ソフトウェア工学　85－1
（１９９２．５．２１）

エラーチェック仕様のデータモデル

米川　清 *　　会田邦夫 **

*　三井情報開発株式会社
** 川口短期大学

経済社会のダイナミズムへの対応を目差したトランザクション制御に関する「エラーチェック仕様のデータモデル」の試論を述べる．当モデルは、大規模ビジネスアプリケーションを対象とし、意思疎通に対応する「構文データ」と意味作用に対応する「意味データ」から構成されている．大手総合商社の会計システムに当モデルの適用を検討した結果、エラーチェック仕様変更の容易性、テスト環境の向上等の効果が期待でき、利用者のダイナミックな要求変更への対応に寄与することが推察できる．

Error check specification by Data Model

Kiyoshi Yonekawa * , Kunio Aida **

*　Mitsui Knowledge Industry Co.,Ltd
** Kawaguchi Junior College

Symptoms of a problem in conventional application development have been evident for a long time.　Almost every programmer has experienced cost and schedule overruns,　long debugging times,　and difficult and costly maintenance of programs.　Conventional programming concentrated on the program technique at a higher level, but it dose not yield useful generalized functions for data handling.　Therefore, we propose application development in Data Model.

1.はじめに

昨今、従来の関係データモデルでは動的変化への理論を欠く点が指摘されている(1). 現実世界そのものを構造化することは勿論不可能であるが、存在する諸構造を踏まえて、ダイナミックな運動を理論化する試みは必然であろう(2)(3)(4)(5).

上記文脈の中で、経営組織体の生産性向上を超越論的中心命題とする「ゼロ記号」としデータおよびそのエラーチェック仕様をメタデータとした大規模ビジネスアプリケーションの「エラーチェック仕様のデータモデル」の試論を述べる.

2.エラーチェック仕様のデータモデルの概念

従来のシステムでは、データ中心アプローチ(DOA) でも指摘されている通り(6)、データ項目の組合せやレコード間の組合せの整合性チェックは、プログラマーのアルゴリズム追求により支援された. その結果、開発期間の長期化を招き、更にメンテナビィリティィの悪化や仕様変更への柔軟性を欠き、利用者のダイナミックな情報生産性上のネックとなった

このような問題点を解消するトランザクション制御の「エラーチェック仕様のデータモデル」を図1に示す.

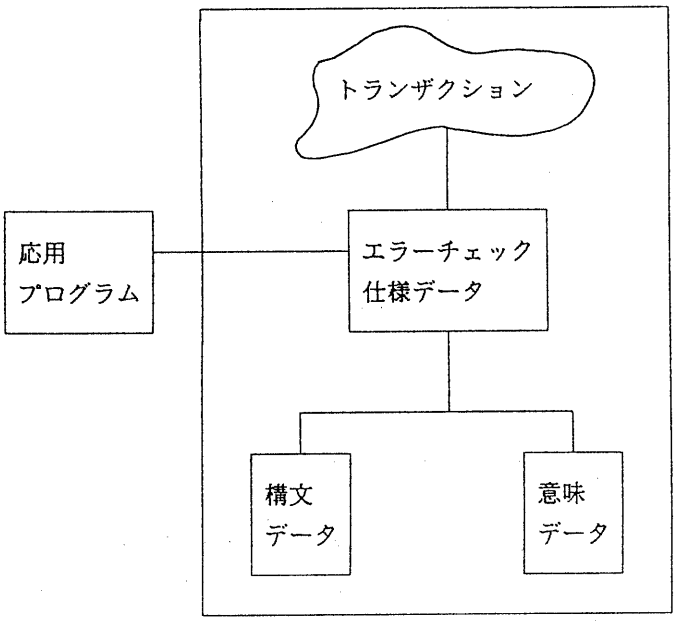

図1. エラーチェック仕様のデータモデル

構文データは、恒常的要素を組合せて同型的な構造を構築したものであり、構文データをメタレベルとし、意味データは異質な構造の異なる要素の錯綜体であり、オブジェクトレベルの下位とする. 両者のデータモデルの主要概念を表1に示す.

表１．データモデルの主要概念

主要概念	構文データモデル	意味データモデル
概念定義	母集団を特定し、データの相互関連や交換的要素を構造的に位置付ける．	異質構造の異質要素の錯綜体．
方法論	形式重視の集合論．	要素主義的で操作指向．
形状	木構造、ネットワーク構造等のように構造的．	分散的な側根．
設計仕様	ア・プリオリ（事前列挙）	ア・ポステリオリ（事後発生）
特性	静態モデル．対象世界とのコミュニケーションの文法を示す．	動態モデル．意味作用の解釈や現象（コード）の余剰価値を示す．

3.エラーチェック仕様のデータモデルの詳細

言語理論には、文を記号の列とみなしその構造を取り扱う構文論、単語、句、文等の意味およびそれらの指示物との関係を取り扱う意味論等の分野がある(7)．そこで、言語理論を援用して、対象世界とのコミュニケーション（意思疎通）に対応するデータと、意味作用に対応するデータに分類する．前者は、構文データモデルと定義し、後者は意味データモデルと定義する．

3.1 構文データ

システム設計の実際は、個別的かつ逐次的に行われ、汎用化システムを目標としながら処理の重複や意味表現の前段であるデータ名やコード体系は、不統一になりがちである．即ち、あるシステムでは取引先をコードで表現し、別のシステムでは略称で表現するという不具合を生ずる．

以上から、全てのシステムのデータ名を画一化し、統一した命名基準を設定する．また平仄を合わせ、利用者の社内用語、コード等のビジネスプロトコルの統一を図り、項目のデータ名（内部世界）とコード体系（外部世界）を一体化した構文データモデルを構築する．

(1)データ名のカテゴリー化

データ名を年月日を表現するグループ、数値を表現するグループ、営業項目を表現するグループ、会計項目を表現するグループ、システム内部で使用する項目のグループ等に分類整理する．具体例では、年月日を表現するグループをＡとすれば、発行年月日、決済期日、受渡年月日、入金日等に、Ａ１，Ａ２，Ａ３，Ａ４・・・と連番にして構文データへ登録する．

(2)コード体系の一元化

利用者組織のビジネスプロトコルの標準化を図り、部署コード、商品コード、取引先コード等を一元管理する．以上から、全てのシステムが参照するコード体系は一意となる．

3.2 意味データ

構文データに登録されたコード体系およびデータ名は、「記号表現・階層の規則」の文法を表現するのに対し、意味データは「記号内容および意味されるもの」を記号化している．構文データとのリンケージを堅持する必要性から、構文データを経由して意味データの登録を可能とする仕組みとする．また、データモデル間は、一意に対応しているわけではなく、両者の関係は多義的である．

(1)コード体系の意味論

コードの意味は、比較的安定している固定的内容と時間の経過と共に変化している変動的内容に分けられる．取引先コードであれば、固定的内容として、取引先の略称、名称、所在地等が登録され、変動的内容として、売上高、売掛金、受取手形等のポジションが提供される．即ち、コードの解釈や現象が登録され、レコードに発効、失効の日付を保持し世代管理を行う．

(2)データ名の意味論

全てのデータ名の意味を文章情報として保管する他、システムでエラーとなったデータは、後述のサブルーチン名からエラーコメントを出力する．このエラーコメントも文章情報化し、検索に役立てる．

4.エラーチェック仕様のデータモデルの適用への考察

大手総合商社の会計システムに当モデルの適用を検討した結果について、次に述べる．エラーチェックの内容を全てデータ化し、当モデルにより、トランザクションと構文データ、意味データの制御を行う．画面番号、ヘッダー項目・明細項目の区分を示す識別子をキー項目とし、データ名、チェック区分、サブルーチン名等を登録する．

4.1 キー項目

売上、入金、支払、外国為替等のように、入力データの性格を考慮して画面設計をし、画面番号は、入力画面と対応したものとする．識別子は、画面のヘッダー項目か、明細項目かの区分を示す．

4.2 属性項目

データ名毎に下記の属性項目を登録する．

(1)チェック区分

構成チェック、コードチェック、項目チェック、項目間（レコード間）チェック等を示す．

(2)ＡＮＴ区分

英字、数字、右詰、左詰、特殊文字等の項目チェック内容を示す．

(3)サブルーチン名

サブルーチン名は、データ名と関連付けて登録する．先述の発行年月日のデータ名をＡ１とするならば、下記の通り．

サブルーチン名Ａ１Ａ：発行年月日はカレンダー日付であること

サブルーチン名Ａ１Ｂ：発行年月日は決済期日と同一か、以前であること

応用プログラムでは、Ａ１Ａ，Ａ１Ｂのように想定されるチェックのパターンをコーディングしておく．また、当該チェックが不要となればサブルーチン名を削除し、新たなチェックが発生した際は、サブルーチン名を追加する．

4.3 トランザクション制御

図２のように、トランザクションはエラーチェック仕様データの制御により、先ず構文データを参照し、次に意味データを参照して、構成チェック、コードチェック、項目チェック、意味チェックおよび項目間（レコード間）チェックをする．意味の失効等により、invalid であれば、サブルーチン名からエラーコメントを出力する．

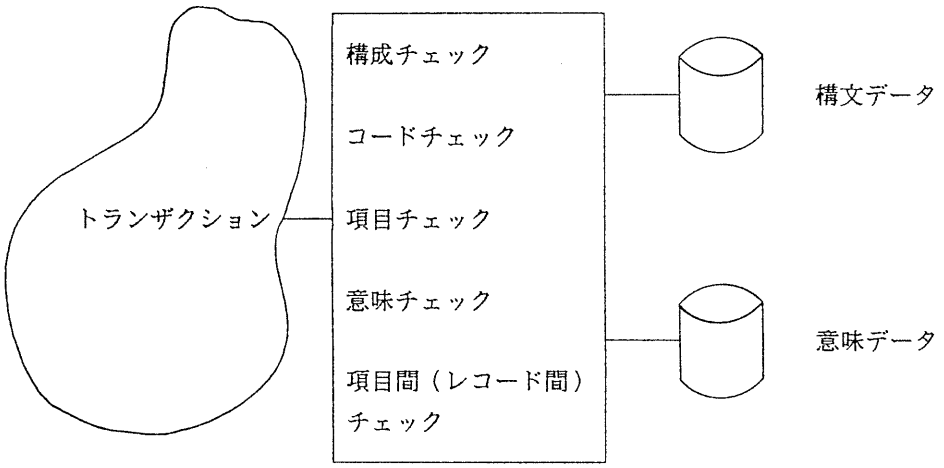

図２．エラーチェック仕様データによるトランザクション制御

4.4 当モデルの効果

⑴開発工程の短縮

従来のアプローチでは、詳細設計工程で作業量がスパイラル的に増大するが、当モデルは他の工程から独立しているので他の工程に拘束されない．即ち、画面設計、応用プログラム開発等とエラーチェック仕様が並行して開発される．また、画面単位の段階的なテストも可能となる．この結果、開発作業量が平準化し、開発工程の短縮が実現できる．

⑵仕様変更の容易性

既存のエラーチェックであれば、変更はエラーチェック仕様データの登録変更のみで可能となる．また、設計途上の仕様変更にも対応でき、特にコード追加の際も追加登録のみで可能となり、制御独立性が向上する．

⑶テスト環境の向上

仕様変更が発生の際、変更部分だけのサブルーチン名を登録した仮仕様のエラーチェック仕様データを作成し、他の項目は不要となる．テストデータは、ミニマムの項目だけでよい．

⑷妥当性のクラス概念の内部化が可能

具体例で示すと、構成チェックのパターン集合＝｛０、１、Ｎ，Ａ，＊｝とすれば、下記の通り．

０：存在を否認（エラー）

１：一意であれば可

Ｎ：１からＮ件でも可

Ａ：存在、不在は不問

＊：他の項目にエラーが検出されても容認

⑸利用者によるダイレクトコントロールの推進

コードの追加は利用者が追加登録することにより、エラーチェックのプログラムは修正不要となる．このようにして、利用者が直接エラーチェック仕様の一部を制御できる．

5.おわりに

当モデルを更にリファインし、利用者がエラーチェック仕様データを直接操作して、ダイナミックなビジネス活動により迅速に対応することが今後の課題である．即ち、大規模ビジネスアプリケーションにおいて、利用者のダイレクトなトランザクション制御を追求し、その維持の仕組み、セキュリティ等のフィージビリティスタディを更に進めたい．

参考文献

⑴田中克己、オブジェクト指向データベースの基礎概念、情報処理、Vol.32,No.5,1991

⑵レヴィ＝ストロース（大橋訳）、「野性の思考」、みすず書房

⑶ウンベルト・エーコ（池上訳）、「記号論」Ⅰ・Ⅱ、岩波現代選書

⑷ドウルーズ＝ガタリ（豊崎訳）、「リゾーム」，朝日出版社（'77.10臨時増刊）、1977

⑸浅田彰、「構造と力」　、けいそう書房、1983

⑹堀内一、「データ中心システム設計」、オーム社、1988

⑺栗原俊彦、吉田将、「言語理論」、共立出版、1970