

(株)NTTデータ 技術開発本部  
我妻 智之 (mda-info-ml@rd.nttdata.co.jp)

## 1 本連載について

前回はUML2.0の概要を説明した。また、UML1.Xにおける課題とそれらを解決するためにUML2.0で新たに導入された記法や概念を紹介した。

今回以降はUML2.0の中身を具体的に紹介していく。第2回目は、UML2.0の静的構造の記述方法に焦点をあてて、その概要とUML1.Xとの違いを中心に説明する。

なお、本稿はUML1.Xをご存知の読者を対象としたものである。UML1.Xの詳細に興味がある読者は、恐縮だが、本誌で過去に連載さ

れたUML紹介記事<sup>[1]</sup>等を参照していただきたい。

UML2.0ではソフトウェアの静的構造を記述するために、クラス図、パッケージ図、オブジェクト図、コンポジットストラクチャー図、コンポーネント図、配置図を定義している。また、これらのダイアグラムを総称して「構造図」と呼ぶ。それぞれの概要を表1に示す。

次項以降、各ダイアグラムの詳細を説明する。

なお、本稿で紹介する図例は参考文献[2]および[3]から引用した。また、記法やメタモデルの詳細については[3]を参照されたい。

## 2 クラス図

UMLを利用する際に最も利用頻度が高いダイアグラムはクラス図であろう。UML1.Xでもクラス図は頻繁に利用されていた。ここでは、クラス図の概要とそれがUML2.0でどのように変わったかを説明する。

### (1) クラス図の概要

クラス図は、オブジェクトの型の宣言とそれらの静的な関連を定義したものである。各型の定義であるクラスは、実世界のオブジェクト構造から導き出すことができる。クラス図の例を図1に示す。

表1 UML2.0のダイアグラム一覧

ダイアグラム名	概要
クラス図	システム内のオブジェクトの型とそれらの関連の静的な関係を定義する
パッケージ図	パッケージ内の要素の名前空間を定義する
オブジェクト図	クラスのインスタンス例を記述する
コンポジットストラクチャー図	インスタンスの内部の構造を定義する
コンポーネント図	アーキテクチャ構築のためのサブシステムやその内部構成要素を定義する
配置図	物理的なソフトウェアユニットやハードウェアの関係を記述する

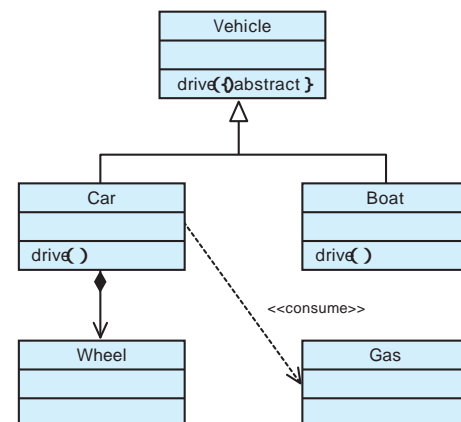


図1 クラス図

クラス図は、クラス名、属性、操作をまとめて記述したクラスと、それらクラス間を結ぶ関連線によってできた図である。図1に示すように関連線では「継承 (generalization)」、

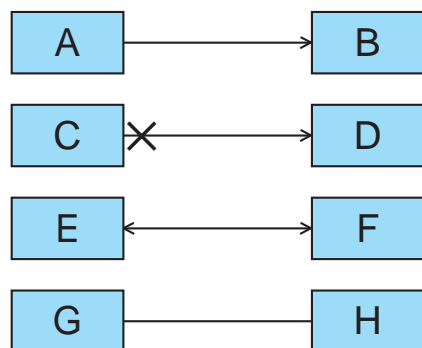


図2 誘導可能性

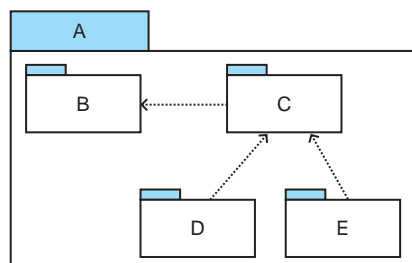


図3 パッケージ図

「関連 (association)」、 「依存 (dependency)」、 「集約 (aggregation)」など、様々な種類の関係を表すことができ、その種類によって様々な種類の線を用いる。

(2) UML1.Xからの変更点

UML2.0では細かい点でいくつか記法が拡張あるいは修正されたが概ね変更なく利用することができる。

ただし、クラス間の関連の参照方向 (誘導可能性) に関する表記が明確になったことを変更点としてあげることができる。図2はこの例を示している。一番目の例ではクラスAからクラスBは参照可能であり逆は未定義であることを示している。2番目の例は、CからDは参照可能であるが、DからCは参照不可能であることを示している。3番目の例は双方向に参照可能であることを示している。4番目の例は双方向とも未定義であることを示している。

3 パッケージ図

(1) パッケージ図の概要

パッケージ図はUMLモデル要素をグループ化するために利用する。また、パッケージはパッケージ化されたモデル要素に対して名前空間を提供する。一般的なパッケージ図の例を図3に示す。

(2) UML1.Xからの変更点

UML2.0のパッケージ図で、UML1.Xから変更されたもののうち大きなものは、マージ (<<merge>> のステレオタイプが付けられた依存関係) が追加されたことである。これは2つの同じ名前のモデル要素を1つのモデル要素にまとめる場合に利用する。

この場合、2つの要素をひとつにまとめるとは、マージ元からマージ先へモデルを継承することを意味す

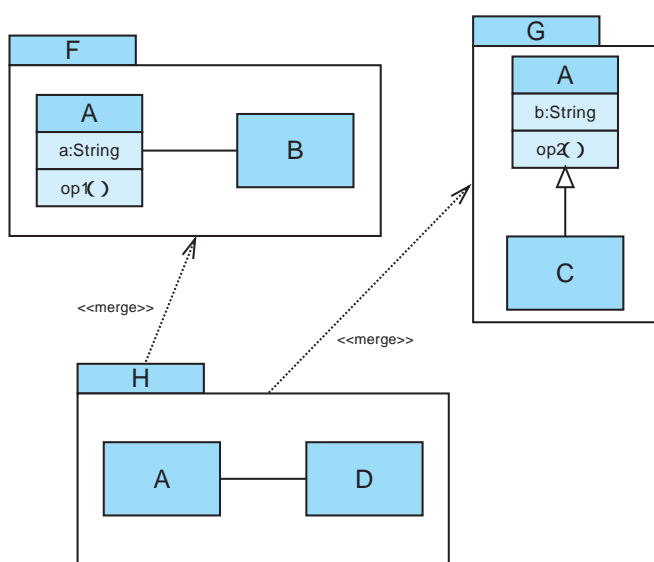


図4-a パッケージのマージ例

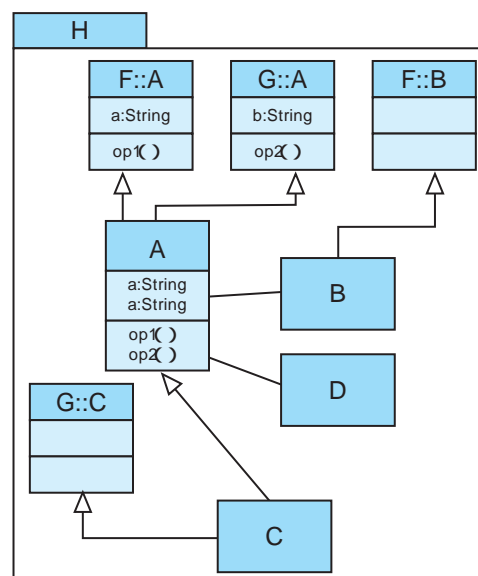


図4-b 図aをクラス図表記

る。マージの例を図4に示す。

ここでは、パッケージFとパッケージGのクラスAから、パッケージHのクラスAが多重継承によって作られる様子を示している。図4-aのパッケージHは図4-bのクラスAと同じ意味である。

このように、マージを使った場合、クラス間の関連は保持されることがわかる。

また、マージの追加に伴い、UML1.X系で意味が曖昧であった“パッケージの継承”は削除された。

#### 4 オブジェクト図

オブジェクト図はUML1.Xでも利用することができたが、UML2.0においても特に変更なく利用するこ

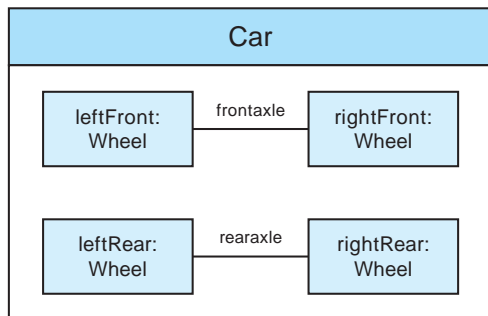


図 6-a コンポジットストラクチャー図

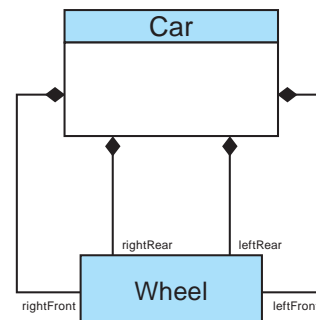


図 6-b 図-aのクラス図標記例

とができる。ここでは、オブジェクト図の概要を紹介する。

オブジェクト図はクラス図から派生したものである。記法はクラス図と後述する2つの例外を除いてほぼ同じである。オブジェクト図はプログラムのある実行時におけるオブジェクト間の関係を具体的に示したものである。これによって、抽象的理解しがたいクラス図を、実行時の

具体例を用いて分かりやすく表記することができる。

オブジェクト図とクラス図の2つの相違点とは、オブジェクトの表記に名前とクラス名を示し、それらにアンダースコアを付けること、および全てのインスタンスの関連を示すことである。図5にクラス図とそれに対応するオブジェクト図を記述した例を示す。

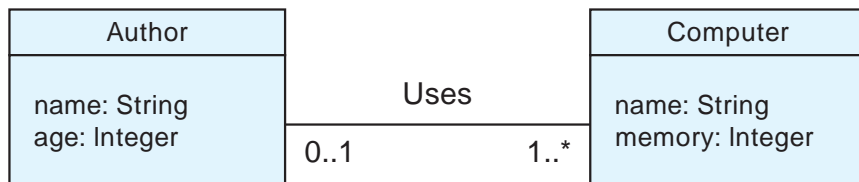


図 5-a クラス図

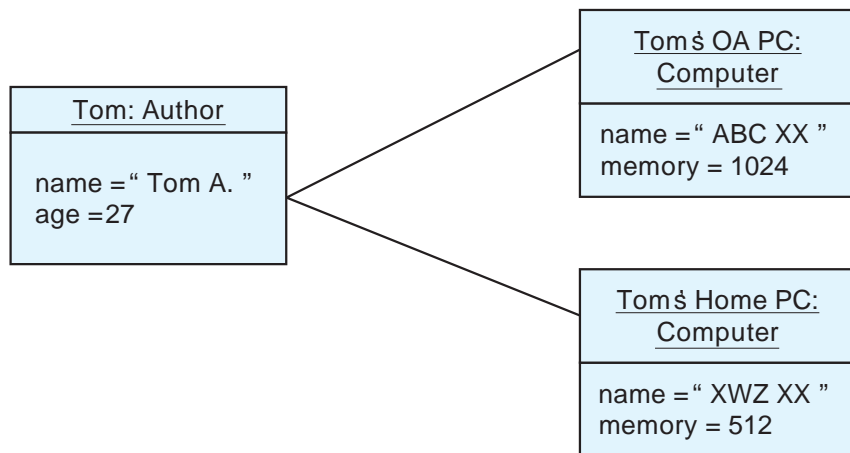


図 5-b 図-aのオブジェクト図の例

この例では、クラス図ではAuthorクラスとComputerクラスが“Uses”関連によって関連付けられている。一方、オブジェクト図ではAuthorのオブジェクトBobと、ComputerのオブジェクトであるBob's OA PCとBob's Home PCの2つのオブジェクトが関連付けられている。

#### 5 コンポジットストラクチャー図

コンポジットストラクチャー図はUML2.0で新たに加えられたダイアグラムである。ここでは、コンポジットストラクチャー図の概要を紹介する。

##### (1) コンポジットストラクチャー図とは

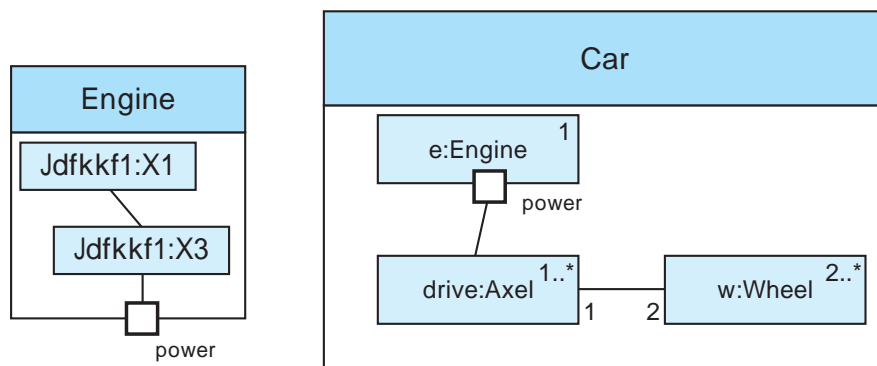


図 7-a コンポジットストラクチャー図

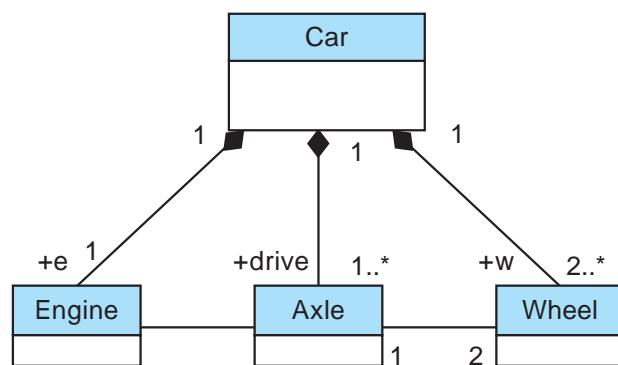


図 7-b 図 a のクラス図表記

コンポジットストラクチャー図は、複数の要素の関係を表し、システムの構造を表記するために利用するものである。

この図は、実行時の（実在的な）構造を表す。ただ、実行時のスナップショットではなく構造を表している。つまり、クラスインスタンスとは異なる概念である。

なお、構造化できる要素はクラス、コンポーネント、コラボレーションで、主に以下の目的で利用される。

- ・アーキテクチャやコラボレーション（協調）などシステム全体の構造の記述
- ・クラスやコンポーネントの内部構造の記述

コンポジットストラクチャー図の

例を図 6 -a に示す。この例では、Carクラスの内部にWheelクラスインスタンスがleftFront、rightFront、leftRear、rightRearの4つのインスタンスとしてCarクラスを構成していることが分かる。また、この図 6 -a をクラス図表記したものを図 6 -b に示す。これを見ると分かるように、図 b に比べて、図 a の方がCarクラスとWheelクラス間の関係およびWheelクラス間の関係が理解しやすくなっている。以上から、UML1.Xによっても記述可能なクラス図の構造が、コンポジットストラクチャー図を用いることで、内部構造を理解しやすく記述することができるようになる。

## (2) 主要要素の記法

コンポジットストラクチャー図には独自の図があるわけでない。構造化できる要素として、クラス図、コンポーネント図、コラボレーション図のクラス、コンポーネント、コラボレーションに次に示す三つの要素 (notation) を追加して記述する。

### パート (Part)

全体を構成する一つ一つの要素である。UML1.Xにおける合成集約 (Composition aggregation、whole-part関係) における part に相当する。パートはクラスやコンポーネントの内部に矩形で表記される。

### コネクタ (Connector)

パート間の関係を明記する（と共に役割の明確化も実現する）ための関連要素である。コネクタは、パート間を結ぶ実線で表される。

### ポート (Port)

パート間あるいは、外部とのやり取りを行うためのポイントを表す。外部に公開するインターフェースの意味もあるが、インターフェースほど厳密ではなく、UML1.Xにおける役割 (role) に近い。また外部へ要求するサービスも含む。さらに1つのパートが複数ポートを持つことも認められている。ポートは、小さな正方形をパートやクラスの境界線上に配置して記述される。

上記3つの要素を使ってクラスの内部構造をコンポジットストラクチャー図によって記述した例を図 7 -a に示す。また、そのクラス図を図

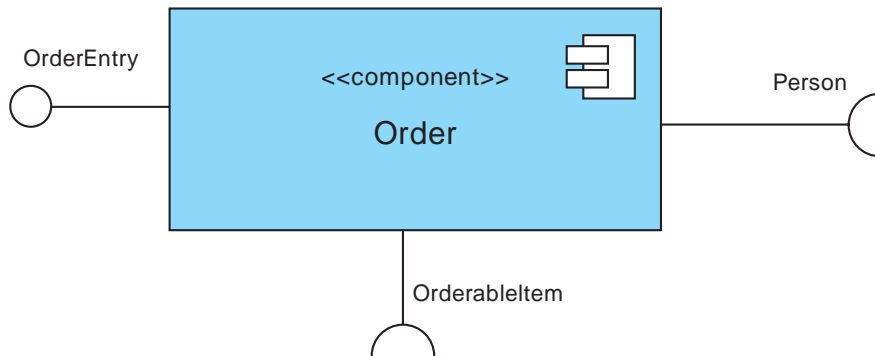


図8 コンポーネント図

7-bに示す。

ここでは、Carクラスの内部構造および、Carクラス内のパートの一つであるEngineクラスの内部構造を示している。

## 6 コンポーネント図

UML1.XからUML2.0で大幅に改定されたのがコンポーネント図である。ここでは、変更点を中心にコンポーネント図の概要を紹介する。

### (1) コンポーネント図の概要

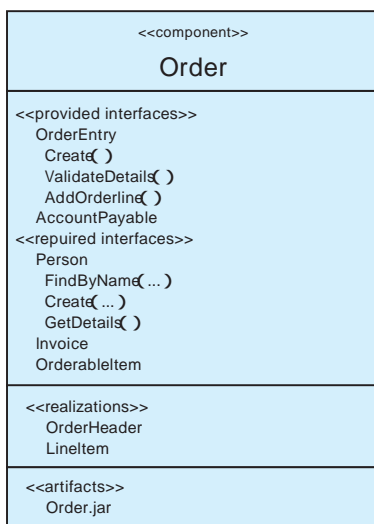


図9 インタフェースを明示的に書くコンポーネント図の例

コンポーネント図はアーキテクチャを記述するために利用する。UML1.Xでは、物理的な実装を表すために利用されていたが、UML2.0ではサブシステム構造を表すために利用することを想定している。

コンポーネントを用いてシステムを分割する目的は次の3つである。

- ・開発を容易にする
- ・問題をサブシステム内に閉じ込める
- ・再利用性が向上する

システムをサブシステムに分割して上記を実現するためには、それぞれのコンポーネントがどのようなサービスをどのようなインタフェースで提供するのかを定義しなければならない。

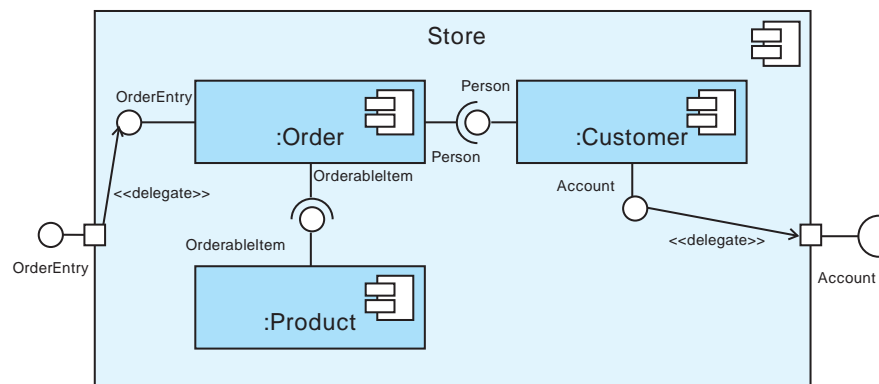


図10 入れ子構造のコンポーネント図の例

### (2) 記法

コンポーネントは矩形で表し、矩形の上部にコンポーネント名を書く。コンポーネントであることを表すために、矩形の上部右側にコンポーネントのアイコンを書くか、またはコンポーネントの上部に<<component>>のステレオタイプを表記する(両方記述しても良い)。

さらに、コンポーネント図では、コンポーネントが提供するインタフェース名と要求するインタフェース名をそれぞれ、ボールおよびロリポップ(半円形)のアイコンを利用して表記する。このコンポーネント図の例を図8に示す。

また、コンポーネント図は、要求するインタフェース、提供するインタフェースを明示的に記述することも可能である。この場合のコンポーネント図の例を図9に示す。

コンポーネント図は、入れ子構造にすることも可能である。この場合コンポーネントに含まれるコンポーネントを矩形内に配置して、利用関係を明示する。コンポーネントの入れ子構造の例を図10に示す。

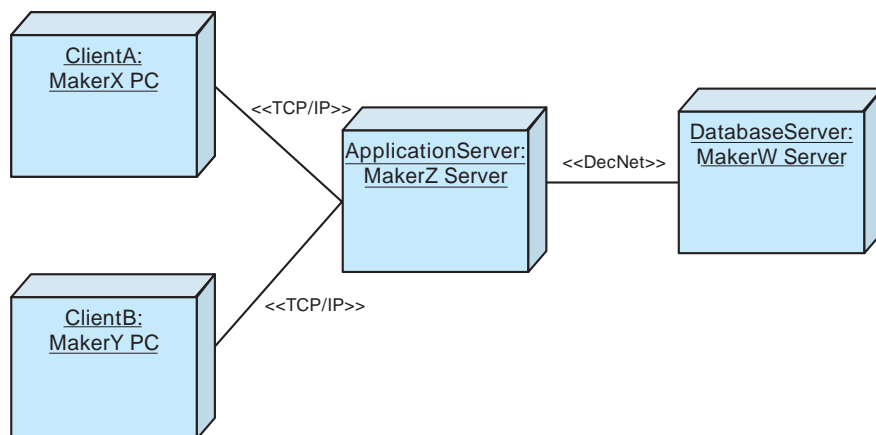


図 11 配置図

## 7 配置図

### (1) 配置図の概要

配置図は実行時のシステムの物理的なアーキテクチャを表す場合に利用する。このアーキテクチャは、コンピュータやプロセッサといった計算装置からカードリーダーやモバイル装置、通信装置といったものも表すことができる。

さらに配置図では物理的なデバイスを含んだ実行環境上に成果物（プログラムの実行コードなど）を配置した様子も記述することができる。

### (2) 記法

配置図を記述する場合には、次の3つを利用する。

- ・アーティファクト
- ・ノード
- ・通信パス

アーティファクトとはノード上に配置される物理的なファイルやソースコード、スクリプト、実行コードの

ことを表す。アーティファクトは、矩形表記して上部に名前とその上にステレオタイプ<<artifact>>を記す。また、上部右側に、ノート型のアイコンをつけることでアーティファクトであることを示す。ノードはアーティファクトを配置することができる物理的な計算装置や周辺機器を表す。ノードは直方体で表記する。

ノードは型およびインスタンスとして記述される。型の場合は型名を直方体図形の上部に記述する。インスタンスの場合は、名前と型名を記述してそれらに下線を引く。

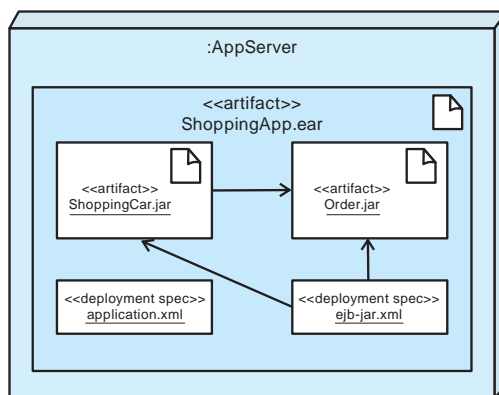


図 12 アーティファクトの記述例

ノード間は通信パスで接続することができる。通信パスはノード間を結ぶ線で表記される。通信パスにプロトコルを明記する場合はステレオタイプで表すことができる。配置図の例を図11に示す。

配置されたコンポーネントにおいて、実行パラメータが必要になる場合がある。そのアーティファクトを<<deployment spec>>というステレオタイプを用いて表すことができる。この例を図12に示す。

## 8 おわりに

今回は、UML2.0の静的構造について、UML1.Xとの違いを中心に説明した。次回は、振る舞い図の1回目として相互作用図を紹介する。

### 参考文献

- [1] 山本修一郎,UML の基礎と応用(連載記事)ビジネスコミュニケーション,Vol.38, No.9, 2001- Vol.40, No.3, 2003
- [2] H.Eriksson, M.Penker, B.Lyons, D.Fado, UML2 Toolkit, Wiley, 2003
- [3] OMG, UML2.0 Superstructure Final Adopted specification, <http://www.omg.org/UML>, 2003