ITジャーナリストや現役書店員、編集者が選ぶ デジタル人材のための Book Review ブックレビュー

今月の書籍

レビュワー:高橋 征義

- 『エリック・エヴァンスのドメイン駆動設計』
- 『エンタープライズアプリケーションアーキテクチャパターン』

理想的なソフトウェア開発の理想と実践法とを示した DDDの核心をついた、読み継がれる良書

ソフトウェア開発において、使いやすいクラスやコンポーネントを設計するのは難しい。とりわけ大規模になると複雑怪奇になりやすい。そこで先人の良い知恵はないかと探してみると、「ドメイン駆動設計(Domain-driven design, DDD)」というソフトウェア分析・設計・開発技法があり、長きにわたって広く支持されていることがわかる。

本書は、このDDDに関する原典であり、原点とも言える。

本の詳細はこちら
（外部サイト）

DDDはある程度知っていたり、実際に使っていたりしても、本書を読んだことのない方も少なくないだろう。しかし、残念ながらそのような方には本書が語るDDDのエッセンスが伝わっていないことが多いのではないか、という懸念がある。

DDDにはさまざまな側面がある。現在は「第2部モデル駆動設計の構成要素」で紹介される一種のアーキテクチャ技法としての側面が広く知られている。だが、実はDDDの核心はそこにはない。

本書が書かれたのは2004年。当時はオブジェクト指向分析・設計の華やかなりし頃で、分析者（アナリスト）、モデラと呼ばれる技術者が作る「分析モデル」と、実際のソフトウェアの設計や実装にはギャップがあった。設計時の成果物として、分析モデルとは異なる「設計モデル」なるものが作られることもあったほどだ。

それを解決すべく、分析モデルと設計モデルを区別しないで、唯一のモデルで完結させ、それを設計・実装と結びつけさせる（「モデル駆動開発」）。さらに、そのモデルも、ドメインエキスパート（ソフトウェアの対象となる分野の専門家）が理解できるように、ドメインエキスパートの使う言葉と同じ用語で表現できるようにする（「ユビキタス言語」）。また、ドメインエキスパートと対峙しモデルを作る人が設計・実装も行う（「実践的モデラ」）。

もっとも、ドメインに詳しくない技術者と、ソフトウェアに詳しくないドメインエキスパート同士ではいきなり良いモデル、良い設計を行うことは難しい。そこで両者でソフトウェアに向かい合いつつ対話を行い、ドメインへの理解を深め、柔軟性の高い設計技法(「しなやかな設計」)のもと**「より深い洞察へ向かうリファクタリング」を繰り返す**ことでモデルを洗練させながら、「深いモデル」を見つけ出す――そのような一連の流れが**本書で示すDDD**である。

「集約」や「値オブジェクト」、「リポジトリ」といった実装寄りの概念とは異なる、著者の考える**ソフトウェア開発の理想像とその実践方法**を知りたい人には、ぜひ一度本書を紐解いてほしい。

> **『エリック・エヴァンスのドメイン駆動設計』**
>
> 著者:エリック・エヴァンス
> 監修:今関 剛
> 翻訳:今関 剛、牧野祐子、和智右桂
> 出版社:翔泳社
> https://www.shoeisha.co.jp/book/detail/9784798126708

ソフトウェアアーキテクチャの基礎を振り返り ボキャブラリーを定義するために、今も変わらず有用な古典

エンタープライズアプリケーションのパターンを扱った本書の原著は2003年に出版された。DDD本とほぼ同年代の書籍である。とはいえDDD本に比べ、2021年現在では広く読まれているとは言い難い。良くも悪くも「古典(classic)」の扱いになってしまったのだ、と感じられる。

本書を**今改めて読むべき意義**は2つある。1つは**現在のソフトウェアアーキテクチャの基礎**となっていること、もう1つは**設計に使えるボキャブラリーを提供**してくれることだ。

本の詳細はこちら
(外部サイト)

本書のいう「エンタープライズアプリケーション」というのはいわゆる「エンタープライズ」とは直接の関係はなく、かなり広い意味を持つ。DB等の永続的なデータストアがあるWebアプリやスマホアプリは、すべて本書で言うエンタープライズアプリケーションになる。そこではドメインロジックパターン、データソースのアーキテクチャに関するパターン、Object-Relationalについての各種パターンやロックについてのパターンなど、**現代でも本書の概念がそのまま使える**場面が多い。

また、後者の役割も重要である。第18章のベースパターンのようなものは、**オブジェクトの関係性を示すパターンとして重宝**するものだが、「ゲートウェイ」「マッパー」「レジストリ」など、一般的な言葉として用語の揺れも避け難い。そのようなときに広く参照できる文献として、本書での定義を基本に論じることができるのは大変有り難い。

一方で、本書の弱点は、現代的アーキテクチャであるクラウドやコンテナの普及による分散、マイクロサービス化に関して、役立つ知見をあまり期待できないところだ。分散については第15章で扱っているとはいえ、「リモートファサード」と「データ変換オブジェクト」のみで、これだけでは心もとない。マイクロサービスについては『マイクロサービスパターン』(クリス・リチャードソン著)(https://book.impress.co.jp/books/1118101063)、分散アプリケーションについては『分散システムデザインパターン』(ブレンダン・バーンズ著)(https://www.oreilly.co.jp/books/9784873118758/)も参考にしたい。

もう一つ、訳語が微妙なところも困るところではある。著者のサイト「Martin Fowler's Bliki」の日本語版サイト「Bliki (ja)」にあるパターンカタログ(日本語)(https://bliki-ja.github.io/pofeaa/CatalogOfPofEAA_Ja/)も併せて参照することをすすめたい。

> 『エンタープライズアプリケーションアーキテクチャパターン』
>
> 著者:マーチン・ファウラー
> 監修/翻訳:長瀬嘉秀
> 翻訳:株式会社テクノロジックアート
> 出版社:翔泳社
> https://www.shoeisha.co.jp/book/detail/9784798105536

今月のレビュワー

高橋征義(たかはし・まさよし)

札幌出身。Web制作会社にてプログラマとして勤務する傍ら、2004年にRubyの開発者と利用者を支援する団体、日本Rubyの会を設立、現在まで代表を務める。2010年にITエンジニア向けの技術系電子書籍の制作と販売を行う株式会社達人出版会を設立、現在まで代表取締役。著書に『たのしいRuby』(共著)など。好きな作家は新井素子。