5F-2

データベースセントリックに行うモデルベースの開発手法*

堀野　智久　　齋田　雄一　　岡野　信保　　遠藤　浩　　石川　貞裕 †
日立製作所　情報・通信グループ　生産技術本部 ‡

1．はじめに

完成された機能であるコンポーネントの再利用促進が叫ばれて久しいが、業務依存性が高い部品はカスタマイズに難があり、なかなか流通しない。他方、近年では Model Driven Architecture(MDA)と称される設計モデルをベースとするシステム開発手法とその再利用性に注目が集まっている。

日立製作所では「3 階層アーキテクチャ」「データベースセントリック」をキーワードとした開発方法を推進し、生産性の高い開発を実現している。特にデータベース製品の依存度が高い DB アクセス部分に専用のツールを利用し、モデルベースのシステム開発手法を確立してきている[1]。

本論文ではこのツールを中心に、ドメインモデルベースの開発手法について紹介する。

2．3階層アーキテクチャ

システム開発は、どのようにクラス設計をするかで作り込みに大きな差が生じる。優れた設計のもとに作られたプログラムは保守性に優れ、こうしたプログラムは堅牢性を備えている。

そこで弊社では、アプリケーションの設計にあたっては図 1のように3階層（プレゼンテーション層、ファンクション層、データベースアクセス層）を明確に分離し、実装・テストまでこの分類に従って開発を進めている。これによりビジネスロジックやアクセスパターンの洗い出し、洗練化による集約・共有が可能となる。また、システム全体を通じて各層を管理するクラスマスタを置き、無駄な重複開発を排除する。

開発はデータベースアクセス層（以下、D 層）から着手する。ここで D 層は、次の二点をもとに品質の高いプログラムを提供する。

(1) 専任担当者による D 層の洗練化
(2) 整備されたツール群による D 層の自動生成

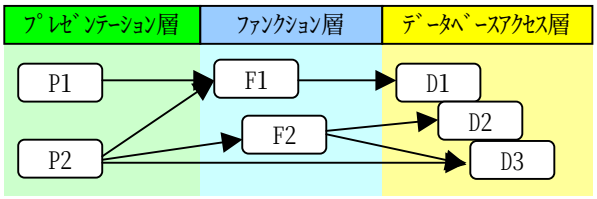

図 1：3階層アーキテクチャ

こうして作成した高い品質の D 層をベースにファクション層を開発することで、ビジネスロジックの品質を高めることができる。そして、ファンクション層の品質を確保した上でプレゼンテーション層を開発し、最終的にシステム全体において高い品質を実現する[2]。

3．データベースセントリックな開発について

3 階層アーキテクチャに基づく開発手法では、D 層をビジネスロジックから完全に分離し「データベースセントリック」に開発を進めていく。この理由としては、

(1) アプリケーションの内部ロジックの多くは DB に対する操作で占められている。このため明確に分離・共有をしないと、同じアクセス処理が複数箇所に実装される可能性がある。
(2) 開発プロジェクトの失敗要因には、DB 設計、操作に起因した性能問題等がある。そこで DB 部分を切り離すことで、障害発生時に迅速に調査、対策を実施できるようにする。

なお、D 層には専任の担当者を置き、メソッドマッピング時に抽出されたデータ項目とアクセスパターンについて検討を行う。こうして D 層の品質と効率的な開発を実現する。

4．データベースアクセス層の開発手法

D 層の開発について、作業の流れと主なツールの対応付けを図 2に示す。

D 層の設計では、まず業務内容等からデータ項目を洗い出す(①)。ここでは、データ項目を整理し、次にそのデータをもとに DB 設計を実施する(ER 図の作成 ②)。この DB 設計情報はプロジェクト全体で共有する。

次のステップでは、上記②で行った設計情報をもとに、SQL の定義を行う(③)。ここでは専用

---

* The model based development using database centric technique
† Tomohisa Horino, Yuichi Saida, Nobuyasu Okano, Hiroshi Endoh, Sadahiro Ishikawa
‡ Hitachi, Ltd. Information & Telecommunication Systems. Engineering Support Division.

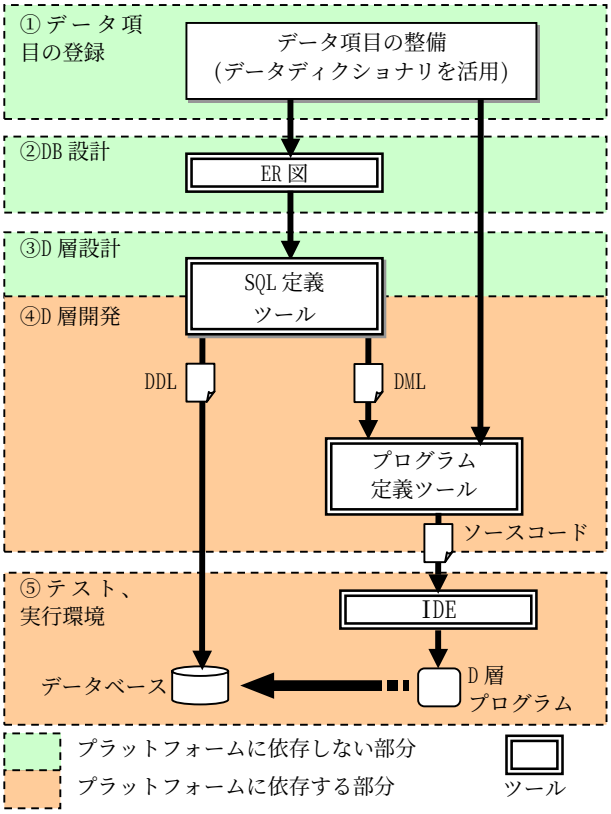

図 2：データベースアクセス層の開発の流れ

の UI(User Interface)を持つツールを使って定義を行う。このツールを、本論文では「SQL 定義ツール」と呼ぶ。SQL 定義ツールは、モデルベースの設計をもとに、プラットフォームに依存した DML 等の生成を行うツールである。さらに、他の専用のツール群と組み合わせて、D 層ソースコードの自動生成を行う(④)。こうして高い生産性も実現している。

さらに、自動生成した D 層プログラムと、SQL 定義ツールから生成した DDL を用いて、テスト・実行環境を構築することができる(⑤)。

以上のように、自社開発のツール群を活用し、モデルベースによる設計を行うことで、実装までを自動化する仕組みを実現している。

モデルベースの開発手法のメリットには、業務システムの仕様変更に素早く対応できる点や、別のプラットフォームに移行しやすい点がある。テクノロジに依存した部分を分離することで、技術変化の影響を局所化した開発を実現できる。

5．O/R マッピングフレームワークとの比較

一般に、図 2の③～④では「O/R マッピングフレームワーク」を使うことができる。多くの O/R マッピングフレームワークが、マッピング定義を記述したファイルから DDL やソースコードを自動生成する仕組みを持っている。そこで日立の開発手法と、オープンソースで提供されている代表的な O/R マッピングフレームワークとの比較を表 1で行った。

表 1：主な特徴の比較

#	比較項目	O/R	日立
1	複数 RDBMS 製品への対応	可能	可能
2	対応プログラム言語	Java のみ	複数
3	ソースコードの自動生成	可能	可能
4	SQL の編集	困難	可能
5	インターフェース	ｺﾏﾝﾄﾞﾍﾞｰｽ	画面
6	DB アクセス解析支援	―	あり

O/R：ｵｰﾌﾟﾝｿｰｽの O/R ﾏｯﾋﾟﾝｸﾞﾌﾚｰﾑﾜｰｸ、日立：日立の開発手法

O/R マッピングフレームワークは、発行する SQL をフレームワークが隠蔽するため、問題発生時の切り分けやチューニングが困難になるデメリットがある。一方、弊社の SQL 定義ツールは SQL の編集が可能であり、DB の特徴や実行環境の制約、データ分布の関係などで、どうしても SQL 文をユーザが書き換えなくてはならないプロ仕様のニーズも満たすことが可能である。

また、SQL 定義ツールは専用の UI で作業が行えるため、操作性の面でも優れている。

以上から SQL 定義ツールは、モデルベースの開発や生産性の向上だけでなく、D 層の品質確保と効率的な開発を狙うことも可能としている。

6．まとめと今後の課題

今回、データベースセントリックに行うモデルベースの開発手法を適用することにより、効率的で保守性の優れたシステム開発を実現する施策とノウハウをシステム開発者へ展開できたと評価している。

弊社生産技術部門ではこれまでも、ここに記載した開発プロセスを各プロジェクトに適用、支援する活動を行ってきた。今後も各プロジェクトでの実績を評価し、手法の改善、ツールの機能強化を図っていく。

参考文献
[1] 石川貞裕、他『企業システムの構造改革を加速するアプリケーションアーキテクチャ』，日立評論 2004 年 6 月号
[2] 斉藤岳、他『DOA を取り入れたコンポーネント指向開発手法』，FIT2003.
[3] 山村喜恒、他『データディクショナリ共用による効率的な多言語プログラム自動生成手法』，FIT2004.