，現状では必ずしも明確でない．

本論文では，われわれの開発した実際のOMS（TENSE OMS）に即して，その有用性を考察する．このOMSでは，ソフトウェア構成要素の概念の導入で情報の粒度を高め，拡張フレーム表現の採用で柔軟なカスタマイズ機能を実現した．また，高粒度の情報を活用して，異なる時点でOMSが保持する情報の食い違いを解析する，時差解析機能を備えている．

STATIC ANALYSIS BETWEEN DIFFERENT PHASES ON AN OBJECT MANAGEMENT SYSTEM

Kenji Nishioka, Yoichiro Hirata, Taeko Watanabe

Yokogawa Electric Corporation, R&D Department 4
2-9-32 Nakacho,Musashino-shi,Tokyo,180 Japan

An object management system (OMS) is an infrastructure of the information integration for a software engineering environment. But the merits of the integration are not so clear.

In this paper we introduce our OMS named TENSE OMS. In this OMS high granular information based on the Internal Objects, and flexible customization faculties based on the extended frame system are available. We will present faculties of the OMS including the static analysis between different phases, and discuss concrete merits of the OMS.

1．はじめに

OMSはソフトウェアオブジェクトを組織的に管理するための機構であり，レポジトリはソフトウェアオブジェクトを格納するデータベースである．ソフトウェアオブジェクトとは,開発プロジェクトのライフサイクルで発生する情報と，使用する情報の総称である．たとえば，ドキュメントなどの開発成果物，プロジェクトの規模などの資源情報，開発計画，開発支援機構，各種解析／評価結果がソフトウェアオブジェクトにあたる[1]．OMSやレポジトリは，SEE(Software Engeneering Environments)やCASE環境の中核であり，情報の側面から支援環境を統合する要である[2][3][4]．

OMSやレポジトリの実現には，ソフトウェアプロセスの多様性に応じるために，カスタマイズ機構が必要であり，情報の整合性を効果的に保つために，高い粒度で情報を統合する必要がある．従って，これらの要件を満たすOMSを実現し，効果的に運用するには，高度な機構を必要とする．しかし，OMSやレポジトリがソフトウェア開発，管理にもたらす効果は現状では必ずしも明確でない．

本論文では，カスタマイズ機能を備え，高粒度の情報を扱う，実際のOMSに即して，その機能と効果について考察する．

このOMSとして，現在構築中のソフトウェア開発支援環境TENSEのOMSを取り上げる．TENSEの開発は，下流から上流に向けて段階的に構築し，提供する方式を採っている．現状は，OMSの開発が終った段階にある．また，OMSのカスタマイズによって，実現したＣ言語向けのプログラミング環境，統合化Ｃプログラミングシステムは社内のソフトウェア開発部門でβサイトテスト中である．このシステムでは，現在，ドキュメントやプログラムのような開発成果物のみを扱うが，多くの支援機能を実現している．

以下，TENSE OMSの構成，支援機能の実現方式，OMSの効果のまとめの順に述べる．特に，支援機能の実現方式については，時差解析という，高粒度の情報を扱うOMS独特の機能を中心に述べる．

なお，本論文では，ドキュメントとプログラムを総称してドキュメントと呼ぶことにする．

2．TENSE OMSの構成

ここでは，OMSの機能の背景として，TENSE OMSの情報モデルと，その表現，実装について紹介する．詳細については，[7]を参照されたい．

TENSE OMSの本来のねらいは，情報の観点からの開発支援環境の統合による，知的活動に紛れ込んだ機械的作業の自動化である．この種の機械的作業には，形式を変えた，同一の内容の記述の繰り返しや，記述の変更に伴う，他のドキュメントへの変更反映作業などがある．これらの作業は一般に煩雑で，開発担当者の負担となるばかりか，知的活動を阻害する要因ともなる．

TENSE OMSでは，これらの作業を自動化，あるいは，不要化するために，ドキュメント相互の情報の整合性に着目する．この観点から，TENSE OMSは実現上，次のような特徴をもつ．

・粒度の高い情報処理を実現するための，ソフトウェア構成要素の考え方の導入．
・柔軟なカスタマイズ機能を実現するための，拡張フレーム表現システムの導入．
・高い応答性能を実現するための，Prolog言語によるOMSの実装．

2．1　ソフトウェア構成要素

ドキュメント間の情報の整合性を損なう大きな要因は，複数のドキュメントで共有する情報の存在である．情報の共有は，上流ドキュメントで定義した手続きを下流でより詳細に定義する場合，あるいは，モジュールで定義したデータを，別モジュールで参照する場合などに発生する．ソフトウェア構成要素は，このような共有情報を一元管理するために導入した，TENSE OMSの情報単位である．

ソフトウェア構成要素はクラスとインスタンスに大別できる．クラスは，ファイルやモジュールと

いった，ソフトウェアの内容の物理的，論理的な分類に対応し，その各々の特性情報を保持するソフトウェア構成要素である．各クラスは対応する分類の部分関係に基づいて，図1のように階層的に表わすことができる．

インスタンスは，実際のソフトウェアの情報を保持するソフトウェア構成要素で，クラスに属し，その情報を継承する．たとえば，あるモジュールに関する情報を一括したものが，一個のインスタンスとなり，モジュールクラスに属す．

インスタンス相互にも，図1のような部分関係が存在する．部分関係の頂点にあるのが，システムクラス唯一のインスタンスsystemである．systemから部分関係を辿ることにより，すべてのインスタンスを探索できる．従って，この探索によりインスタンスの各種解析を効率的に行なうことができる．

インスタンスの保持する情報は，属性と関係に分かれる．インスタンスの情報構成を，統合化Cプログラミングシステムの関数ソフトウェア構成要素を例にとって表1に示す．この表では，多くの情報項目を詳細設計書とソースプログラムの両方で定義していることが分かる．インスタンスは，このようにドキュメント間で共有する情報を一元的に保持する．従って，属性はドキュメントの書式に依存しない情報の断片である必要がある．すなわち，属性はドキュメント間で共有する情報が形式的に一致するレベルまで，各ドキュメントを論理的に細分化したものと考えることができる．

以上のソフトウェア構成要素の特性から，各々のドキュメントと，対応するすべてのインスタンスの情報を形式的に，常に一致するように管理することにより，ドキュメント間の整合性を保証することができる．この整合性の保証がTENSE OMSの基本的機能である．

このような管理には，ドキュメントの変更によって影響を受けるドキュメントを機械的に検出する機能が必要である．インスタンスのもつ関係情報は，このような解析に重要な役割を果たす．

関係情報は，ソフトウェア構成要素の相互関係で，TENSEでは，この関係の種類を4種類，部分，参照，記述，継承に絞る．部分関係，継承関係については既に述べた．参照関係はインスタンスの相互参照関係である．大域変数とその外部参照宣言を含むモジュール，手続きとそれを呼び出す手続きの関係が参照関係の例である．また，記述関係は，ドキュメントファイルのような物理情報と，その記述対象となるモジュールのような論理情報を示す関係である．

インスタンスに変更のあった場合は，部分，参照，記述関係を逆に辿ることにより，影響の及ぶドキュメントを検出することができる．なお，管理効率とメモリ効率上インスタンスは逆方向の関係情報をもたないため，逆方向の関係探索は，systemから始まり，すべてのインスタンスをノードとしてもつ部分関係木を利用する．

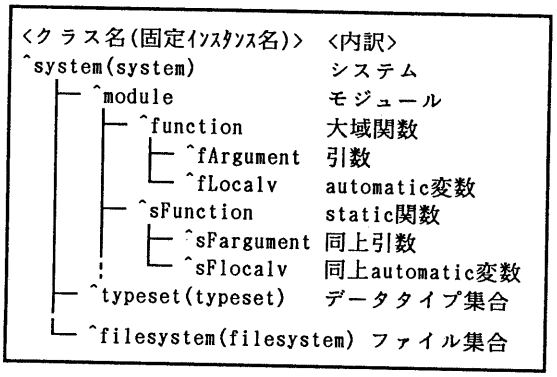

図1　ソフトウェア構成要素の部分関係

表1　関数ソフトウェア構成要素とドキュメント
(注 DD:詳細設計書，SP:ソースプログラム)

属性	定義部 DD	定義部 SP	関係	定義部 DD	定義部 SP
名称	○	○	継承:上位ｸﾗｽ	−	−
タイトル	○	○	部分:仮引数	○	○
説明	○	○	部分:auto.変数		○
データタイプ	○	○	参照:関数	○	○
戻り値	○	○	参照:変数	○	○
例外処理	○	○	参照:ﾏｸﾛ定義		○
注意	○	○	参照:関連関数	○	
アルゴリズム	○				
実行コード		○	○：定義有り		

２．２　整合性の保証

ドキュメント間の基本的な整合性の保証を司る処理をフィルタと呼ぶ．フィルタはドキュメントと，インスタンス情報を蓄積するTENSE データベースとのインタフェースである．従って，原則として，ドキュメントの種類毎に異なるフィルタを用意する必要がある．

フィルタの処理は，ドキュメントからインスタンス情報を抽出する処理，この抽出情報をデータベースに格納する処理，データベースの内容を再びドキュメント形式に逆生成する処理からなる．格納処理は関係情報に基づき，情報変更の影響解析や，影響を受けるドキュメントへの通知なども行なう．

フィルタによる整合処理の概要を図２に示す．この図では，まず，ドキュメントの抽出情報に基づいて，格納処理がデータベースを更新する．このとき，インスタンスの属性に変更があれば，それによって影響を受けるドキュメントを検出し，そのドキュメントに対して変更通知を出す．また，他モジュールの参照に誤りを発見した場合は，格納中のドキュメント自身に変更通知を出し，かつ，警告メッセージを出力する．

通知を受けたドキュメントは，次回の参照／編集の開始時点で逆生成処理を受ける．この逆生成処理の出力は，情報の変更を反映し，あるいは，参照の誤りを訂正したドキュメントとなる．

以上のように，フィルタを適当なタイミングで使用して，不完全な参照情報の記述の補完や，情報の変更の影響を受けるドキュメントへの反映を自動化できる．また，上流のドキュメントの格納後，下流ドキュメントの逆生成処理により，未記述のドキュメントを部分生成することもできる．

統合化Ｃプログラミングシステムでは，詳細設計書やソースプログラムの編集の直前直後にフィルタを自動的に起動する運用システムを用意してユーザの便宜を図っている．また，ドキュメントの書式からの逸脱を防止し，部分生成したドキュメントを空欄記述風に編集できる構造エディタ（フレームエディタ[6][11]）を提供して，開発作業の効率化を図っている．

２．３　拡張フレームとPrologによる実装

次に，OMSのカスタマイズ機能や，フィルタなどによるデータベースアクセスの応答性にとって重要な，ソフトウェア構成要素の表現形式について触れる．

TENSE OMSでは，ソフトウェア構成要素の表現モデルとして，フレーム表現に手続きをカプセル化して，オブジェクト指向の色合いを強めた拡張フレームを採用した．手続きには，メソドとメタメソドがある．

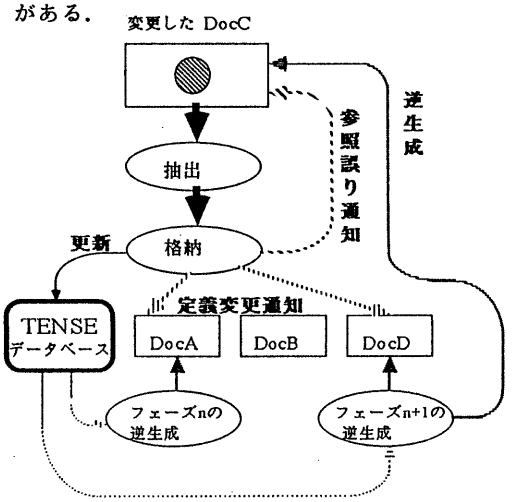

図２　フィルタによる整合処理の概要

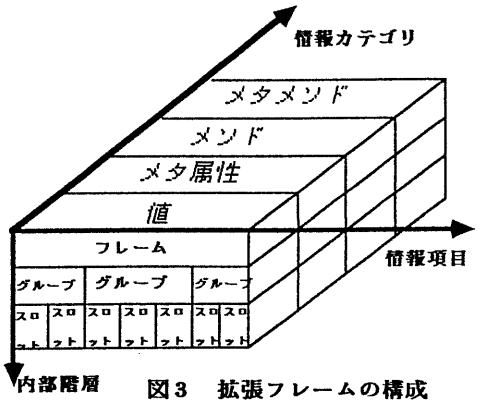

図３　拡張フレームの構成

フィルタの格納，逆生成処理はこのメソドで構成している．また，メタメソドとはメソドを生成するための手続きである．

拡張フレームのもう一つの特徴は，スロットのグループ化の採用である．スロットは表１の属性，関係の各情報項目にあたる．このスロットが多種類に及ぶ場合，あるいは，いくつかのスロットの一括処理の便宜上，スロットをグループに分けることができ，グループ共通の情報を一括保持することもできる．以上を図３にまとめる．

図中のメタ属性とは，そのクラスに属すインスタンスに共通な特性である．メタ属性は，データ独立性を確立するための情報で，メタメソドがクラス固有のメソドを生成する時に必要となる．メタ属性には，スロットやグループの構成，継承関係上位のクラス名などがある．ドキュメントの種類に応じた書式情報も各グループ毎に決まるメタ属性となる．

拡張フレームの実装にはPrologを採用した．Prologを選んだ理由は，ユニフィケーションによるデータアクセスの高速化と，プログラムの保守性の向上のためである．Prologによる実装方式としては，メモリ効率を上げ，応答性を更に向上させるために，一個の拡張フレームをアクセス単位と呼ぶ複数の事実節に分ける方式をとる．

アクセス単位の形式は値，メタ属性で次のように共通の構造となる(網掛部分は必要な場合のみ)．

ｿﾌﾄｳｪｱ構成要素名(上位ｸﾗｽ名,Gｸﾞﾙｰﾌﾟ名:ﾒﾀ属性名,ﾛｰｶﾙ識別情報:ｽﾛｯﾄ1/ﾒﾀ属性値,ｽﾛｯﾄ2...).

ここで，ソフトウェア構成要素名は，クラスではモジュール，変数といったクラス名，インスタンスでは，実際のモジュール名，変数名である．また，ローカル識別情報は，ソフトウェア構成要素名と上位クラス名でフレームが一意に決まらない場合必要になる情報である．Ｃ言語のautomatic変数では，その所属する関数の名称がローカル識別情報となる．

TENSE OMSのカスタマイズ機能は，以上のようなデータ独立性の高いデータ構造に基づいている．格納，逆生成処理は，各々のメタメソドによりクラス情報から，応答性の高いメソド集合として自動生成する．また，抽出処理の一部もクラス情報から自動生成する．従って，新たなドキュメントの組み込みは，その書式情報の定義と抽出処理の整備によって実現する．

新たなドキュメントがクラスの構成や特定のクラス情報の変更を要求するば場合は，クラス定義言語の書換えで対応することができる．この言語により，図１のようなクラス構成と全く異なる階層を定義することもできる．

これらのカスタマイズ機能により，OMSを開発支援環境の進化に追随させることができ，異なるソフトウェアプロセスに応じたOMSを容易に構築することができる．

なお，TENSE OMSの提供する支援機能の多くは，メタメソドとして開発してあるため，これらの機能は環境の進化や異なる様式の支援環境に対応して開発しなおす必要がない．

３．　時差解析

前節の整合性の保証の観点から離れ，本節では整合性の検査の観点から，高粒度のソフトウェア構成要素を活用した，時差解析機能を紹介する．また，この機能は汎用的なデータ処理の組合せで実現しており，これらの処理から得ることのできる他の機能についても触れる．

３．１　時差解析機能とは

設計やコードに対するウォークスルーやレビューの目的の一つは，上流フェーズの決定を下流フェ

ーズで正しく実施していることの検査である．この決定には，開発途上で発生する仕様や設計の変更も入っている．通常，この検査は，参加メンバがドキュメントを通読して，収集した情報に基づく議論で進行する．しかし，一般にドキュメントの量は膨大で，ドキュメント相互の食い違いも多いため，メンバの負担は大きく，検証結果も高い信頼性を期待できないことが多い．

時差解析とは，新旧2個のTENSEデータベースを比較して，両者の食い違いをレポートとして出力する機能である．従って，新旧のデータベースをコーディングフェーズ終了時のものと，設計フェーズ終了時に保存したものとすれば，出力レポートは上記の検査に必要な調査資料となる．

解析結果例として，レポートの先頭部分を図4に示す．この解析の特徴は，diffのような字面の食い違いでなく，項目別の食い違いを出力できる点である．また，注目すべき項目に絞ってレポートできる点である．更に，このレポートは，読みやすいtex形式をとる．

レビュー，ウォークスルーで，このレポートの内容をどのように判断するかは，検査メンバの責任であるが，調査の負担は大幅に削減する．また，整合性の保証機能により，ドキュメント相互の形式的な食い違いがないため，本質的な問題の検討に専念することができる．

このように，時差解析はOMSの特性を活かした，OMS独特の機能と考えることができる．

3．2　時差解析の処理方式と派生機能

時差解析の処理の流れのほとんどの部分は，図5のように汎用的なデータ処理の組合せである．この図は現在と過去のデータベースの時差解析の例である．

まず，旧データベースの内容を抽出形式に変換する．これには空のデータベースとの差分を採る形で，差分生成処理を使用する．次に，この抽出情報と現データベースから差分情報を生成する．この差分情報を，時差解析固有の処理である差分格納処理で，現データベースとマージして，差分付きデータベースを生成する．このデータベースは，現状のインスタンスに削除インスタンスを加え，各インスタンスの変更のあったスロットを，特殊な形式にして，新旧両情報を保持するものである．

以下に，関数引数arg1のデータタイプに変更のあった場合の差分付きインスタンス例を示す．

'arg1'('^fArgument','func1 ',....,'char ':'int ',...) ...下線部が変更のあったスロット

1　モジュール mmain の変更内容

- キャプション：メンバチェックメインモジュール
- mmain.c のディレクトリの変化は次のとおり．
  (新) /aux0/projects/togokac/tense/member/iwaoka/diff
  (旧) /aux0/projects/togokac/tense/member/iwaoka/icheck
- バージョンの変化は次のとおり．
  (新) 20 Wed Nov 16 14:07:03 1988
  (旧) 19 Wed Nov 16 08:51:39 1988
- キャプションの変化は次のとおり．
  (新) メンバチェックメインモジュール
  (旧) メンバチェック，及び，更新メインモジュール

1.1　関数 main の変更内容

- キャプション：メンバチェックメイン

1.1.1　引数 ghpst の変更内容

- キャプション：A:幽霊メンバ，B:通常メンバ
- データタイプの変化は次のとおり．
  (新) char _
  (旧) int _

1.2　追加関数 usage の概要

TENSE時差解析レポート: xxxx システム

1991年5月30日

対象となるデータベースファイルの名称，サイズ，最終変更日付は次のとおり．

新：xxx.idb xxxxxxxx bytes May 29
旧：xxx.idb oooooooo bytes Apr 1

目次

図4　時差解析レポート例

この形式は関係スロットにも適用する．従って，部分関係スロットから，追加，削除インスタンスを知ることができる．

最後の逆生成処理は，フィルタ処理を上記の特殊なスロット形式向けに拡張したものである．差分付きデータベースは，すべてのインスタンス，スロットに対する差分情報を含んでいるが，この逆生成段階で着目すべき差分のみレポートに出力する．なお，図５の処理はメソドの集合で，これらのメソドは対応するメタメソドで自動生成したものである．

この処理の流れは，途中までガーベジコレクション（GC）と，差分による版管理の流れと一致している．GCは部分関係木から外れたインスタンスを削除する機能で，版管理は差分情報から過去のデータベースを復元する機能である．

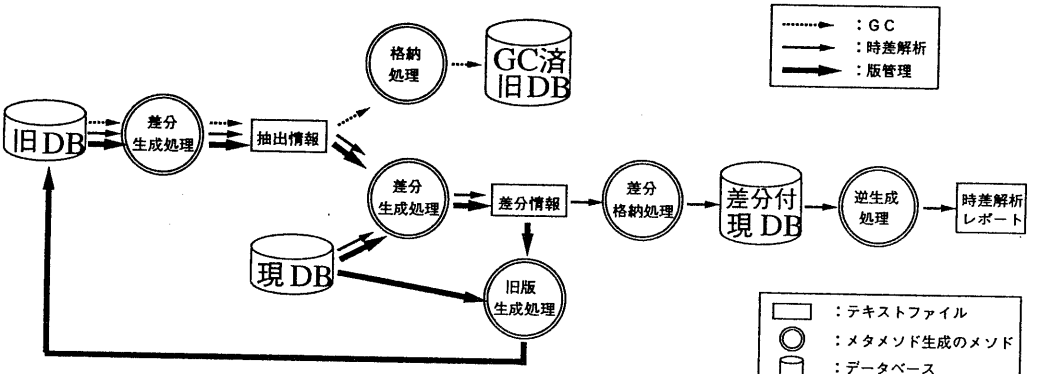

図５　時差解析処理と関連処理の流れ

４．まとめ

以上，TENSE OMSの実現方式と，特徴的機能の紹介を行なった．ここでは，最後にTENSE OMSの現状における機能の全容と，その効果の考察，及び，今後の課題について述べる．

まず，情報の整合性の保証や検査の他，TENSE OMSには表２のような機能がある．この表で，管理支

表２　TENSE OMSの機能概要

対象	分類	機能項目	概要	現状（○:運用可能,△:開発中,他:開発予定）
開発支援	整合性の保証	・ドキュメントの部分生成 ・参照記述の自動補完 ・変更の自動反映	他のフェーズの既存で，重複する情報からの未記述ドキュメントの部分生成． 他のモジュールの既存情報に対する，不完全な参照記述の完全化． 定義の変更等により影響を受けるドキュメントへの変更情報の反映．	○ ○ ○
	整合性の検査	・静的解析レポートの出力 ・時差解析レポートの出力	TENSEデータベースの静的解析結果の出力． 新旧TENSEデータベースの，注目すべき食い違いの検出とレポート化．	○ ○
	情報ｻｰﾋﾞｽ	・各種一覧表の生成 ・異なる書式のドキュメント生成 ・データベース問い合わせ（参照） ・データベース問い合わせ（書換）	ソフトウェア構成要素を分かりやすく配置した各種一覧表の生成． 必要に応じた，記述時と異なる書式のドキュメントの生成． TENSEデータベースへの直接の問い合わせによる，必要な情報の取り出し． TENSEデータベースの直接の書換えと，それに伴う変更の自動反映．	○ ○ △ △
	再利用	・ソフトウェアの部品化 ・ソフトウェア部品の変換	ソフトウェアの汎用性の高い部分の，独立な形での取り出し． 組み込み対象に合わせた，ソフトウェア部品の形式変換．	
管理支援	開発製品管理	・レヴィジョン番号管理 ・版管理	フェーズ間で対応するドキュメントのレヴィジョンの一元管理． 新旧TENSEデータベースの差分の保存による，過去の版の生成．	○ ○
	情報ｻｰﾋﾞｽ	・プロジェクト進捗度レポート出力 ・残作業ナビゲーション	各ドキュメントの完成度に基づく進捗度レポートの生成． 未完成のドキュメントの明細提示による，次にすべき作業の明確化．	○
環境維持	カスタマイズ	・多様なソフトウェアプロセス対応 ・開発支援環境の進化への対応	データ構造，管理処理の拡張支援や生成系により多様なドキュメント種類に対応． 上記機構，及び，既存データベース内容のデータ構造自動更新による対応．	○ ○

援機能は，TENSEデータベースの内容がドキュメントの相互関係を保持して，開発の進行につれて徐々に増加する点を利用している．また，開発，管理支援の情報サービスの多くは逆生成処理のカスタマイズによって実現している．

データベース問い合わせによる書換え処理では，インスタンス名称の変更のコードへの反映などを実現する．このような処理は，ドキュメント編集に連動した整合保証では実現しきれない．

これらの機能は，ソフトウェアの開発，管理の上で，次のような効果を発揮すると考えることができる．なお，この項目で，i～iiiは従来のウォータフォールモデルの欠点[5]を改善する効果である．

i) ドキュメントの部分生成や情報サービス機能などによって，フェーズやモジュールで共有する情報を，異なる形式で何度も記述する必要がなくなる．
ii) 変更の自動反映機能によって，下流から上流へのフィードバックの経路を実現できる．
iii) 他の作業との調和を損なわずに開発プロセスを拡張，改善でき，多様な開発プロセスにも柔軟に対応できる．
iv) TENSEデータベースに蓄積する情報を監視することにより，定量的なプロジェクト管理を行える．
v) ドキュメント情報の一元管理により，各種情報サービスや，版管理などの製品管理の自動化を実現できる．
vi) 高い抽象度でソフトウェアを蓄積することにより，汎用部分の部品化，再利用を行い易くなる．

しかし，TENSE OMSは基盤整備の終了した段階にあり，現在はその効果を引き出す端緒についたばかりの状況である．TENSE OMSが，現状では予想し得ない新しい効果を発揮する余地は大きい．再利用は今後の課題であり，オブジェクト指向やプロトタイピング環境での活用も検討する必要がある．

【参考文献】

[1] M.H.Penedo, etal."Object Management Issues for Software Engineering Environments -Workshop Report-" ACM SDE3 Proc., pp.226-234 (1988)

[2] R.N.Taylor, etal."Foundation for the Arcadia Environment Architecture" 同上, pp.1-13

[3] Y.Matsumoto, etal."A Data Model in the Software Project Databese KyotoDB" Advances in Software Science and Technology, vol.2, pp103-121(1990)

[4] R.W.Matthews, etal."Data modeling for software development" IBM SYSTEMS JOURNAL, Vol.29 NO.2, pp.228-235(1990)

[5] 有澤誠: ｿﾌﾄｳｪｱ工学, 岩波書店, (1988)

[6]原田賢一編 "構造ｴﾃﾞｨﾀ" 共立出版(1987)

[7] 西岡, 他 "統合開発支援環境TENSEのｿﾌﾄｳｪｱ ﾃﾞｰﾀﾍﾞｰｽ" 情報処理学会「ｱﾄﾞﾊﾞﾝｽﾄ・ﾃﾞｰﾀﾍﾞｰｽｼｽﾃﾑ」ｼﾝﾎﾟｼﾞｳﾑ'90 予稿集, pp33-42(1990)

[8]渡邊, 他 "TENSE:ｿﾌﾄｳｪｱﾃﾞｰﾀﾍﾞｰｽでﾌﾟﾛｸﾞﾗﾐﾝｸﾞの整合性を保障するC言語向けCASE環境", INAP'90「PROLOG産業 応用ｼﾝﾎﾟｼﾞｳﾑ」論文集, 6-4 (1990)

[9]西岡, 他"統合化Cﾌﾟﾛｸﾞﾗﾐﾝｸﾞｼｽﾃﾑの実現"情報処理学会「CASE環境」ｼﾝﾎﾟｼﾞｳﾑ予稿集, pp.81-88(1989)

[10]西岡, 他 "C言語ﾌﾟﾛｸﾞﾗﾑ情報の統合化" 情報処理学会 ｿﾌﾄｳｪｱ工学研究会資料(46-16) (1986)

[11]西岡, 他 "ｺｰﾃﾞｨﾝｸﾞｽﾀｲﾙを支援するｿｰｽﾌﾚｰﾑｴﾃﾞｨﾀのﾃｷｽﾄ変更方式" 情報処理学会第37回全国大会, 3K-3(1988)