未来に向かって

6 開かれたソフトウェアのモデリング

基 専

野田夏子（芝浦工業大学） 岸 知二（早稲田大学）

ソフトウェアモデリング＝UMLの図を描くこと？

ソフトウェア工学のコンテキストでモデリング（ソフトウェアモデリングのこと，本稿では以降モデリングと略記）と言えば，まずはUML（Unified Modeling Language，オブジェクト指向をベースにしたモデリングを行うための標準的な言語）の図を描くことを思い浮かべる人が多いかもしれない．そして，UMLとは少し古い話題だな，とも思われるだろうか．

確かに，UMLの研究や現場導入の議論が活発になされるという時代は過ぎたようだ．ソフトウェア工学のトップカンファレンスICSE（本特集の1，4を参照）でも，10年ほど前にはUMLをソフトウェア開発の中でどのように使うかをテーマにセッションが立てられるなど，UMLに関する研究が複数見受けられた．しかし最近ではそうしたこともない．UML自身は，さまざまなドメインで用いられるさまざまな図（ダイアグラム）を包含するようになり，改訂を重ねながら発展を遂げてきた．その結果，仕様書は膨大なページ数になり，分かりやすく単純化することが求められ，2015年6月にそれまでの2分冊構成を改め1冊に統一したUML2.5が発行された．仕様書の構成としてはシンプルになったが，図式言語としての記法や意味に大きな変化はなく，UMLはおおむね安定したように思われる（その後の改訂も，本原稿執筆時点ではない）．

表-1はモデリング技術やモデル活用方法の変遷を示すものである．UMLそのものの研究は落ち着いてきたが，それを活用する研究はさまざまな広がりを見せていることが分かる．このようにモデリングの重要性は一層増していると言える．

モデリング＝対象の抽象化

モデリングとは何か．それは対象を抽象化し，形式性のある記法で記述することである．

現実世界で使われるソフトウェアは実に大規模で複雑だ．（独）情報処理推進機構（IPA）の最近の調査によれば，ソフトウェアの総行数は，エンタプライズ系で平均約17万行，組込み系では平均約38万行．100万行を超えるものも珍しくはなく，中には1,000万行を超えるものもある[1), 2)]．このような大規模なものをそのままの形で理解することはもはや不可能であり，特定の側面ごとに抽象化してコンパクトに表現することが不可欠である．

その表現形式として標準化されたものの1つがUMLである．前述のように，その発展の中でさまざまな図を包含することになったので，現在ではさまざまな対象をUMLで記述できる．また，UMLはそれ自身に拡張して使うための機構を含むため，目的に

時期	1970年代～	1980年代後半～	2000年代～	2010年代～
IT	メインフレーム	PC/NW	モバイル	IoT
モデリング技術	（方法論） データフロー図 実体関連図 状態遷移図	（モデリング支援） オブジェクト指向モデリング／アスペクト指向モデリング 記法統一（UML） フィーチャモデル ゴールモデル	（自動化） メタモデル 変換定義（QVT）	（SDx時代への適用） システム記述 非機能モデリング
モデル活用	手動 ドキュメント	記述支援ツール	モデル駆動 形式手法	統計 AI・探索

表-1 モデリング技術の変遷

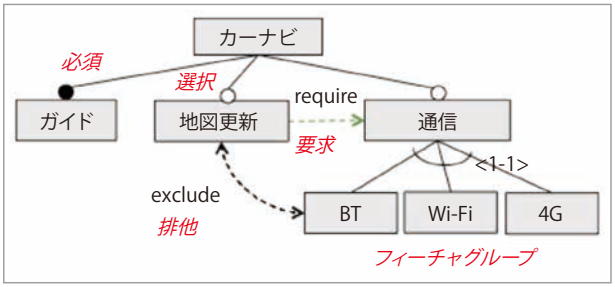

図 -1　フィーチャモデルの例

合わせてカスタマイズして使うことができる．だからといって UML が唯一の表現形式ではなく，ある目的に対してより素直な，あるいは分かりやすい表現があればそれらを使うことも可能だ．一例として，ソフトウェアプロダクトライン開発で用いられるフィーチャモデルの例を**図 -1** に示す．フィーチャモデルは，ある製品群が共通に持つ特徴（フィーチャ），製品ごとに異なり得る特徴を表現するものである．図 -1 ではすべてのカーナビにはガイドが備わっているが，地図更新や通信は一部のカーナビだけにあることなどを示している．

さて，対象を抽象化する，と書いたが，対象とは何か．もちろん一義的にはソフトウェアであるが，ソフトウェア（により実現されるシステム）によって解決したい問題を表現することもあれば，問題の解決策としてのソフトウェアを表現することもあるので，その区別は重要だ．典型的には，前者は要求定義におけるモデリングであり，後者は設計におけるモデリングということになる．

さらに同じ対象を抽象化するのであっても，ものごとの切り取り方，つまり視点は複数存在する．大きな軸として，静的な構造を見ているのか，動的な構造を見ているのか，という分類がある．また構造の一例を例示するためのモデルと，構造を一般化して示すためのモデルがある．たとえば UML のクラス図は一般化した構造を示すものであり，オブジェクト図は構造の例示である．

なお，一般的には，図法は対象や視点と一対一に対応するものではない．たとえば，クラス図は解決したい問題の構造を表現することも，解決策としてのソフトウェアの構造を表現することもできる．

これからのモデリングの課題

現在すでに，さまざまなものがソフトウェアで定義されるようになってきている．従来はソフトウェアの外側にあるものと捉えられてきたストレージやネットワークまでもが，ソフトウェア的な見方で捉えられコントロールされるようになってきた．また，複数の独立したシステムを接続して 1 つのシステムとするシステム・オブ・システムズも増えている．従来対象にしていたのは，いわば「閉じた」ソフトウェアであったものが，「開かれた」ソフトウェアになってきているのである．このような変化の中で，モデリングの性格や要件も多様化してきている．重要な課題はいくつもあるが，その中から特に 3 つの点を取り上げる．

➡ システム記述

コントロールの対象がハードウェアに広がり，ほかのシステムとも接続されるようになると，モデリングの対象もこれらハードウェアやほかのシステムを含むことになる．したがって，これからのモデリングはシステム記述としての性格がより強くなってくると考えられる．

ソフトウェアプロダクトライン開発を例にとると，ソフトウェアだけではなくシステム全体のプロダクトラインを対象とする研究や実践例がここ数年増えている．前述したフィーチャを単位にシステムの製品系列を扱おうとするフィーチャベースの開発手法が，システムエンジニアリングの業界団体から提案されるなどの動きもある．このようなシステム全体のプロダクトラインの開発では，フィーチャモデルも，用いるハードウェアやその特性を含めて記述されるようになってきている．

➡ 非機能特性

これまでも，ソフトウェア開発において非機能特性への考慮は非常に重要であった．今後，コントロールの対象がハードウェアを含めたさまざまなものに広がるにつれ，システム全体の振舞いはリアルな世界とのかかわりがより強くなり，非機能面の扱いがさらに重要となる．モデリングにおいては，従来はややもすると補足情報といった形で扱われることもあったが，モ

デルの一部として明示化し解析することが重要となる．UML を用いたモデリングにおいて非機能特性を記述できるプロファイルもすでに存在するし[3]，フィーチャモデル上に非機能特性情報を付加し製品の非機能特性を解析する技術なども研究されている．

➡ 規模の爆発

システムのすべてがソフトウェア化し，多くのシステムが統合されたシステムを扱わなければならないという変化の中で，モデリングの対象の規模が爆発的に大きくなっている．たとえば，先に紹介したフィーチャモデルでは，1つのプロダクトラインに含まれるフィーチャの数は今や数千に上るものも珍しくなく，数万にもなるものもある．

対象が大規模化しているのだから，単純にモデルだけ小さくしたところで意味はないが，かといってモデルの規模が大きすぎればその活用も難しくなる．

こうした問題に対して，たとえばスライシング技術をモデルに適用し，注目する部分だけ抜き出すといったことが行われている．フィーチャモデルに対しても，さまざまなスライシングのアルゴリズムが提案されている[4]．また，フィーチャモデルは製品群に含まれる製品の導出に活用されるが，フィーチャの数が膨大になると人手で正しい製品の導出を行うことが難しくなる．これに対して，探索技術を用いて効率的に妥当な製品の導出を行うといったことも研究されている[5]．

また，さまざまなものをソフトウェアとして扱ったり，さまざまなシステムを接続したシステムを考えたりする場合には，規模が大きくなっているだけでなく，規模が実は分からない，つまりどこまでがモデル化の範囲なのかその境界が曖昧になるということも起こってくる．このような状況にあっては，常に厳密で正しいモデルを定義・維持することはもはや困難である．重要な部分は厳密にモデル化するが，重要度が低い部分，あるいは検討を後に遅らせても当面問題がなさそうな部分に関しては，ある程度の不確かさを許容するといったことも必要になってきており，研究も始まっているところである．

モデリングの本質は変わらない

上記3つの観点から述べたが，これらがすべてであるということではない．本特集でも，要求工学，ソフトウェア検証等，ソフトウェア開発のさまざまなプロセスの動向が紹介されているが，それぞれにおいてモデリングは必要であり，また特有の問題を包含する．また，ゲームのソフトウェアなのか，車載システムなのかといったドメインごとにも特有の課題がある．それらを網羅することは不可能なので，比較的共通する3つを選んで紹介した．なお，最近のモデリングやモデル活用の動向でフィーチャモデルに関するもの多くを取り上げているが，ほかのモデル（記法）についても同様の研究はある．同じモデルに対しての広がりを感じていただくためにあえてフィーチャモデルを中心に論じた．

さて，さまざまな課題を見てきたが，モデリングが抽象化であるという本質は変わらない．むしろ本稿で述べたようなモデリングの対象範囲や規模の拡大，機械処理を想定した活用の多様化などの状況の中，より明確な視点を持った合目的なモデリングをすることが一層重要となる．そういう意味で，今こそ本質を踏まえたモデリングが重要となっているといえる．

参考文献

1) ソフトウェア開発データ白書 2016-2017,（独）情報処理推進機構.
2) 組込みソフトウェア開発データ白書 2015,（独）情報処理推進機構.
3) UML Profile for MARTE : Modeling and Analysis of Real-Time Embedded Systems (2011), http://www.omg.org/spec/MARTE/1.1
4) Krieter, S., et.al : Comparing Algorithms for Efficient Feature-Model Slicing, 20th Int. Systems and Software Product Line Conference (2016).
5) Sayyad, A. S., et.al : Scalable Product Line Configuration : A Straw to Break the Camel's Back, 28th IEEE/ACM Int. Conference on Automated Software Engineering (2013).

（2017年5月15日受付）

野田夏子（正会員） nnoda@shibaura-it.ac.jp

東京女子大学大学院理学研究科数学専攻修了．2008年北陸先端科学技術大学院大学博士後期課程修了．博士（情報科学）．日本電気（株）を経て，2013年より芝浦工業大学准教授．

岸　知二（正会員） kishi@waseda.jp

1982年京都大学工学研究科情報工学専攻修了．2002年北陸先端科学技術大学院大学博士後期課程修了．博士（情報科学）．日本電気（株），北陸先端大学を経て，2009年より早稲田大学教授．