5.2 状態図

状態図ではオブジェクト、サブシステム、システムのライフ・サイクルを捉える。その状態図はオブジェクトが取りうる状態を表し、時間とともにイベント（メッセージの受信、時間経過、エラー、真になる条件）がその状態にどのように影響を及ぼすかを示す。はっきり認識できる状態と複雑な振る舞いを持つクラスには、すべて状態図を付けるべきである。というのは、状態図は振る舞いと、現在の状態に応じて振る舞いがどのように異なるかを表せるからである。また、状態図は、どのイベントがクラスのオブジェクトの状態を変えるかもダイアグラムに表すことができる。

状態と遷移

すべてのオブジェクトには状態がある。状態は、オブジェクトが処理した、前の動作の結果であり、一般的にその属性値と他のオブジェクトへのリンクで決まるものである。状態を特定するための特定の属性をクラスに持たせることができる。あるいは、オブジェクトの「ノーマル」の属性値から状態を決めることができる。オブジェクトの状態の例をあげよう。

- 請求書（オブジェクト）が支払われる（状態）。
- 車（オブジェクト）がまだ止まっている（状態）。
- エンジン（オブジェクト）が動いている（状態）。
- ジム（オブジェクト）がセールスマンの役を演じている（状態）。
- ケイト（オブジェクト）は結婚している（状態）。

オブジェクトの状態が変わるのはイベントが起きた時である。例えば、請求書を支払う、車を動かす、結婚するというようなものである。動作には2つの次元があって、それは、相互作用と内部状態の変化である。相互作用はオブジェクトの外部的な振る舞いと、どのように他のオブジェトに作用（メッセージを送ったり、互いにリンクしたり、リンクをはずしたりすることによる）するかを説明する。内部状態の変化によって、オブジェクトが状態、例えば、内部の属性値などをどのように変えようとしているのかが分かる。

状態図はオブジェクトがイベントにどのように反応して、そのオブジェクトの内部の状態を変えるかを表す。例えば、請求書が支払われると、その請求書は、未払いから支払い済みに変わる。請求書が作成される時は、その請求書は未払の状態になる（図5.2を参照）。

状態図には始点と終点があってよい。始点（最初の状態）は黒丸で表され、終点（最後の状態）は小さな黒丸を囲んだ丸（目玉）で表される。状態図の中の状態は角が丸くなった矩形で表現される。状態の間は状態遷移であり、1つの状態から別の状態に向かう矢印

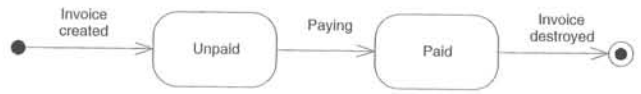

図5.2 請求書（Invoice）の状態図。黒丸は請求書（Invoice）始点を表す（オブジェクトが作られる）。黒丸を囲む丸は終点を表す（オブジェクトの破壊）。状態間の矢印は状態遷移とそれを起こすイベントである

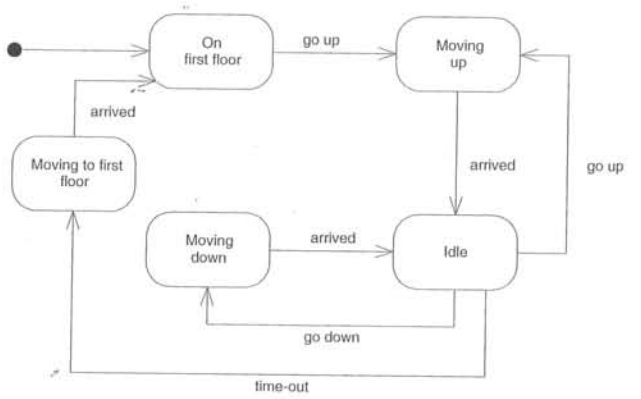

図5.3 エレベーターの状態図。エレベーターはOn First floorで始まり、上がり降りする。ある階でidleになると一定時間でタイムアウトのイベントが発生し、On first fllorに戻る。この状態図は終点（最終状態）を持たない

の付いた線で表される。状態遷移には図5.3に示すように状態遷移を引き起こすイベントにラベルを貼り付けることができる。イベントが起きるとある状態から他の状態へ遷移する（遷移が「発生する」とか「引き起こされる」といわれることもある）。

状態は図5.4に示すように、3種類の区画を持つ場合がある。最初の区画は状態の名前を示す。例えば、アイドル（idle）、支払い済み（paid）、移動中（moving）などである。2番目の区画は任意の状態変数区画であり、属性（変数）を列挙し、設定することができる。属性とは、状態図により表されるクラスの属性のことである。各状態で一時的な変数が役に立つこともある。例えばカウンターのような。3番目の区画は任意のアクティビティ区画であり、イベントとアクションを列挙することができる。アクティビティ区画には3つの標準的なイベント**entry**、**exit**、**do**を使うことができる。entryイベントを使って、状態の入り口で起きるアクションを示すことができる。例えば、属性の設定とメッセージ送信などがある。exitイベントを使って状態から出る時に起こるアクションを示すことができる。doイベントを使って、状態に留まっている間に起きるアクションを示すことが