Appendix

LSI設計者のための UML 2.0表記法ガイド

塚田雄一

ここではUMLの表記法について紹介する．UML 2.0では12種類の表記法が規格化されている．LSI設計では，例えばユースケース図やクラス図，シーケンス図などが用いられる．ハードウェア設計者にとってはなじみの状態遷移図やフローチャートに相当する表記法もある．（編集部）

UMLは“unified modeling language”，すなわち「統一モデリング言語」の略です．「言語」というとC/C++やJavaなどのプログラム・コードを思い浮かべるかもしれませんが，UMLはテキスト・ベースの記述言語ではなく，モデリング言語です．システム分析やシステム設計などを行うためのもので，システムをモデル化（抽象化）して図示するための表記法を規定しています．さらに，UMLによるシステム設計ではオブジェクト指向の考えかたを適用できます．そのため，UMLは「オブジェクト指向のための統一モデリング言語」と呼ばれています．

UMLの最新バージョンは2.0で，UML 2.0では12種類の表記法が存在します（**表1**）．ここで注意していただきたいのは，UMLはあくまでも図の書きかたにすぎないということです．まずは表記法を覚え，表記したいときに必要に応じて使いたい図を描きます．つまり，UML表記法をどのように使うかが重要となってきます．

● LSI開発に適用すると何がうれしいのか？

まず最初に，UMLをLSIの開発に適用した場合，何がうれしいのかについて説明します．

第1に，ドキュメントとしての効果があります．UMLはソフトウェア開発の標準言語です．よって，UMLをLSIの開発に導入すると，ソフトウェア技術者もレビューに参加できます．また，担当者が変わったときの引き継ぎなども容易になります．そして，英訳も比較的容易に行えるため，海外とのコミュニケーションにも役立ちます．

第2に，UMLはシステム全体の構造を把握するのに便利な表記法です．UMLを使うと，システムの構造を考えながら設計しやすくなります．その際，オブジェクト指向を利用することで，似たような処理を一つにまとめられます．こうすると，むだな処理の記述を減らせます．また，実装後の回路面積を小さくすることに役立ちます．

第3に，ほかのモジュールなどとの関連が記されているため，仕様変更などの際にほかのモジュールへの影響を把握しやすくなり，開発効率や設計品質が向上すると考えられます．

以下では，UML 2.0のモデルを紹介します．とくに，LSIの開発において重要と思われるモデルの記述を詳細に説明していきます．一方，使用頻度が少ないと思われるモデルの記述は，細かい説明を省略しています．

● ユースケース図：要求される機能をユーザの視点で表現

システムに要求される機能をユーザの視点から見た形で表現するのがユースケース図です．

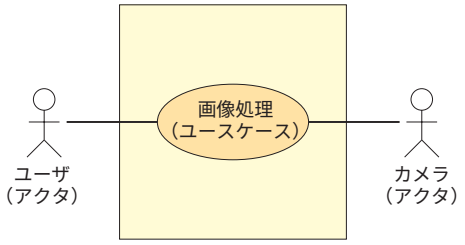
1）ユースケース図の基本表記

ユースケース図は，「アクタ（ユーザ）」と「ユースケース（システムの機能）」の二つから成り立っており，アクタである「ユーザ」とユースケースである「システム」の対話関係をモデル化します．

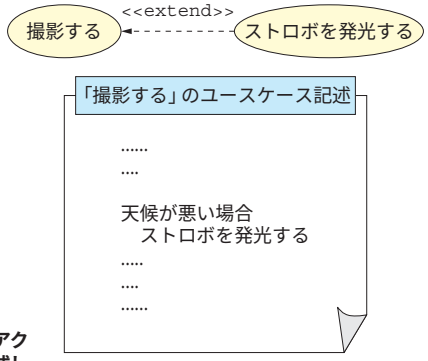
アクタは人の形で示します．そして，ユースケースはだ円で示します．また，システムの外部と内部を明確にするための境

表1　UML 2.0がサポートしている図

名　前	説　明
ユースケース図	システムのユーザ要求を表現
クラス図	部品（クラス）の内容と関係を表現
オブジェクト図	部品（オブジェクト）の関係を表現
シーケンス図	時間軸のオブジェクトの流れを表現
ステートマシン図	オブジェクトの状態や遷移を表現
アクティビティ図	処理の流れを表現
パッケージ図	グループ化した単位の関係を表現
コミュニケーション図	オブジェクト間のメッセージのやり取りを表現
タイミング・チャート	時間軸の変化を表現
コンポーネント図	部品の構造を表現
コンポジット・ストラクチャ図	クラスやコンポーネントの内部構造を表現
配置図	システムの物理的な配置を表現



（a）ユースケース図



（b）ユースケース図の拡張記述

ユースケース番号：
ユースケース名：
概　要：
アクタ　：
事前条件：
事後条件：
トリガ　：
基本シナリオ：
①処理1
②処理2
③処理3
④処理4
⑤処理5
代替シナリオ：
異常シナリオ：
コメント：

（c）ユースケース記述の例

図1　ユースケース図

ユースケース図はシステムのユーザ要求を表現する．(a)はアクタ(ユーザ，カメラ)とユースケース(画像処理)の関係を記述している．(b)のユースケース図の拡張記述は，すでに存在するユースケースを拡張する際の表現である．例えば，ユースケース記述のシナリオで「天候が悪い場合」という特定条件で，「撮影する」というユースケースから<<extend>>を使用して，「ストロボを発光する」と拡張した記述が可能となる．

界線として長方形を書きます．最後に，アクタとユースケースの関連を線で結びます．例えば，ユーザ，カメラというアクタから画像処理というユースケースを記述した例を図1(a)に示します．モデリングを行ううえでは，アクタからの視点で考えるということがポイントになります．

すでに存在するユースケースを拡張して別のユースケースを表現する場合，<<extend>>や<<include>>などを使用します．例えば<<extend>>は，特定の条件が成立したときのみ実行する記述です．これをユースケース図の拡張記述と呼びます．例えば，「天候が悪い」という特定条件の場合，「ストロボを発光する」という処理の拡張表現は図1(b)のようになります．特定条件は，次項で説明する「ユースケース記述」を使って示します．

2）ユースケース記述

ユースケース図は見てもわかるとおり，非常に単純なものです．したがって，ユーザとのレビューを行う際には補足説明が必要になります．その補足説明に当たるものがユースケース記述です．ユースケース記述にはとくに決まった記述方法があるわけではありません．ユースケース記述では，「アクタは何であるか？」，「実行するまでの前提条件は何か？」，「実行するためのトリガは何か？」といったことを記述します．また，実行のためのシナリオなども記述します．

ユースケース記述ではシナリオ内の分岐(代替)異常をもれなく記述することが重要になります．図1(c)に，記述フォーマットの例を示します．なお，ユースケース記述において，図1(c)の項目のすべてをかならず記述しなくてはならないというわけではありません．

● クラス図：部品の構成や関連を表現する最重要モデル

クラス図は，部品の構成や関連などを表現するのに用いられます．クラス図はUMLの中でもいちばん基本となるモデルです．そもそもクラスとは，部品や集合を示します．学校にたとえると，生徒(という部品)と先生(という部品)が集まって一つのクラスになっているのと同じです．

1）クラス図の基本表記

クラス図は，三つの領域で区切られた長方形で表現されます(図2(a))．そしていちばん上の領域に「クラス名」を記述し，中央の領域に「属性(データ)」を記述します．そして，いちばん下の領域に「操作(処理)」を記述します．ちなみに操作では，関数であることを示すため，ことばの末尾に()を付けます．一つのシステムは複数のクラスから構成されます．個々のクラスはほかのクラスと何らかの関連を持っています．関連は線で表現します．

2）クラスの内部詳細記述

次にクラスの内部詳細記述について説明します(図2(b))．クラスには可視性という項目があります．クラスを使う側から見える場合は属性や操作の前に「+」を記述し，クラスを使う側から見えない場合は「-」を記述します．属性にはデータ型と初期値を記述できます．そして操作にはパラメータ(引き数)と戻り値(リターン値)が記述可能です．

3）誘導，関連クラス

クラス間の関連は，通常，実線で接続されます．ただし，あるクラスから別のクラスに誘導可能な場合は「→(矢印)」で記述します(図3(a))．

クラスとクラスの接続において詳細記述を行いたい場合は，図3(b)のような関連クラス記述を利用します．