第9講　UMLの概要（5）

- アクティビティ図
- ユースケース図
- シーケンス図

アクティビティー図

- プロセスの逐次的なステップを表現するタメの図
- さまざまな処理の流れを記述
- フローチャートによく似ている
- フローチャートとの違いは、並行処理が表現できる点にある
- ものごとの進行や経過を図示するために利用

アクティビティ図の要素（1）

- アクティビティ
  - 一連のプロセスの中でおこなわれる振る舞いをまとめて表現
  - 四角形の角を丸めたような記号を用いて表現し、その内側にアクティビティ名を記述
  - アクティビティが含んでいるさらに詳細な内容を記述する場合には、四角形を大きくし、アクティビティ名を左上に記載して詳細内容を記述する
  - 詳細内容を別途用意し、図面全体をすっきりさせたい場合には、アクティビティに熊手のようなアイコンを付けて、「詳細な内容を含んでいる」ということを示す

アクティビティ図の要素（2）

- アクション
  - アクティビティと同様に、四角形の角を丸めたような記号を用いて表現し、その内側にアクション名を記述
  - 一連のプロセスの中でおこなわれる振る舞いの中でも最も粒度の小さな振る舞いを表現
  - アクションは振る舞いの最小単位。その内部にさらにアクションを含めることはできない
  - アクティビティやアクションの名称には、「動詞」を使った言葉が用いられるのが一般的

アクティビティ図の要素（3）

- コントロールフロー
  - アクティビティやアクションの流れを表現
  - アクティビティやアクションの間にひく矢印
  - コントロールフローに名称を付けたい場合には、その矢印の付近に名称を付記
- オブジェクトフロー
  - コントロールフローと同じ実線に「＜」の字型をした先端をもつ矢印を用いるが、流れを表現する対象が異なる

アクティビティ図の要素（4）

- 開始ノード
  - 一連のプロセスの開始を表現
  - 開始ノードは黒く塗りつぶされた円で表現
- 終了ノード
  - 一連のプロセスの終了を表現
  - 終了ノードはその終了の仕方によって2種類ある
    - アクティビティ終了ノード
      - アクティビティ図で表されている一連のアクティビティが完全に終了することを表現
    - フロー終了ノード
      - エラーによる異常終了やアクティビティ図の中の繰り返し処理の終了などによるフローの終了地点を表す

アクティビティ図の要素（5）

- デシジョンノード
  - 処理の流れが2つ以上に分かれる場合の分岐点を示す要素
  - 分岐の経路には[]で囲まれた「ガード条件」と呼ばれる条件を記述し、その条件が満たされる方向に処理が流れる
- マージノード
  - デシジョンノードとは反対に、2つ以上の処理の流れを統合することを示す要素
- フォークノード
  - 2つ以上の処理を同時並行的に実行したい場合に使用
  - フォークノード以降に同時並行的に実行された処理はジョインノードで処理の完了を待ち合わせる必要がある
- ジョインノード
  - フォークノードによる同時並行的に処理されている処理の完了を待ち合わせ、同期させてから次の処理へ遷移することを表現

アクティビティ図の要素（6）

- アクティビティパーティション
  - 縦もしくは横に大きく幅をとった長方形と、その端に名称を書く長方形をラベルのように配して表現
  - 一連のプロセスをグループに分け、名前をつけて整理するための要素
  - アクティビティパーティション内に記述された一連のプロセスを実行する主体や役割をアクティビティパーティションの名称としてつければ、「誰が（どの役割が）」「どんな振る舞いをするのか」をすっきりと表現できるようになる

アクティビティ図の要素（7）

- 例外ハンドラ
  - あるアクティビティの実行中に例外が発生した場合に、対応する処理を記述する
  - 例外ハンドラと、例外時の振る舞いを定義しておきたいアクティビティとは稲妻のようなジグザグの実線に「く」の字型の先端を持つ矢印で結ぶ
  - 小さな四角形のそばには、例外の型名を付記する

ユースケース図 （1）

- システムに必要とされる機能や要件を分析し、その結果を表すために利用
- 「システム」と「システムの利用者」の間のやり取りの様子をあらわす
- ユースケース図の構成要素
  - ユースケース
  - アクター
  - 境界枠
  - 関連
  - 汎化
  - 包含
  - 拡張

ユースケース図 （2）

- ユースケース
  - 外部から見たシステムの機能
  - システムの持つ内部的な機能ではない
  - 個々のユースケースが内部的にどのように実現されるかについては触れない
  - 内部にユースケース名を書き込んだ楕円形のシンボルとして表現
  - ユースケース群の周囲は、システム境界を表す四角形で囲んでおくことができ、このシステム境界は、ユースケース群を含むシステムを表す

ユースケース図 （3）

- アクター
  - システムの外部利用者を表す
  - 人であったり、システムと接続された別のシステムや、システムで制御されるハードウエアを表す
  - つまり、ユースケース図で表現するシステムを外部から利用するもの
  - 人型のアイコンや四角形の中にアクター名を書き、その上にステレオタイプ表記<<actor>>したもので表現
  - 役割を代表するものであり、個々の「実体」を表現するものではない

ユースケース図 （4）

- 境界枠
  - 対象となるシステムの内部と外部の境界を表す
  - 境界枠は長方形で表現し、システム名の上部に記述する
  - システムの機能であるユースケースは、境界枠の内側に、利用者であるアクターは外側に配置する
- 関連
  - アクターがそのユースケースに関わっていることを示す
  - ユースケースという「機能」をそのアクターが「利用する」ことを表現
  - 関連は、アクターとユースケースを実線で結んで表現

ユースケース図 （5）

- 汎化
  - ユースケースやアクター間の抽象－具象の関係を意味する
  - 汎化は、具象要素から抽象要素に向かって実線を引き、抽象要素の側に白抜きの矢印を付けて表現
- 包含
  - あるユースケースが別のユースケースを内部に含めことを意味する
  - 包含は、包含するユースケースから包含されるユースケースに向かって点線の矢印を引き、ステレオタイプ<<include>>を付加して表現
- 拡張
  - すでに存在するユースケースに対して、新たな機能を追加することを意味する
  - 拡張は、拡張するユースケースから基となるユースケースに向かって点線の矢印を引き、ステレオタイプ<<extend>>を付加して表す

シーケンス図 （1）

- オブジェクト間におけるメッセージのやりとりを記述するに利用
- オブジェクト間のメッセージを時系列で表現（作業の進行表）
- どのようなオブジェクトがどのようなメッセージをどういう順序でやり取りしあうかをわかりやすくまとめることが可能
- シーケンス図の構成要素
  - オブジェクトシンボル
  - ライフライン
  - メッセージ
  - テキストシンボル

シーケンス図 （2）

- シーケンス図の構成要素
  - 相互作用
  - ライフライン
  - メッセージ
  - 実行仕様
  - 相互作用使用
  - 複合フラグメント

シーケンス図 （2）

- シーケンス図の構造
  - 水平と垂直の二次元の軸により表現
    - 水平軸
      - シーケンス図が表す場面に登場するオブジェクト
      - 各オブジェクトがやり取りするメッセージ
    - 垂直軸
      - 時間の経過を表す
      - 図の上方に並べられた各オブジェクトが生成されてから消滅するまでの時間の流れを表現
      - 時間は図の上から下に向かって進む

シーケンス図 （3）

- 相互作用
  - システムが外部に提供するサービスや機能を表現する処理の単位
  - 相互作用は長方形のフレームで表現し、左上隅に「sd」という文字（sequence diagramの略）をつけた相互作用名を記述
- ライフライン
  - 相互作用へ参加する要素を表したもの
  - シーケンス図中に登場するオブジェクト、クラス、コンポーネント、アクタなどと、そこから下方に向けて引かれた破線
  - 破線は、各ライフラインが生存している期間を示している
  - ライフラインの生成は、生成するライフラインへ直接メッセージを送ることで表現
  - ライフラインの停止は、破線上にターミネーションアイコン（×）を付けることで表現

シーケンス図 （4）

- メッセージ
  - ライフライン同士のやりとりを表現
  - あるライフラインから別のライフラインに向かって、ライフラインに垂直に引かれている矢印
    - 同期メッセージ
      - 閉じ実線矢印
      - 呼び出し先の操作が終了するまで、呼び出し元は次のステップに進めない
    - 非同期メッセージ
      - 開き実線矢印
      - 呼び出し先の操作が終了していなくても、呼び出し元が次のステップに進める
    - 同期メッセージへのリターン
      - 開き破線矢印
    - 自己メッセージ
      - ライフラインは自分自身の操作を呼び出すために、自分自身にメッセージを送ることができる
      - 文字の「コ」のように記述することによってメッセージを表現

シーケンス図 （5）

- 実行仕様
  - ライフライン上である動作が持続している区間を表す
  - 一般に、実行仕様はメッセージの受信から始まり、そのメッセージによって起動された操作が終了するまでの間となる
  - 表記法としては、ライフライン上に灰色もしくは白色の長方形を配置する

シーケンス図 （6）

- 相互作用仕様
  - 別の相互作用の参照を表す
  - 相互作用使用を利用することで、大規模なシーケンスズの一部を別のシーケンス図として表現したり、共通の処理を1つのシーケンス図として切り出し、複数のシーケンス図で共用したりできる
  - 長方形のフレームで表現するが、フレーム左隅上には「ref」という文字（referenceの略）をつけた相互作用名を記述
  - 相互作用名は、参照先の相互作用名と一致させる必要がある

シーケンス図 （7）

- 複合フラグメント
  - 相互作用の流れを簡潔に表現できるようにするための要素
  - １２種類の要素が定義
  - 複合フラグメントも、相互作用と同様に、長方形のフレームで表現し、左上隅に処理の意味を記述

シーケンス図 （8）

- 複合フラグメント
  - オルタナティブ（alt）
    - 相互作用の分岐を表現
    - 分岐後のそれぞれの流れは破線で区切った上下の領域内に記述
    - それぞれの領域に対応する分岐条件が、ガード条件で記述
  - オプション（opt）
    - ある条件が成立する場合だけに実行される相互作用を表現
    - 実行するための条件は、ガード条件で記述
  - ループ（loop）
    - 対象となる相互作用を繰り返し実行することを表現
    - 繰り返し条件は、左上隅に記述するloopに続いて、minint, maxint、ガード条件の順にカンマ区切りで記述
  - ブレイク（break）
    - 複合フラグメントを中止することを表現
    - 処理の途中で複合フラグメントを抜け出したい場合に利用
    - ブレイクを実行するための条件もガード条件で記述
  - パラレル（par）
    - 相互作用を並行して実行することを表現
    - 並行しておこなう相互作用の処理順は問わない

シーケンス図 （9）

- 複合フラグメント
  - クリティカル領域（critical）
    - 他の処理に割り込みを行わせたくないことを表現
    - この複合フラグメント内では、他の処理からの割り込みを受けずに処理を実行
  - 弱シーケンス（seq）
    - 以下の条件を満たせば、相互作用の順番が入れ替わってもかまわないことを意味する
      - 領域内のメッセージの順序は最終的に保たれる
      - 異なるライフラインから送信されるメッセージは、どのような順番でもよい
      - 同じライフラインから送信されるメッセージは順番が保たれる
  - 強シーケンス（strict）
    - 複合フラグメント内のメッセージの順番が厳密に保たれる必要があることを表現
    - 先行するメッセージの返り値が、次のメッセージの引数になっている場合など、メッセージの順番が保たれていないと動作しないことを強調する場合に用いる
  - アサーション（assert）
    - シーケンス図をテストする場合に使用
    - テスト対象となるメッセージをフラグメントで囲い、ライフラインが満たすべき条件をライフライン上に記載する

シーケンス図 （10）

- 複合フラグメント
  - 無効（ignore）
    - 処理の流れから一時的に除外しておきたいメッセージを表現
    - 開発途中で不完全な場合や、信頼性の低いメッセージを実行させないようにする場合に利用
    - 除外するメッセージは、フレームの左上隅のignoreという単語の後ろに、{メッセージ名１，メッセージ名２，…}という形式で記述
  - 有効（consider）
    - 実行する処理の流れを表現
    - 開発途中に、一部の実装だけをテストしたい場合などに利用
    - フレームの左上隅のconsiderという単語の後ろに、{メッセージ名１，メッセージ名２，…}という形式で記述
  - 否定（neg）
    - 処理の流れから除外したいメッセージを表現
    - アルゴリズムの検討段階などで、一部の処理を削除したくないが、実行もさせたくない場合に利用