解説：

メタモデル標準化の意義と最新動向

Current Trends on Metamodel Standardization :

後編：－MOF，XMI仕様と応用－

Part-2 : MOF, XMI Specification and Application

堀内　一　東京国際大学　hori@tiu.ac.jp
大林正晴　（株）管理工学研究所　obayashi@kthree.co.jp
藤川泰之　富士通（株）　fujikawa.yasuyu@jp.fujitsu.com

前編では，メタモデル標準化の意義と最新動向について，メタモデルの基本的概念を紹介してから標準化動向を紹介し，今後，情報技術分野においてメタモデルが重要性を持つことなどを解説した．

本編では，モデリング言語の意味的側面を規定する「メタモデル」と，メタモデルを記述するための仕様である MOF（Meta Object Facility），および，メタモデルを交換する標準仕様である XMI（XML Metadata Interchange）について解説する．

MOF

◆ MOF の沿革

MOF は，OMG（Object Management Group）[1] で標準化されたメタモデル記述仕様である．1997 年に MOF V1.0 が発行され，現在の最新版は MOF V1.4（2002 年 4 月）である [2]．

MOF 仕様は，相互運用可能なメタモデル，およびそのメタモデルに対応するモデルを定義し，操作するための枠組みを，ODP IDL[8]（CORBA IDL）準拠のインタフェースとして規定したものである．

MOF で規定される相互運用可能なメタモデルの大半は，UML メタモデルの一部と共通する．MOF 仕様では，MOF メタメタモデルも規定し，メタモデルで定義される他の標準技術の設計，および再利用のためのリポジトリを実装するための基盤を提供する．

また，MOF 仕様では，メタモデルから CORBA インタフェースを自動生成するための厳密なマッピング規則を与えている．これは，分散アプリケーション開発の場面で，一貫性を維持して，メタデータを操作するような環境の提供を意図したものである．

実際，MOF 仕様開発では，アーキテクチャ上の調整に，かなりの努力が注がれた．特に，UML と MOF の関係は，その意味的コアの部分で，同じ意味のオブジェクトの共有化が図られた．この連携で，MOF のメタモデルを視覚化するのに，UML 記法を利用することが可能となった．しかし，意味的な違いを必要とする部分も存在し，それらの違いは，メタモデル間のマッピング規則として与えてある．

◆メタデータアーキテクチャ

前編でも述べたように，MOF は，ISO の IRDS や CDIF と同様の 4 階層のメタ階層概念を基本にした MOF メタデータアーキテクチャに従っている．その特徴は，情報層を支配するモデル層，モデル層を支配するメタモデル層，そして，メタモデル層を支配するメタメタモデル層を持ち，異なる種類のメタデータを統一的に管理し，拡張性のあるメタデータ管理機構を提供していることである．

ここで，用語“メタデータ”は，“データ”に関するデータ，“メタメタデータ”は，“メタデータ”に関するデータを指す．また，以下のように“モデル”は，“メタデータ”を集約したもの，“メタモデル”は，“メタメタデータ”を集約したものである（メタメタメタデータ，メタメタモデルも同様である）．

- 情報層（M0）は，記述したいデータから構成される．
- モデル層（M1）は，情報層のデータのことを記述するメタデータから構成される．メタデータは，モデル中に集約されて存在する．
- メタモデル層（M2）は，モデル層のメタデータの構造と意味の記述（つまり，メタメタデータ）から構成される．メタメタデータは，メタモデル中に集約され

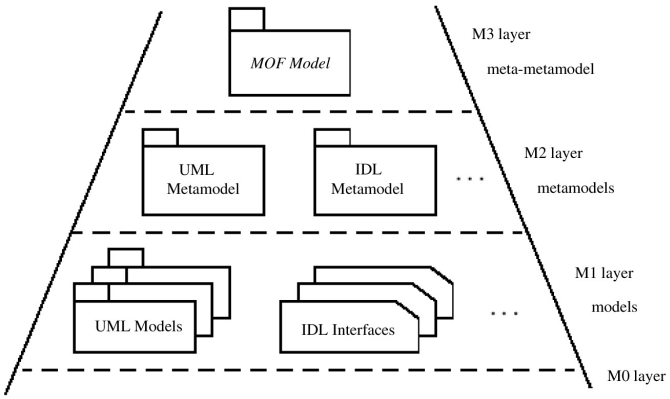

図 -1　MOF メタデータアーキテクチャ [2)]

て存在する．メタモデルは，種々のデータを記述するための言語モデルであり，具体的な構文や記法を持たない抽象言語を規定する．

- メタメタモデル層（M3）は，メタメタデータの構造や意味の記述（つまり，メタメタメタデータ）から構成される．メタメタメタデータは，メタメタモデル中に集約されて存在する．一方，メタメタモデルは，種々のメタデータを定義するための抽象言語を規定する．

図 -1 は，典型的な MOF メタデータアーキテクチャの例であり，UML 図と OMG IDL を表現するためのメタモデルが例として挙がっている．

実は，現在の MOF メタデータアーキテクチャは，上述の初期のアーキテクチャとは異なっている．従来は，固定的に 4 つのメタレベルを考えていたが，より柔軟に MOF 仕様を，解釈，運用するため，MOF メタデータアーキテクチャが見直された．

たとえば，UML での Class は，UML メタモデルの中のクラス "Class" のインスタンスによって記述される．次に，それは，MOF モデルでのクラス "Class" のインスタンスによって記述される．最後に，MOF モデルでのクラス "Class" は，それ自身で記述される．このように概念と名前の複数のレベルに渡る重なりが混乱をまねく．これは，特に，前述のようにメタ - メタやメタ - メタ - メタのような修飾子で区別されることが多いが，よく慣れた人でも，メタ - メタ - メタは会話で用いるには厄介である．

このような混乱を避けるため，MOF では，一般的に，メタ - 修飾子の使用を避け，MOF のコア部は，（4 つのメタ層があると仮定して）メタ - メタモデルであるが，単に“MOF モデル”と呼ぶことにしている．また，用語“メタモデル”は，“メタデータ”のモデルのことと定義されている．

これにより，MOF モデルを，M3 以外のレベルにも配置して考えることができるようになる．どのレベルに配置されるかに依存して，メタかメタ - メタかなどの修飾子は異なってくることになる．つまり，MOF メタレベルは，異なる種類のデータとメタデータの間の関係を理解するための純粋に便宜的なものである．

MOF モデルは，自己記述的である，言い換えれば，MOF モデルは，形式的には，自分自身のメタモデリング構成要素を用いて定義される．このことは，MOF モデルが，実際的なメタモデリングを表現するのに十分であることを示している．

◆構成

実際の MOF 仕様は，以下の内容で構成されている．

- MOF メタメタモデルの形式的な定義，つまり，MOF メタモデルを定義するための抽象言語定義
- 任意の MOF メタモデルから，メタデータを管理するための IDL インタフェースを生成する CORBA IDL へのマッピング
- メタモデルのメタデータ非依存性を管理するための自己反映的な CORBA IDL インタフェースの集合
- MOF メタモデルを表現し，管理するための CORBA IDL インタフェースの集合
- MOF メタモデルの交換のための XMI 形式

◆応用シナリオ

MOF は，モデル作成を通じて，広く使用されるパターンや共通のアプリケーション作成の支援を意図している．MOF の応用シナリオを考える場合には，まず第 1 に，MOF に対して 2 つの異なる視点があることを理解する必要がある．

モデリングの視点

設計者の視点である．モデリングの視点からメタレベルを見る．MOF は，関心がある特定ドメインの情報モデルを定義するのに使われる．この定義は，一連のソフトウェア設計から，実装ステップまでをうまく管理するために，関連するソフトウェアの情報モデルとして使われる．

データの視点

プログラマの視点である．現在のメタレベルだけでなく，可能な限り，より上位のメタレベルを見る．データの観点からは，MOF（あるいは，より正確には，MOF の製品）は，分散計算パラダイムに適用し，与えられた情報モデルに対応する情報を管理するために使われる．このモードでは，クライアントが，動的に情報モデル記述を獲得し，自己反映的な処理を実行することを可能にする．

モデル構成要素	概念	概要
Classes	クラス	MOFメタモデル（M2）のメタオブジェクト（M1インスタンス）を作る基本的な要素．1つのクラスは，属性，操作，参照の3つの特性を持つことができる．また，継承，関連の概念を持つ．
Associations	関連	MOFメタモデルでの関連を表現する．関連を持つクラスの両端に，関連末端（名）の概念を持つ．
Aggregation	集約	クラスとデータ型は，関係または属性で他のクラスと関連を持つ．ＭＯＦでは，インスタンス間の関係として2種類の集約（概念的に強い結合と弱い結合）を持つ．
Reference	参照	クラス間の参照関係を表す2つの概念を持つ．つまり，関連の参照と属性の参照，それらは，計算モデルとしてみるか，対応するインタフェースとしてみるかの立場によって異なる．
DataTypes	データ型	属性値と操作のパラメータ値を表現するためのデータ型．基本型とユーザ定義の拡張データ型の2種類のデータ型がある．
Packages	パッケージ	モデル要素をグループ化する概念である．2つの目的がある．M2レベルでは，メタモデルのモジュール化の役割があり，M1レベルでは，パッケージインスタンスとして，メタデータの入れ物としての意味を持つ．
Constraints	制約	属性，データ型がついたクラスノードと，関係ノード間の関連からなるメタデータ情報を定義する．

表-1　MOF モデル構成要素

MOF モデル クラス	概念	MOF モデル クラス	概念
ModelElement	モデル要素	BehavioralFeature	振舞的特徴
NameSpace	名前空間	Operation	操作
GeneralizableElement	汎化可能要素	Exception	例外
TypedElement	型付要素	Association	関連
Classifier	分類子	AssociationEnd	関連末端
Class	クラス	Package	パッケージ
DataType	データ型	Import	インポート
Feature	特徴	Parameter	パラメータ
StructureFeature	構造的特徴	Constraint	制約
Attribute	属性	Constant	定数
Reference	参照	Tag	タグ

表-2　MOF モデルの UOD（クラスとなる概念）

第2は，このMOF仕様が，オープンな情報モデリング機能を提供することである．MOF仕様は，一般のオブジェクト指向情報モデリングに必要な構成要素を含み，最小ではないが比較的小さいコアのMOFモデルを定義している．そして，ユーザが定義したい情報モデルに対して，MOFモデルを1つの基底モデルとして利用することができる．このように，MOFモデルを核として，必要に応じて要素を加えて，継承や合成を用いて，より豊富な情報モデルを定義し拡張できる．この特徴は，設計者が，MOFモデルと思想や詳細が異なる情報モデルを定義することも可能とする．

最後に，MOF仕様で定義されるCORBA IDLマッピングに対するMOFモデルの役割と制約を理解する必要がある．マッピングの主な目的は，標準インタフェースと相互運用性の意味を持つMOFモデルの用語で定義される情報モデルに対するCORBAインタフェースを定義することである．それらのインタフェースは，クライアントがメタモデルで記述された情報を生成，アクセス，更新することを可能にする．そこでは，情報モデル定義の中で規定された構造的，かつ論理的な整合性制約が保持されることが期待される．

以上のような特徴を活用することで，MOF仕様の応用は，ソフトウェア開発，型管理，情報管理，データウェアハウス管理など広範囲のシナリオが描ける．UMLツールを始め，それらを支援するツールの開発が，多くのベンダでなされている．

特に，MOF仕様は，UMLおよびCWMを含む多くの商用製品で採用されている数々のOMG勧告仕様の基礎となっている．後述するXMIは，MOFを基礎にして多様なメタデータを技術的に選択できるようにしたインタフェース形式である．これまで，30以上のUMLツールと約10のCWMの商用ツールでXMIがサポートされている[3),5),7)]．

◆基本概念の概要

MOF仕様は，358ページにおよぶかなり大きなドキュメントである．ここでは，MOFの基本的な概念について説明する．

MOFモデル（MOFのコアメタメタモデルを指す）は，オブジェクト指向モデルであり，UML[12)]のモデル構成要素と，メタモデル構成要素は，“クラス”，“関係”，“データ型”，“パッケージ”などを共通の要素としている．たとえば，MOFのメタモデル全体を表現するのに，UMLモデルと同様，UMLのパッケージ記法を使用する．

◆MOF モデルのメタモデル構成要素

まず，メタモデルを定義するために必要なMOFのコア部分のメタモデル構成要素（すなわち，MOF抽象言語）について述べる．MOFでは，UMLのサブセットを構成要素として用いて，メタデータに関する情報モデルをオブジェクト指向モデリングの手法で定義する．

つまり，次の4つの主なるモデル構成要素を持つ．

1．MOFメタオブジェクトをモデル化する“クラス”
2．メタオブジェクト間の2項関係をモデル化する“関連”
3．他のデータをモデル化するための“データ型”
4．モデルをモジュール化する“パッケージ”

まず，MOFを定義するモデルを記述するため，モデルを構築するためのモデル要素（概念）を定義している．具体的には，**表-1**のようなモデル構成要素が，その使い方の規則や意味などとともに，自然言語で定義されている（M4に相当）．

次に，それらのモデル構成要素を用いて，MOFモデルの対象となる領域の概念（UOD），つまり，モデルを記述するための言語の定義モデルが定義されている．**表-2**は，MOFモデルで定義している概念の一覧である

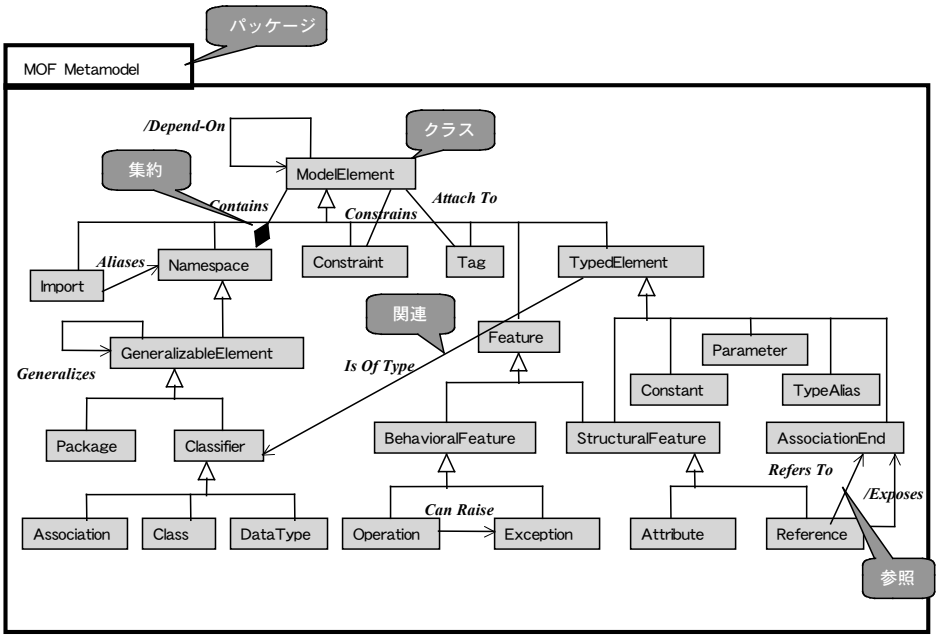

図 -2　MOF モデルパッケージ

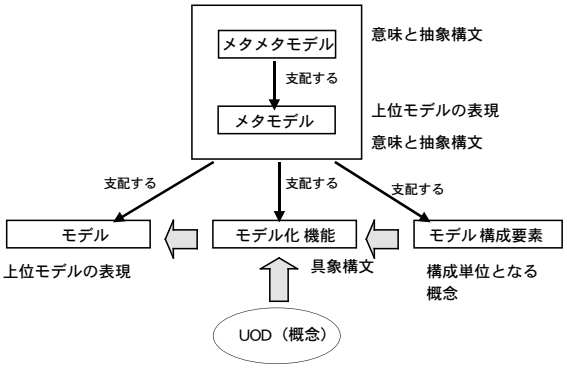

図 -3　MOF モデルとメタモデル

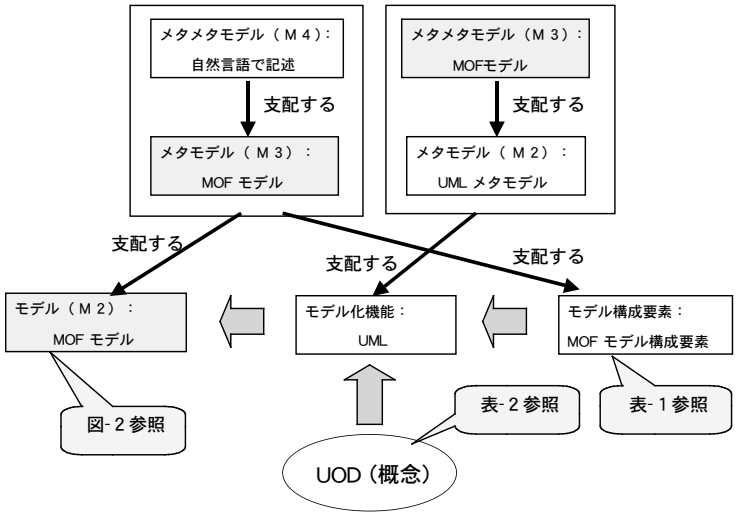

図 -4　MOF モデルの位置付け

（一部省略）．それらを，UML をモデル化機能として，UML ダイアグラムを用いて定義したものが，**図 -2** であり，詳細な仕様は，モデル構成要素とその規則を用いて記述されている．

◆ MOF モデルの位置付け

前編で解説したモデルとメタモデルの関係を**図 -3** に示す．この図式を，MOF モデルにあてはめると，**図 -4**，**図 -5** のようになる．

図 -4 は，MOF モデル自身が，どのように記述されているかの構造を示し，表 -1，表 -2，図 -2 との関係を表している．

ここで，前述のように，MOF モデル（M2）が，自身（M3）を定義するのに使われていることに注意してほしい．本当は，MOF モデルは，それ自身メタモデルである．しかし，その循環定義を区別するようなモデルの表現は持っていない．MOF 仕様を，同じ記法を用いて，つまり，UML 記法，表，および，OCL（オブジェクト制約定義言語）式を使用して記述している．ここでの UML 記法の利用は，MOF の設計者にも，MOF 仕様の読者にも仕様の理解を助ける意味で都合がよい．

表 -1 の MOF モデル構成要素の classes（クラス）を用いて，表 -2 の MOF モデルクラス，たとえば，ModelElement（モデル要素），NameSpace（名前空間）などが規定されている．それらは図 -2 では，矩形（UML のクラス表記）で描かれている．

また，表 -1 の Associations（関連），Aggregation（集約）を用いて，たとえば，TypedElement（型付要素）と Classifier（分類子）との関連 "IsOfType"，ModelElement（モデル要素）と NameSpace（名前空間）との関連 "Contains" などを規定している．図 -2 では，菱形（UML の合成集約の表記）などで描かれている．さらに，表 -1 の package（パッケージ）を用いて，MOF コアモデルの範囲（名前空間）を限定している．

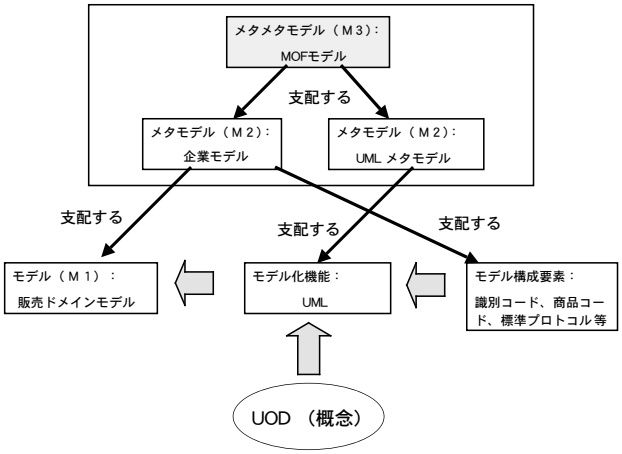

図 -5　MOF モデルの位置付け（メタメタモデルとして）

一方，図 -5 は，MOF モデルが，一般の企業モデル，および販売ドメインモデルのメタモデル記述に利用される場合の図式である．モデル構成要素は，企業モデルのメタモデルによって支配される概念であり，“識別コード”，“商品コード”，“標準プロトコル”などが考えられる．なお，モデル構成要素は，モデル化機能，つまり，モデルの具象構文（表現方法）に依存しない概念的なものである．

ここで，図 -4，図 -5，いずれの場合も，モデル化機能として UML を採用した場合の図式になっているが，一般には，モデル化機能としては，IDEF1X など，別のものでもかまわないことに留意してほしい．

◆ MOF で定義されたメタモデルの例

以下の OMG 仕様のメタモデルは，MOF で記述されている．

- UML メタモデルの例については，**図 -7** を参照．
- MOF の MOF モデル自体が MOF で記述されている（図 -2）．
- CWM（Common Warehouse Metamodel）

CWM[6] は，データウェアハウス間のメタデータの相互運用性を図るために策定された OMG の仕様である．異なったフォーマットを持つデータ資源（UML モデル，Java クラス，リレーショナルモデル（RDB），IMS，など）のメタデータを MOF ベースで記述している．**図 -6** は，リレーショナルモデルの一部である．

XMI

XMI（XML Metadata Interchange）も，MOF と同様，OMG で標準化された仕様である．1999 年に XMI V1.0 が発行され，現在の最新版は XMI V1.2（2000 年 11 月）である [4]．

◆ XMI と MOF

XMI 仕様をひとことで表現すると，「MOF で定義され

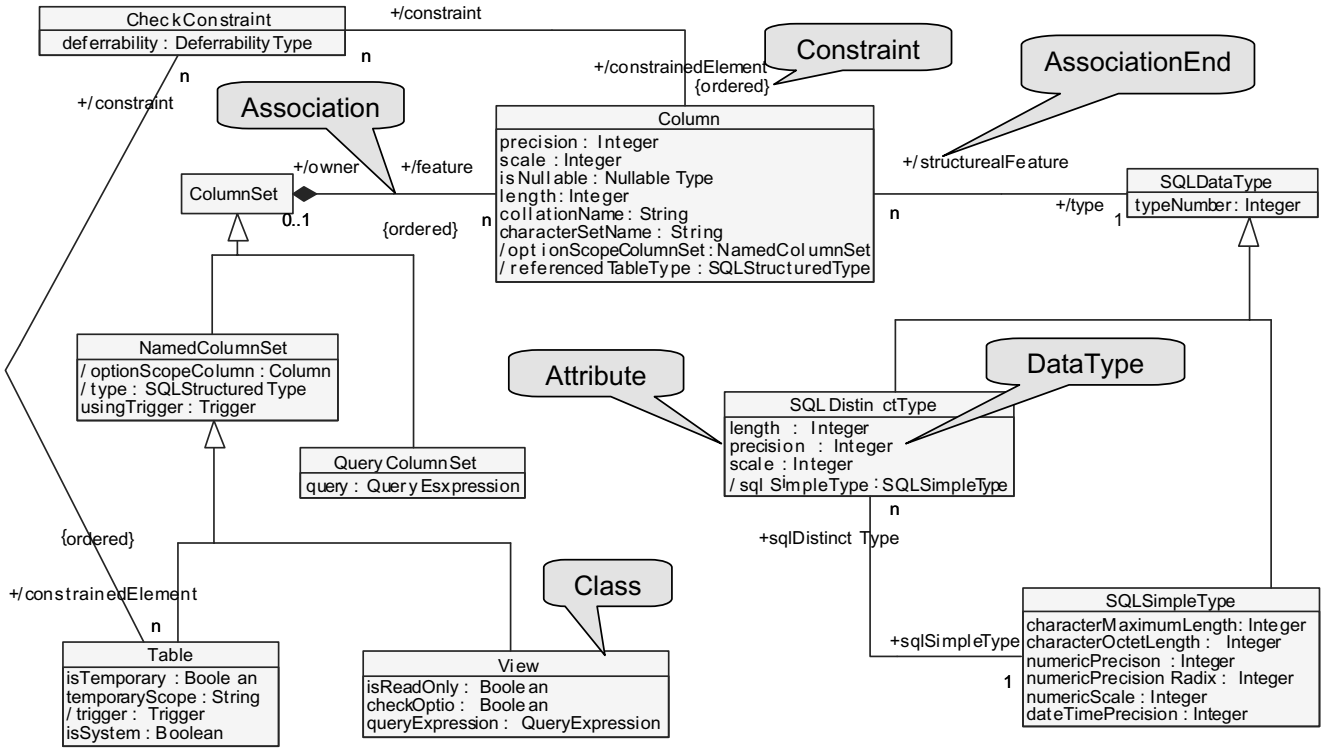

図 -6　CWM メタモデル

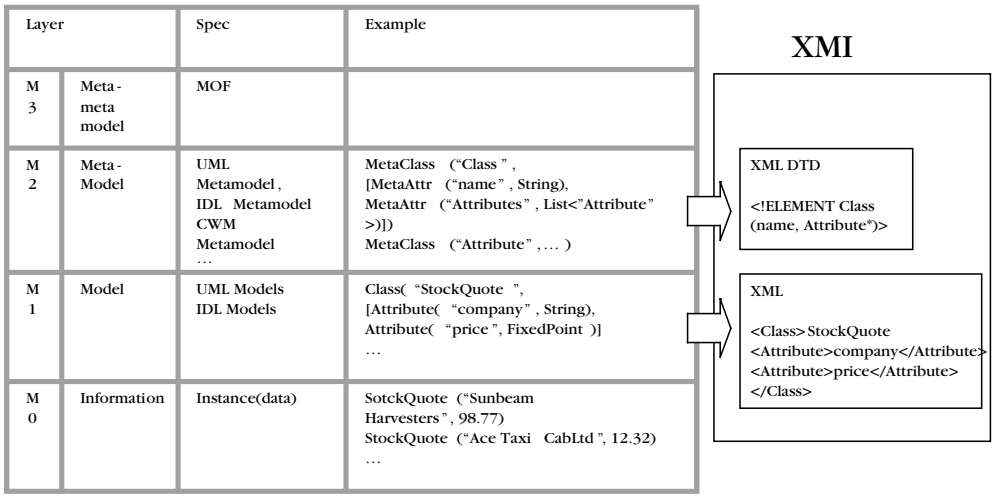

図 -7　4 階層メタデータアーキテクチャと XMI の関係

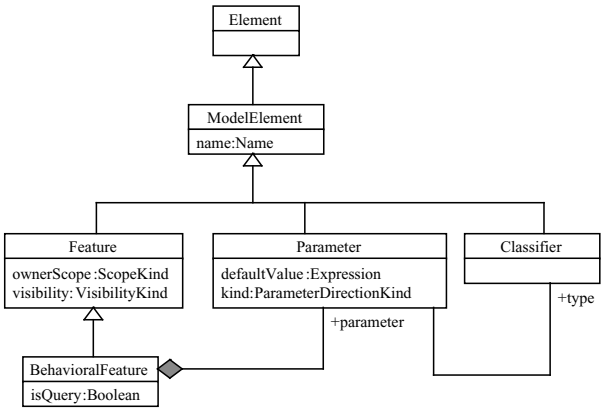

図 -8　UML モデル

たメタモデル，およびそのメタモデルに従って記述されたモデル情報を，テキストストリームで交換するための仕様」である．テキストストリームの形式としては，W3C[9] の XML（Extensible Markup Language）[10] が採用されている．XMI 仕様では以下の 2 つの仕様を提供している．

(1) XML-DTD 生成ルール
XMI 仕様では，MOF ベースのメタモデルの定義情報から，メタモデルに対応した DTD（Document Type Definition）を機械的に生成するルールが規定されている．DTD とは，XML 世界におけるスキーマ定義であり，XML 文書の形を規定するものである．XMI 仕様により，個々のメタモデルに対応した DTD が自動的に定められる．

(2) XML 文書の生成ルール
MOF で定義されたメタモデルに従って定義されたモデル情報を XML 文書として符号化するための仕様である．

これらの仕様により，メタモデル情報，およびモデル情報を XML 形式で表現することができる．さらにこの XML は，メタモデルを共有する手法・ツール間で共通であるため，情報の交換が可能となる．図 -7 に，4 階層メタデータアーキテクチャと XMI の関係を示す．

◆ XMI の例（UML-XMI）

上に述べたように，XMI は，そもそも，いかなる MOF ベースのメタモデル・モデルも交換可能な仕様である．しかし，XMI は UML のメタモデルおよびモデルの交換形式として中心的に用いられるため，しばしば XMI を UML 専用の交換仕様として説明されることがある．ここでは，この最もポピュラーな UML のメタモデルを例にとり，UML メタモデルと生成された DTD，およびモデル情報と XML 文書との関係を説明する．

前編で，UML メタモデルの一部を例示したが，その部分のごく一部の "BehavioralFeature" 部分を再掲する（**図 -8**）．

この BehavioralFeature に対応して生成された DTD を示す．

```
<!ELEMENT　BehavioralFeature
（ModelElement.name
|　Feature.ownerScope
|　Feature.visibility
|　BehavioralFeature.isQuery
|　BehavioralFeature.parameter）* >
<!ELEMENT　BehavioralFeature.parameter
（Parameter）* >
<!ELEMENT　Parameter
（ModelElement.name
|　Parameter.defaultValue
|　Parameter.kind
|　Parameter.kind
|　Parameter.type）* >
```

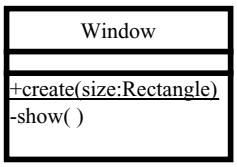

図-9　単純なクラスの例

Behavioral Feature が持つ属性は，isQuery，および parameter のみである．複数の Parameter 集合を BehavioralFeature.parameter として定義し，BehavioralFeature が複数の Parameter を所有できるように定義されている．

それ以外の属性は，継承関係の上位メタクラスが保持する属性である．DTD では，オブジェクト指向の「継承」にあたる記法がないため，上位オブジェクトから引き継いだ属性についても，すべて列挙する必要がある．1 行目の name は ModelElement から，ownerScope, visibility は直上の Feature から継承したものである．このように，XMI の DTD は，メタモデル構造を直接的に反映したものになっている．

さて，今度は，この DTD に従った XML でモデル情報がいかに表現されるかを見ていく．モデルの例として，**図-9** の単純なクラス Window を用いる．

このクラス内の操作[☆1]create は，以下のような XML 文書として表現される．

```
<BehavioralFeature>
 <ModelElement.name>create</ModelElement.name>
 <Feature.ownerScope>classifier
</Feature.ownerScope >
 <Feature.visibility>public</Feature.visibility>
 <BehavioralFeature.isQuery>false
</ BehavioralFeature.isQuery>
 <BehavioralFeature.parameter>
   <Parameter>
<ModelElement.name>size</ModelElement.name>
<Parameter.type>Rectangle</Parameter.type>
 </Parameter>
</BehavioralFeature.parameter>
</BehavioralFeature>
```

このように，XMI 形式のモデル情報は，メタモデルから生成された DTD 内のエレメント（タグ）を用いて構成される．

◆ XMI の今後

2001 年 5 月，W3C において，DTD に変わる XML 用の新しいスキーマ仕様 XMLschema[11] が勧告となった．XMI においても XMLschema に対応した仕様 XMI Production of XML Schema の策定が進んでいる．

MOF/XMI とメタモデリング技術

本編では，MOF や XMI のメタモデリング技術を外観してきた．では，このようなメタモデルがなぜ重要であるのか，もう一度，まとめておきたい．

メタモデルは，モデリング手法のセマンティクスを規定した仕様という意味で重要である．また，実用的見地から見ると，モデリング手法，ガイドラインとしての役割が大事である．メタモデル図を一瞥することで，そのモデリング手法の記述能力の範囲，記述ルールについての概要が理解できる利点がある．メタメタモデルは，このモデリング手法の中核ともいうべきメタモデル自体の共通の表現手段を提供する．メタメタモデルを標準化することで，メタモデルの定義，記述方法，読み方，解釈の標準化が可能となる．さらにモデル間の情報交換が容易になる．そして，メタモデルを含むメタデータを，実行時に参照し，動作するような自動化ソフトウェアの出現を導く可能性を秘めている．

今後は，メタモデリングに関する，ますますの標準化が必要であり，関連技術の発展が期待される．

参考文献

1) OMG: http://www.omg.org
2) OMG, Meta Object Facility (MOF) Specification Version 1.4, 2002 (02-04-03). http://www.omg.org/technology/documents/formal/mof.htm
3) OMG, MOF 関連情報 : http://www.omg.org/technology/cwm/
4) OMG, XMI（XML Metadata Interchange）Specification V1.2: http://www.omg.org/technology/documents/formal/xmi.htm
5) OMG, XMI 関連情報 : http://www.omg.org/technology/XML/index.htm
6) OMG, CWM（Common Warehouse Metamodel）: http://www.omg.org/technology/documents/formal/cwm.htm
7) OMG, UML 関連情報 : http://www.omg.org/technology/uml/index.htm
8) ISO/IEC 14750（ISO Interface Definition Language，OMG IDL specification）
9) W3C: http://www.w3.org/Consortium/
10) W3C, XML: http://www.w3.org/XML/
11) W3C, XML Schema: http://www.w3.org/XML/Schema
12) OMG JapanSIG, 翻訳委員会編 : UML 仕様書，ASCII（2001）.

（平成 14 年 11 月 1 日受付）

☆1 UML メタモデルでは，操作（Operation）は BehavioralFeature の 1 階層下であるが，ここでは簡略化のため Operation と BehavioralFeature を同一視している．