付 録A　UML(統一モデル化言語)の表記について

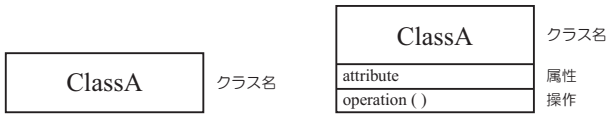
本論文では、モデル・フレームワークやシミュレーション・プラットフォームの構造、そしてシミュレーションモデルを記述するために、UML(Unified Modeling Language: 統一モデル化言語) を用いている。ここでは、この UML の表記法について、本論文に関係する部分のみを説明することにしたい (ここで説明する以外の項目や詳細については、Rumbaugh et al. (1999) などを参照してほしい)。以下では、クラス図、オブジェクト図、シーケンス図、ステートチャート図の順に説明する。

A.1　クラス図の記法

クラス図 (class diagram) は、おもにクラスの内容とそれらの間の関係を表すための静的ビューの図である。

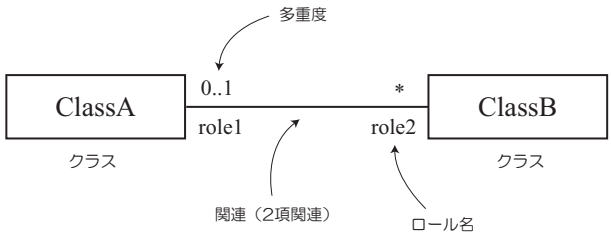
クラス

クラス図では、矩形で「クラス」が記述される。その矩形の中に「クラス名」が明示される。また、矩形の中に区画を区切って、そのクラスの「属性」と「操作」を示すこともできる。その場合には、上の区画にクラス名、中の区画に属性、下の区画に操作が記述される。これらの属性や操作は、どちらか一方を省略することもできる。

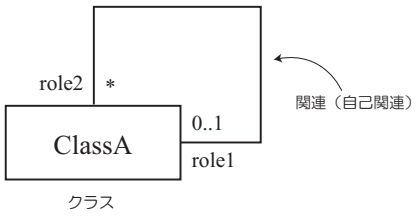


関連

クラス図において、クラスの結びつきを表す「関連」は、クラスの矩形の間の実線で記述する。関連がクラスに接続している部分には、「ロール名」や「多重度」を書くことができる。多重度とは、一方のクラスの１つのインスタンスに対して、もう一方のクラスのインスタンスが、いくつリンクを持つかを表す数である。

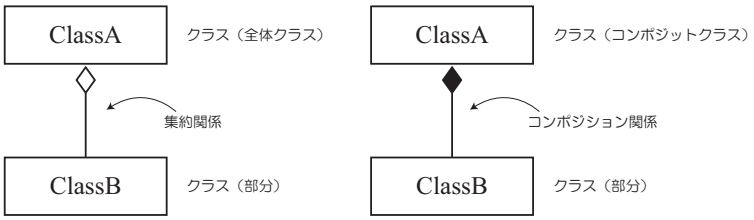


あるクラスから出ている関連がそのクラス自身に向けられている場合には、そのオブジェクトがそのオブジェクト自身にリンクされることもあれば、同じクラスのインスタンスである別々のオブジェクトがリンクされることもある。



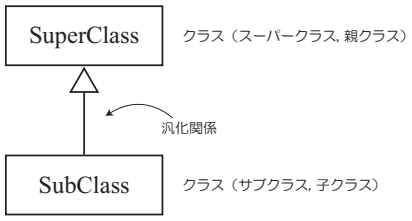
集約とコンポジション

クラス間関係において部分/全体関係を表す「集約」関係は、全体側のクラスに結び付けられる端に白抜きの菱形をつけた実線で表す。また、「コンポジション」(複合化) 関係は、塗りつぶした菱形をコンポジット側の端につけた実線で表す。

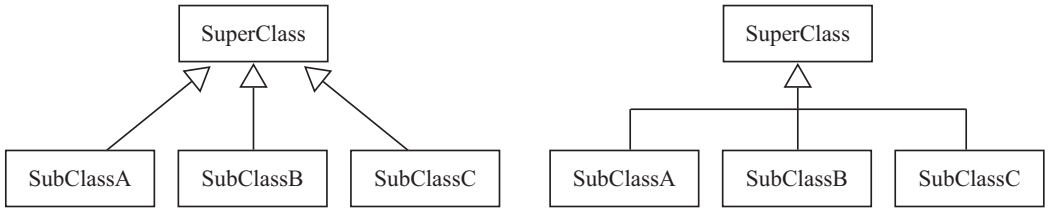


汎化

「汎化」関係は、スーパークラス側の端に白抜きの三角形をつけた実線で表す。



汎化関係や集約関係、コンポジション関係などは、いくつかの関係を 1 本にまとめて、複数に枝分かれするツリーで表すこともできる。一般に、図が体系的になって見やすくなるため、ツリーで表されることが多い。

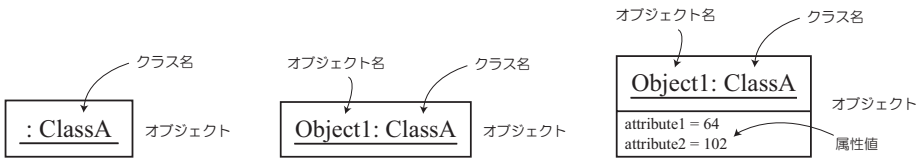


A.2　オブジェクト図の記法

オブジェクト図 (object diagram) は、ある時点でのシステムのスナップショット・イメージである。注意が必要なのは、オブジェクト図はシステムの 1 状態を表す例に過ぎず、システムの定義ではないという点である。

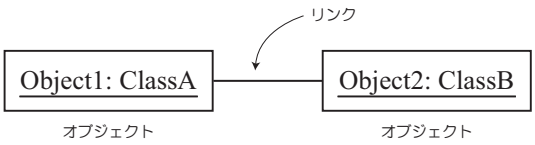
オブジェクト

オブジェクトは、クラスと同じ矩形を用い、名前に下線を引いて表す。どのクラスのインスタンスであるかを表すために「クラス名」が記述され、その名前の前にコロン (:) を書く。これで、そのクラス名のクラスから生成されたオブジェクトであるということを表す。オブジェクト名を明示することもでき、その場合には、コロンの前にオブジェクト名を書く。また、オブジェクトは具体的な属性の値をもっているので、それを明示する必要がある場合には、区画を区切って表示することができる。



リンク

オブジェクトとオブジェクトの個々の関係は、リンクによって表される。リンクは、関連のインスタンスである。リンクは、オブジェクトの矩形を結ぶ実線で表される。



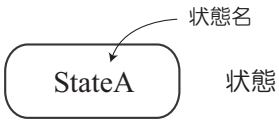
A.3　ステートチャート図の記法

ステートチャート図 (statechart diagram) は、オブジェクトの振る舞いを記述するために用いられる図である。イベントの反応の結果、どのように状態が変化するかの状態動作系列を表す。

状態機械を、状態 (state) を状態シンボルで、遷移 (transition) をそれらをつぐ矢印で表される。そこには、そのオブジェクトの振る舞いのすべての可能性が書かれている。これは１つのオブジェクトに着目して記述するので、局所化されたビューである。

状態 (state)

状態は丸み付き四角で表す。



初期状態 (initial state)

初期状態は、ふつうの状態への遷移をもつ擬似状態 (pseudostate) である。擬似状態 (pseudostate) とは、完全な状態としての振る舞いは行なわないという意味であり、オブジェクトは初期状態にとどまることはできず、直ちに遷移する。初期状態は、トリガーなし遷移 (triggerless transition) をもっており、必ず 1 つ出ていくことが保証されている必要がある。

初期状態は、塗りつぶした小さい黒丸として表示される。

最終状態 (final state)

最終状態 (final state) は、その実行が完了したことを示す状態である。そのため、最終状態からはイベントによりトリガーされて出ていく遷移をもつことはできない。最終状態は、初期状態のような擬似状態ではなく、一定期間アクティブになることが可能である。

最終状態は、標的アイコン、すなわち小さい白丸で囲まれた小さい黒丸で表記する。

遷移 (transition)

遷移は、ある状態からどのイベントに対して発火し、どのようなガード条件を満たし、どのようなアクションを実行し、どの状態に移るのか、ということを指定する。ステートチャート図では、遷移はソース状態からターゲット状態に至る実践の矢印で示す。

遷移の矢印には、次の形式の遷移文字列を記述する。

イベント名 [ガード条件] / アクション式

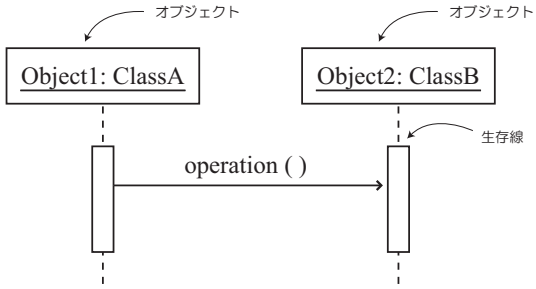
ガード条件はトリガーイベントのパラメータやその状態機械をもつオブジェクトの属性や関連についての論理式である。

遷移には、アクションを記述するアクション式 (action expression) が含まれる。これは、その状態機械をもつオブジェクトに影響を及ぼす手続き的計算を表す式であり、属性値を変更したり、オブジェクトの生成や計算、他のオブジェクトへのイベントの送信などが行なわれる。アクション式の構文は、UML では仕様化されていないため、プログラミング言語、擬似コード、自然言語などを用いてアクションを表現する。

A.4 シーケンス図の記法

シーケンス図 (sequence diagram) は、オブジェクトの相互作用を時系列に表すための図である。注意が必要なのは、シーケンス図は、可能なシナリオのうちの 1 つのシナリオ例を記述するためのものであって、システムの定義ではないという点である。

シーケンス図では、通常、横に相互作用するオブジェクトを並べ、縦に上から下に進む時間軸を設定する。図の一番上の部分には、オブジェクトが書かれる。オブジェクトの配置の順番には意味はなく、わかりやすい配置がとられる。オブジェクトの矩形から、そのオブジェクトが消滅する時点まで、下方に破線が引かれる。これを生存線という。あるオブジェクトから他のオブジェクトへの作用は、作用するオブジェクトの生存線から、作用されるオブジェクトの生存線への矢印で示される。



生成と消滅を伴うシーケンス図を書くこともできる。新しいオブジェクトの生成は、塗りつぶした三角形を先端にもつ矢印で表す。また、オブジェクトの消滅は、×印で表す。