リポジトリを用いたラウンドトリップ的設計作業支援の検討

3K－1

佐藤　友康　　外山　勝保
NTTソフトウェア研究所

1．はじめに

CASEツールの統合のためにリポジトリが用いられている[1]．ソフトウェア設計作業では，仕様変更が頻繁に生じ，これに伴い設計の修正作業が必要となる．しかし，現状ではリポジトリのデータをうまく修正できず，修正以前のデータが再利用できないという問題が生じている．仕様変更は避けられないものであり，変更に柔軟に対応できるリポジトリが必要である．そこで本稿は，変更に柔軟に対応するリポジトリについて述べる．

2．作業対象とその問題点

本稿では，リレーショナルデータベース(RDB)の概念設計と論理設計を対象とする．

・概念設計（ER図やDS図の作成）：現実の業務で管理あるいは処理される具体的な事物（実体）とそれらの関連を洗い出す．各実体のデータ項目（属性）を決定する．
・論理設計（カラム一覧表の作成）：概念設計を元に，テーブルやカラムを設計する．性能要件などの都合により，これらを統合，分割する．

これらの設計作業は，論理設計まで進んだ段階で，概念設計における誤りや不完全性が発見され，概念設計に戻り，修正が行われるような，概念設計と論理設計を行き来（ラウンドトリップ）しながら設計が進められるのが現実である．

このとき，ある仕様書の修正に対し，それを参照して作成される仕様書のどの部分を修正すれば良いか不明で，変更箇所がうまく抽出できず，変更漏れが生じたり，誤って不必要な変更が行われることがある．また，仕様変更が行われ，その内容を後に変更者とは別の設計者がこれを見ると，何が原因でどうなったかということを理解することができない．これは，以下の原因によるものと思われる．

・設計データ間の関係が不明確
・仕様変更の背景が不明確

Repository Schema for supporting round-trip design activities
Tomoyasu SATO, Katsuyasu TOYAMA
NTT Software Laboratories

3．データモデル

3．1　設計方針

このようなラウンドトリップ的作業における問題解決のために，設計作業において扱うデータとそれらの関係を明確化し，変更による波及範囲をできるだけ狭め，変更の背景を明らかにしておくと良い．そこで，以下のような方針を採用する．

(1)粒度を考慮した設計データ管理

データの関係の明確化のために，仕様書（ドキュメント）をその構成要素（オブジェクト）に分解し，オブジェクト間，ドキュメントーオブジェクト間，ドキュメント間で関連（依存関係，参照関係）を明確にする．さらにオブジェクト単位でデータを管理することで，変更影響波及範囲を狭める．

(2)オブジェクト単位での変更管理

(1)のオブジェクト単位の管理に，いつ，誰が，何を，どこで，どのように，なぜ(5W1H)変更したのかという，変更履歴を管理する．このことで設計者の変更作業における判断を促進する．

3．2　リポジトリスキーマ

各ドキュメントの構成要素とそれぞれの対応関係は図1の通りである．対応するオブジェクトは，リポジトリ上では通常は同一オブジェクトとして対応するドキュメントから共有される．

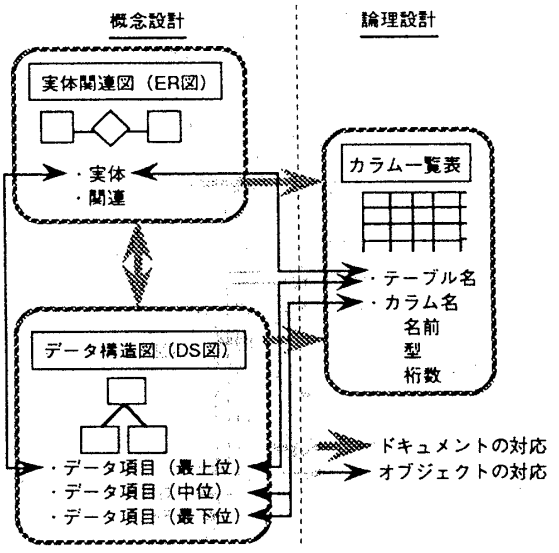

図1．ドキュメントとオブジェクトの対応

ERモデルで表現したデータモデルの基本構成を図2に示す．ここでは，図2上の実体をメタ実体と呼ぶ．関連が矢印であるものは存在依存性を示し，始点のメタ実体が存在しなければ終点のメタ実体は存在し得ないことを表す．その他の関連は相互参照関係である．

■粒度を考慮した設計データ管理

ドキュメント単位には概念設計のER図(ERD)，DS図(DSD)と論理設計のカラム一覧表(Table)を位置づける．概念設計が行われなければ論理設計が行われることはないため，カラム一覧表はER図とDS図に依存関係にある．

オブジェクト単位には，各仕様書の構成要素である，実体(Entity)，関連(Relationship)，データ項目(Data)を位置づける．関連とデータ項目は実体がなければ存在しない．また，各オブジェクトはドキュメントの構成要素であることを示すために，対応するドキュメントに依存関係がある．関連はER図の構成要素であるが，ERDからRelationshipへはEntityを通して参照できるため，冗長であると判断し，特に関連づけない．

先述のように，複数のドキュメントの構成要素であるオブジェクトは，各ドキュメントから共有される．これにより，1つのドキュメントでオブジェクトが変更されると，その変更が共有するドキュメントにも反映されることになり，ドキュメント間での矛盾防止やデータ管理の簡略化につながる．例えば，図1に示すようにデータ項目はDS図とカラム一覧表の構成要素であるから，DataはDSDとTableに依存関係にある．このように，設計で扱うデータをドキュメント単位とオブジェクト単位で粒度を考慮し，それぞれの関係を明確にして管理する．仮にDataに社員氏名を管理するカラムを作成するために“社員”というデータが入っており，DS図とカラム一覧表から共有されているとする．ここで“社員”だけでは名前なのか何なのか不明なため，データ項目を“社員名”とすることにする．この変更は，DS図とカラム一覧表に同時に反映されることになる．一般に，ある設計データが変更されるとその影響が，そのデータに存在依存するデータへは波及し，変更された設計データが存在依存する，あるいは参照関係にある設計データへは波及しない．これを利用して，変更による影響波及範囲を明確化する．

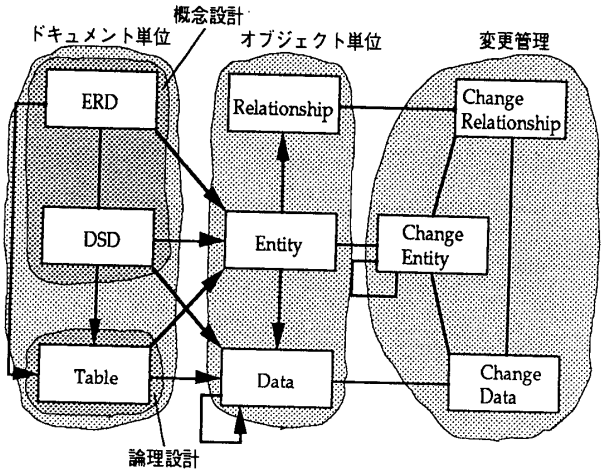

図2．リポジトリスキーマモデル

■オブジェクト単位での変更管理

図2の右側は各オブジェクトの変更管理を行うメタ実体(ChangeEntity，ChangeRelationship，ChangeData)であり，対応オブジェクトと相互参照関係にある．これらに各オブジェクトの変更履歴を管理する．各メタ実体は関連を持っており，ある変更によって波及した変更は，その波及元の変更と関連を持つ．これにより変更の影響波及も管理できる．

通常，リポジトリは変更による影響波及を設計者に提示し，設計者はそれらの情報を修正するか否か判断する．これを支援するため，先述の粒度を考慮した管理に加えて，設計者の判断材料を提供する機能を持たせる．また，ソフトウェア開発には同様の変更作業を行うことが多いため，変更履歴を過去の事例として利用し，それに習って変更を行う判断を促進することも可能である．

4．おわりに

本稿では，RDBの概念設計と論理設計をラウンドトリップする設計作業支援を目的に，リポジトリスキーマの設計について述べた．今後はこの評価を行い，さらに設計について検討を重ねる．今回の情報間の関係は2種類で表現することができた．対象とする範囲を広げ，リポジトリで扱うデータの種類も多くなると，関係の種類についてもさらに検討を加える必要がある．

謝辞

最後に，本研究の機会を与えていただいた石川リーダ，そしてご協力をいただいた多くの方々に感謝いたします．

参考文献

[1]R. Rock-Evans： “Repositories and Frameworks: a Detailed Product Evaluation”，Ovum，1993