第 22 章　情報システムを表現するモデル

モデルとは何か

いつも引き合いに出している ISO と IEC、IEEE が合同で発行した Vocabulary には、「モデル」について次の記述がある[ISO10a]。

「1. 現実の世界のプロセス、デバイス、あるいはコンセプトを表現したもの。
2. モデル化された対象物のある側面を隠した表現。
3. ある理論について、その理論の全てが真実であることの表現。
4. 情報システムや、その一部であるソフトウェア・プロダクトなどを表現するメタオブジェクトの、関連するインスタンスの集まり。
5. ある特別の視点から見たシステムの完全な記述。及び、意味的に閉じたシステムの抽象的表現。」

難しい表現が並んでいるが、詰まるところいろんな側面を持つ対象物のある側面を取り上げて、その立場からその対象物を記述したものがモデルである。したがって、1 つの対象物にはいくつかのモデルによる複数の表現があり得る。

情報システムを表現するモデル。

今のソフトウェア工学では、以下の 3 種類のモデルを使うことによって、必要かつ充分に詳細に情報システムを表現できるとしている。

構造モデル（静的モデル）: 情報システムの処理の対象物として何があり、それらが相互にどういう関係を持っているかを表すもの。
状態推移モデル（動的モデル）: 情報システムの処理の対象物のあるものがその状態を推移させる場合、何がきっかけで、どのように状態が推移するのかを表すもの。
プロセスモデル：コンピュータ内部での処理の内容、順序を記すもの。

ただし、全ての情報システムで常にこの 3 種類のモデルを必要とする訳ではない。ビジネス・アプリケーションでは処理の対象物の状態を特別に重視しない場合があり、その場合には状態推移モデルは記述されないことがある。また制御系システムでは処理の対象物の数が少ない／構造が簡単という場合があり、その場合には構造モデルは記述されないことがある。

図表 22-1　構造化技法で使用されるモデル群

種類	モデル
構造モデル	ER 図（実体関連図）
状態推移モデル	状態遷移図
プロセスモデル	データ・フロー・ダイアグラム

構造化技法のモデル群

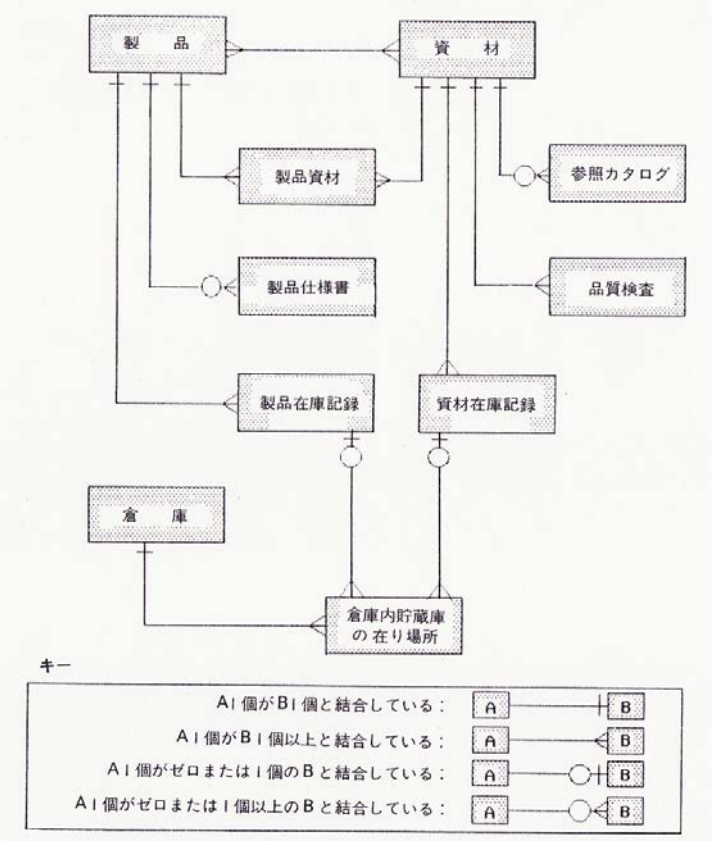
構造化技法では、情報システムを記述するために図表 22-1 で示すモデル群が使用されてい

る[1][MART85]。

以下で、それぞれのモデルについて概略を記す。

ER 図

ER 図（エンティティ・リレーションシップ・ダイアグラム、実体関連図）は、情報システムの処理の対象になるものを「実体」と「関連」に分けて図示するものである。構造化技法では、ER 図はデータベースの構造という意識が強かった[2]。



図表 22-2　ER 図（その 1）（[MART85]より）

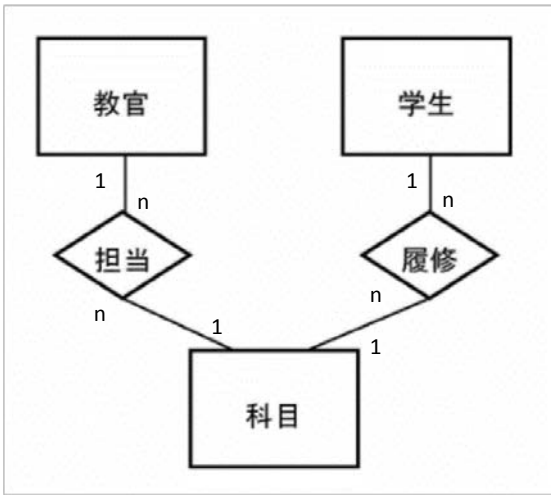
ER 図にはいくつかの表現の方法があり、図表 22-2 はその 1 つの例である。ここでは、実体（純粋の名詞で表されるもの）のみが表示されている[3]。実体と実体を結ぶ線に記された記号は多重度（カーディナリティ）を表している。この多重度については、図表 22-2 の下の部分に説明がある。

---

[1] 構造化技法については、第 15 章で記した。

[2] ER 図は当初、米国マサチューセッツ工科大学（MIT）のチェンが提唱した。

[3] 情報システムの処理の対象物、及びデータベース化の対象物は、構造化技法では「実体」と「関連」、データ中心アプローチでは「管理対象」、オブジェクト指向技法では「オブジェクト」又は「クラス」と呼んでいる。名前はそれぞれ異なるが、基本的な概念は変わらない。



図表 22-3　ER 図（その 2）

図表 22-3 は、ER 図についての別の例である。ここでは実体（純粋の名詞で表されるもの）が四角形で、関連（動詞の語幹で表されるもの）が菱形で表されている。多重度は実体と関連を結ぶ線に沿って、実体と関連の近くにそれぞれ 1（常に 1 つある）とか n（1 以上の複数個存在する）とかで表されている[4]。

状態遷移図

状態遷移図は、ER 図（実体関連図）で取り上げた 1 つの「実体」、又は「関連」（データ中心アプローチ[5]ではいずれも「管理対象」と呼ばれる）の状態の遷移を表すものである。

状態遷移図の例を、図表 22-4 に示す。

ここで、1 つの状態は円で表されており、状態の遷移は円と円を結ぶ矢印付きの線で表されている。その線に沿って記述されているものは、その状態の遷移を引き起こすきっかけである。そのきっかけによって何かの処理が開始され、それで何らかの結果が出る場合には、そのきっかけの次に斜線（／）を引いて結果を記述することもある。

データ・フロー・ダイアグラム

構造化技法では、コンピュータの中での処理はデータ・フロー・ダイアグラム（DFD）で記述される[6]。その DFD の例を、図表 22-5 に示す。

DFD では、4 つの記号を用いている。その最初のものは「処理」で、図表 22-5 では角の丸い四角形を使って表している。四角形の中に、その処理の内容を記述している。

矢印のついた線は、データの流れを表す。データは、矢印のないところから矢印の方向に流れる。双方向に流れる場合には、両方に矢印のついた線を使用して表す。

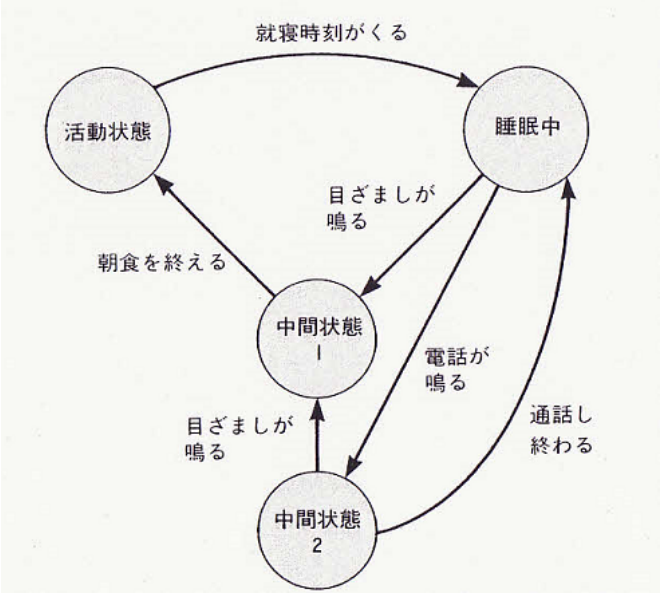
平行線はデータベース、又はファイルである。DFD ではこれを「データ・ストア」と呼んでいる。

最後のものは「ターミネータ」と呼ばれるもので、四角形で表現している。ターミネータは

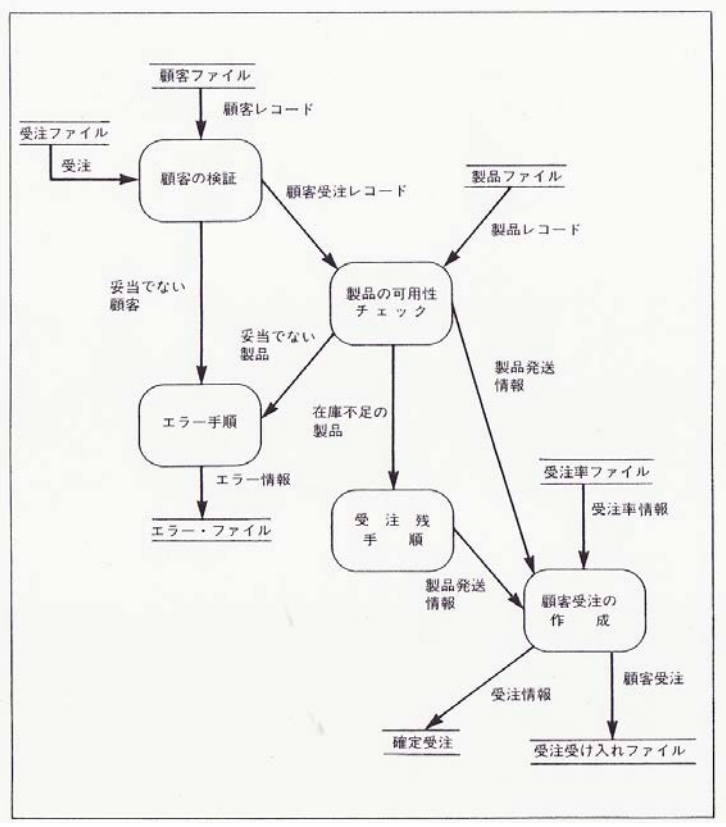
---

[4] この ER 図の書き方は、すでに第 16 章で述べた。
[5] データ中心アプローチについては、第 16 章で記した。
[6] データ・フロー・ダイアグラムは、当初トム・デマルコ（Tom DeMarco）が提唱した。

データを発生させるもの、及び生成された情報を消費するものである。（ターミネータは図表 22-5 では表示されていない。）



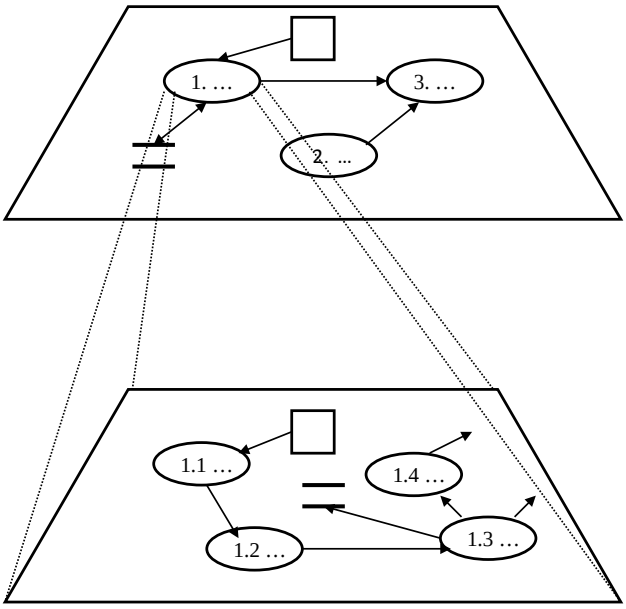
図表 22-4　状態遷移図の例（[MART85]より）



図表 22-5　データ・フロー・ダイアグラムの例（[MART85]より）

1 枚の DFD で全ての処理を書き切れない場合は、階層化して表現することができる。この場合には「処理」に、例えば 1、2 というような番号を振る。その上で、上位の DFD の 1 つの処理の内容を、下位の DFD1 枚を使って表現する。例えば上位の 1. の処理を階層化した場合には、下位の処理には 1.1、1.2 というような番号を振って、上下関係を表現する。DFD の階層化について、図表 22-6 に示す。

これ以上処理を詳細化する必要がないところまで階層化した場合、最も下位の処理の内容を文章や図、デシジョン・テーブルなどを使って誤解のないように表現する。これを、「ミニスペック」と呼ぶ。



図表 22-6　DFD の階層化

図表 22-7　モデル間の関係

	実体と関連	イベント	処理の内容
ER図	実体と関連と、その間の関係について記述する	（なし）	（なし）
DFD	DFD上に、データストアとして表示される	イベント毎に、DFDを作成する	処理の内容を、DFDに記載する
状態遷移図	主な実体と関連毎に、状態遷移図を作成する	イベントが状態の遷移を引き起こす。その状態遷移の線に沿って、イベント名を併記する	イベントと併せて、処理の内容を略記する

3 つのモデルの関係

最近は、これらのダイアグラムの作成はツールを使ってなされるのが普通である。しかし手でこれらを作成していた場合には、これらのダイアグラムを作成するに当たって、相互の整合性をしっかりと確保することに気を遣わなければならなかった。

ER 図、データ・フロー・ダイアグラム（DFD）、状態遷移図の間で、実体と関連、処理のきっかけになるイベント、及び処理の内容がどのように取り扱われるかを図表 22-7 で示す。

オブジェクト指向技法でのモデル群

オブジェクト指向技法[7]では、情報システムを記述するために図表 22-8 で示すモデル群が使用されている[BOO05]。

図表 22-8　オブジェクト指向技法でのモデル群[BOO05]

種類	モデル
構造モデル	クラス図
状態推移モデル	状態マシン図
プロセスモデル	シーケンス図、コミュニケーション図

オブジェクト指向技法ではこれ以外に、情報システムの概要をユースケース図で記述している。

これらは全て、UML（Unified Modeling Language）の中で定義されている[8]。

以下で、これらのモデルについて概略を記す。

ユースケース図

情報システムが誰に、どのように使用されるのかを、全体を 1 枚の図で表すものがユースケース図である。ユースケース図の例を、図表 22-9 で示す。

枠で囲まれた部分の内部が情報システムで、そこに楕円で囲まれた機能が準備されている。機能名は、その楕円の中に記述されている。情報システムの外側から「アクター」と呼ばれる人や他の情報システムがそれらの機能を使用する。どのアクターがどの機能を使用するのかを、両者の間に線を引いて示す。

どのアクターがどのような場合に、どの機能をどのように使用するかを、文章等で表現することができる。その文章等で表現されたものを「ユースケース」と呼んでいる。ユースケースは、要求仕様、あるいは要件定義として使用することができる[COC01]。

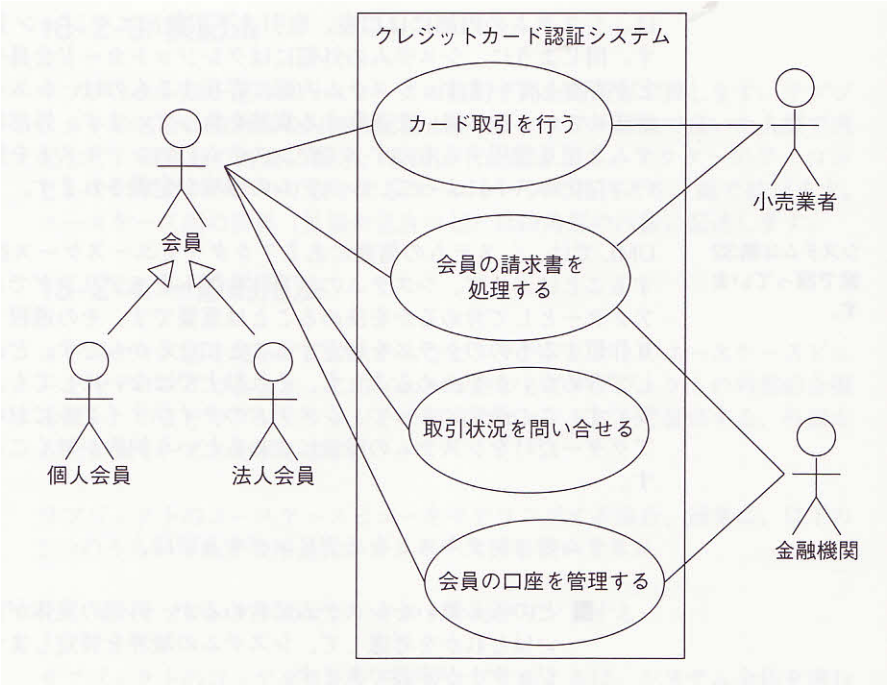
クラス図

ER 図で表される実体と関連が、クラス図ではその両者の区別をなくして全てクラスとなる。それ以外にクラス図では、ER 図では表現できないクラス体系（「is a」構造、及び「part of」構造）を書き込むことができる。図表 22-10 で、クラス図の例を示す。

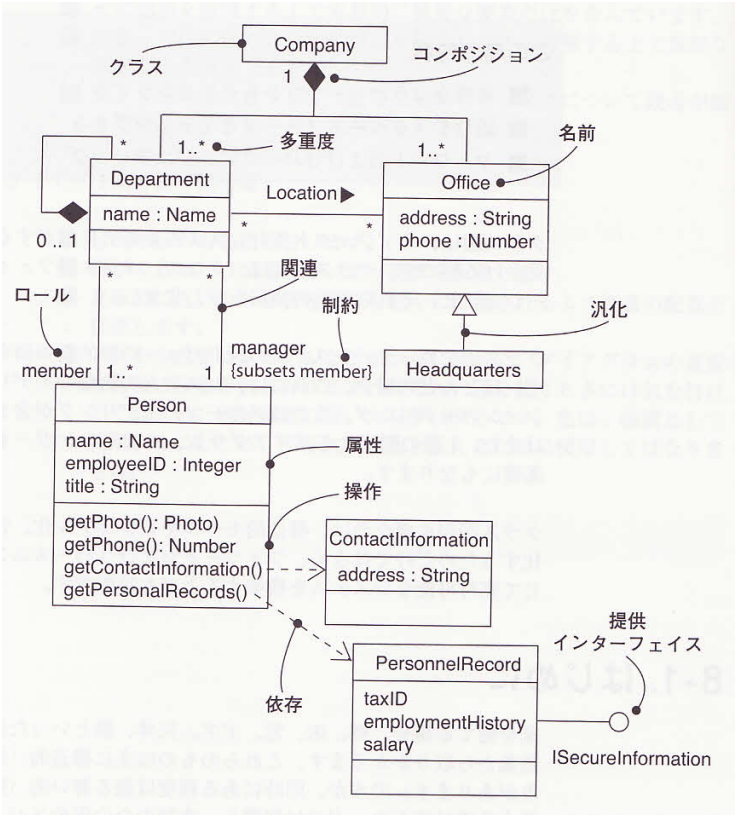
---

[7] オブジェクト指向技法については、第 17 章で記した。

[8] UML は、当初はスリー・アミーゴ（3 人の偉人）と呼ばれているグラディ・ブーチ（Grady Booch）、ジェームズ・ランボー（James Rumbaugh）、及びアイヴァー・ヤコブソン（Ivar Jacobson）が完成させた。



図表 22-9　ユースケース図の例（[BOO05]）



図表 22-10　クラス図の例（[BOO05]）

クラス図では、クラスは 3 つの部分に分けられる。最上段にはクラス名を書き、中断にはそのクラスが持つ属性の名前を、下段にはそのクラスが持つメソッド（プログラム）を記すことができる。最上段のクラスの名前以外は、必要がなければ割愛しても良い。

状態マシン図

状態マシン図は、基本的に状態遷移図と変わらない。状態を入れ子で表すことができる程度である。

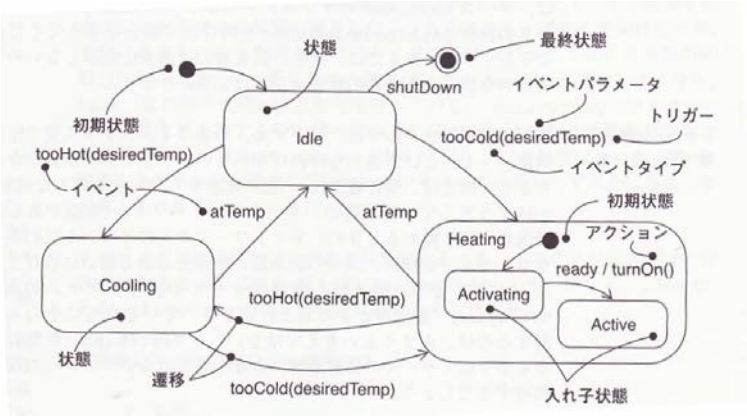
状態マシン図の例を、図表 22-11 で示す。

シーケンス図

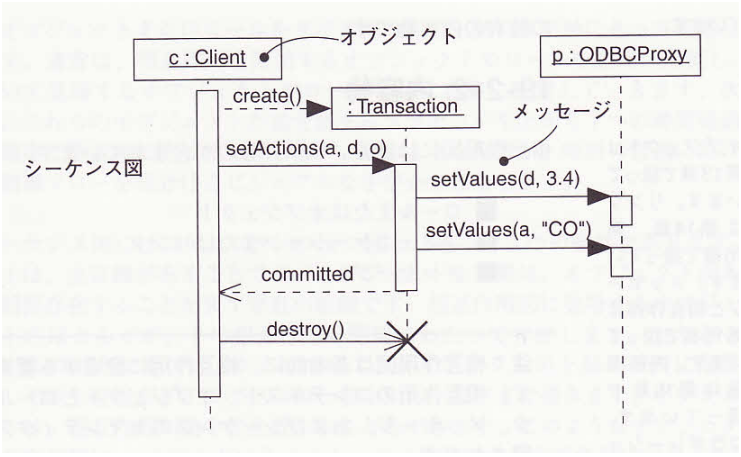
UML では、プロセスモデルの記述の方法はシーケンス図とコミュニケーション図の 2 つの方法が用意されている。

シーケンス図では上段にクラスを並べ、時間の経過と共にどのクラスがどのクラスを呼ぶのかを記す方法で記述する。

シーケンス図の例を、図表 22-12 で示す。



図表 22-11　状態マシン図の例（[BOO05]）

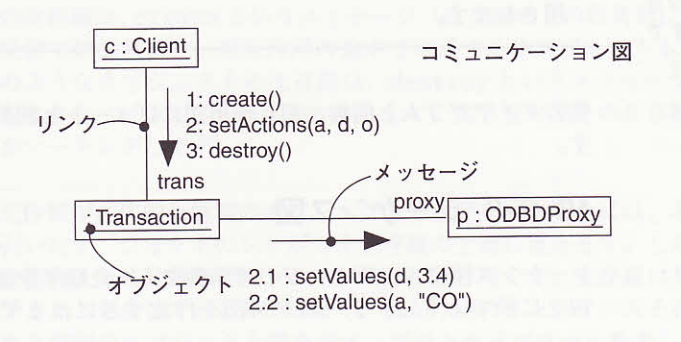


図表 22-12　シーケンス図の例（[BOO05]）

コミュニケーション図

シーケンス図で示したものと同じ内容を、コミュニケーション図を使用して表すことができる。

コミュニケーション図の例を、図表 22-13 に示す。



図表 22-13　コミュニケーション図の例（[BOO05]）

URL でのそれ以外のモデル類

URL には、前記 4 つ以外に多くのモデル類が定義されている。

しかしそれらは主として設計段階や実装段階で使用されるもので、要件定義段階で使用されるものはここで述べた 4 種類が主たるものである[BOO05]。

キィワード

モデル、構造モデル、静的モデル、状態推移モデル、動的モデル、プロセスモデル、ER 図、実体関連図、実体、関連、多重度、カーディナリティ、状態遷移図、管理対象、DFD、データ・ストア、ターミネータ、ミニスペック、ユースケース図、アクター、ユースケース、クラス、クラス図、クラス体系、状態マシン図、シーケンス図、コミュニケーション図

略語

DFD：Data Flow Diagram

人名

トム・デマルコ（Tom DeMarco）、グラディ・ブーチ（Grady Booch）、ジェームズ・ランボー（James Rumbaugh）、アイヴァー・ヤコブソン（Ivar Jacobson）

参考文献とリンク先

[BOO05] グラディ・ブーチ他著、羽生田栄一監訳、越智典子役、「UML ユーザガイド 第 2 版」、ピアソン・エデュケーション、2010 年.

この原書は、以下のものである。

Grady Booch, James Rumbaugh, Ivar Jacobson, “The Unified Modeling Language Use Guide Second Edition. “Addison-Wesley, 2005.

[COC01] アリスター・コーバーン著、ウルシステムズ（株）監訳、「ユースケース実践ガイド -効果的なユースケースの書き方」、（株）翔泳社、2001 年.
この本の原書は、以下のものである。
Alisteir Cockburn, “Writing Effective Use Case,” Addison Wesley Longman, 2001.
[ISO10a] ISO/IEC/IEEE, “System and software engineering – Vocabulary－ISO/IEC/IEEE 24765:2010(E),” ISO/IEC, 2010-12-15.
[MART85] J. マーチン、C. マックルーア著、国友義久、渡辺純一訳、「ソフトウェア構造化技法 ダイアグラム法による」、近代科学社、昭和 61 年.
この原書は、以下のものである。
James Martin, Carma McClure, “Diagraming Techniques for Analysts and Programmers,” Prentice Hall, 1985.

（2014 年（平成 26 年）5 月 15 日　新規作成）