【講義メモ】 担当：平野正喜（ひらのまさき）
この講座ではプロジェクタに講義メモを書きながら進めます。この文字サイズの読める席にお座りください。
18:15～20:45（途中休憩有）。受講者数は１３人です。
この講義メモは講義終了と同時に下記のサイトにPDFで掲載し、ダウンロード可能にします。ご利用ください。次回予告も掲載します。質問やコメントが送信可能です。
https://tkuip.rundog.org
前回の1問： 提案依頼書は？ ア SOA イ ASP ウ RFI エ RFP 正解はエ
p.101「4-1-2 テストの種類」から
・【補足】単体テスト：プログラムまたはその部品であるモジュールごとのテストで、通常、作成者が行う。プログラミングと対応する位置のテスト。
・【補足】結合テスト：モジュールやプログラムを複数組み合わせた状態でのテスト。内部設計に対応し、内部設計と同様に、ベンダ側のみで行う。
・【補足】システムテスト：システム全体でのテスト。外部設計に対応し、ユーザ側の協力を得て行うことが多い。※ここまではテスト用のデータ（ダミーデータ）を用いる
・【補足】運用テスト：実際の運用状態（本番環境）でのテスト。要件定義に対応し、要件定義と同様に、ユーザ主導で行う。
・【補足】退行テスト：主に運用テストで行うもので、この開発や変更により、周辺に悪影響を与えていないかのテスト。レグレッションテスト、回帰テストともいう。
　※「退行」とは「進化」の反対
・ホワイト（透明な、中身の見える）ボックステスト：仕様書やプログラムリストを基に行うテストで、主に、単体テストや結合テストの前半の手法。基の資料に対してのテストの網羅率で進捗（進み遅れ）を評価する
・ブラック（中身の見えない）ボックステスト：設計書や要件定義書を基に行うテストで、機能が要件どうりかをチェックする。主に、結合テストの後半以降の手法。
・【補足】ゴンペルツ曲線：信頼度成長曲線ともいい、時系列でバグ（問題点）検出累積数をグラフ化したもの。システムのテストでは「まだ見つけていないバグがある可能性」をゼロにできないが、十分なテストを行い、新たなバグの検出が一定期間見られなければテストを終了できるというモデル。ゴンペルツ曲線が斜めS字になり先端が平らであれば該当する。
※ 急激に検出数が増える時期があるのが特徴で、これが無い場合、先端が平らになっても「テストが進んでいない状況」を判断される。
p.102 4-1-3 ソフトウェアの見積り
・【補足】ソフトウェアの見積り：テストの終了判断と同様に、ソフトウェア開発に必要なコスト（期間×人数）を予測する＝見積ることは難しい。そこでいくつかの手法が提案され、最も初期から用いられたのが「経験値による見積り」で技術の進化や複雑度の上昇などにより、誤差が大きくなりやすい。
・COCOMO＝コンストラクティブ（積算）コストモデル：システムの規模（処理数、行数）を予測し、開発者の能力などから計算した値で見積もる。正確な見積りにはなりづらい。
・FP＝ファンクション（機能）ポイント（点数）法：システムに必要な機能を、画面数、ファイル数、外部との接続数のような数えられる数で示し、これに種類に応じたウェイト（重み）掛けた値をFPとする。FPの合計を基にコストを算出する。よって、経験のない（浅い）システムでも、正確な見積りになりやすい。

p.103 4-2-1 おもなソフトウェア開発の手法

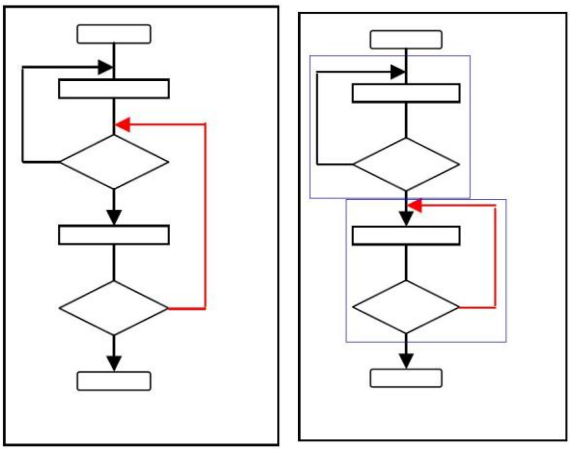
・【補足】初期の手法:「手順のプログラム化」だったので、手順が複雑な場合、絡み合う形になり、理解や改良が難しいものになりやすかった。

・【補足】 構造化手法:プログラムを「順次」「選択(分岐)」「繰返し(ループ)」の3構造(図はp.164)のみで構築する手法。可読性(読みやすさ)と保守性(改造しやすさ)が向上する。

・【参考】構造化手法による改良の一例

左:悪例=構造化されていない

右:判断◇や処理□の内容を見直して、2構造化したもの。



・【補足】オブジェクト指向:構造化手法までは「データ」と「処理」が別々で、作成者の考え方に依存しやすい。そこで、「データ」と「処理」をまとめた「オブジェクト」を単位としてシステムを構築する手法。

・【補足】プロパティ:オブジェクト指向におけるデータ

・【補足】メソッド:オブジェクト指向における「処理」

・【補足】クラス:オブジェクト指向におけるオブジェクトの設計図で、プロパティの形式や、メソッドの内容などを記述したものなので、プログラムに該当する

・【補足】インスタンス:クラスを基にしてメモリ上に生成される実行用の実体。実際のデータを持つ。例: RPGの場合、スライムやドラゴンはクラス。実体になるスラリンやリムルやヴェルドラはそれぞれのクラスから生成されたインスタンスとなる。

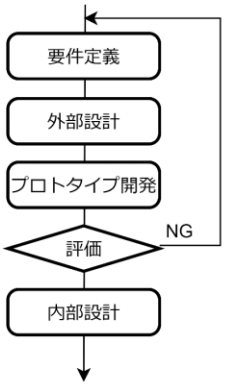
・【補足】カプセル化:データをインスタンスに持つことで、複数の関連するデータをまとめて扱える(例:スライムの名前、HP、MP)。加えて、これらを扱う仕組みをメソッドにすれば、不適切なデータ(例:マイナスの身長)の発生を防げること。バグの防止になる。

・【補足】クラス図:UMLに含まれる図法の一つで、目の形の枠の中に、クラス名、プロパティ(データ)、メソッド(処理)を記述する。

・【補足】 UML=ユニファイド(統合された)モデリングランゲージ(図法):オブジェクト指向の各種の概念や設計結果を記述するための統一図法群。

p.104 4-2-2 おもなソフトウェア開発モデル

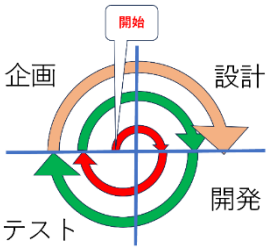
・【補足】ソフトウェア開発モデル:旧来のソフトウエア開発では開発中の手戻りや要件の不一致、開発期間の長期化、低品質になりやすかった。この対策としての開発方法論。



・【補足】ウォータフォールモデル:「滝のモデル」で手戻りの防止と、開発段階の明示が主目的。工程の完了時に行うレビューがポイント。大規模システム向きで長期化は避けられない。

・【補足】プロトタイピングモデル:「試作品モデル」で外部設計を終えたら試作品を作成して、ユーザに確認してもらうことで手戻りや要件の不一致を防止する手法。主に小規模システム向きでユーザの協力次第なのが懸念点。

・【補足】スパイラルモデル:「渦のモデル」でシステムをサブシステムに分割し、中核的なものから順に企画→設計→開発→テストを繰り返す。上記2つの折衷案。分割を誤ると破綻しやすい。



・RAD＝ラピッド（快速）アプリケーション・デベロップメント（開発）：開発ツールの利用、ユーザの協力や参画などにより開発期間を短縮する手法。小規模で低難度な場合に可能。
p.105 4-2-3 アジャイル
・【補足】アジャイル：主に小規模システムの開発を短いサイクルで反復することで、短期開発と環境への変化への対応を可能にする考え方。複数の手法がある。
・XP＝エクストリーム（先進的な）プログラミング：アジャイルを構成する手法群で、チームによる短期開発の要点と実践手法が含まれている。
・【補足】ペアプログラミング：XPの実践手法の1つで、プログラミングを2人1組で行うことで、作成時点で品質を上げるのが目的。
・DevOps＝デベロッパー（開発者）オペレータ（運用者）：開発チームと運用チームが開発時点から協力することで、短期開発と環境への変化への対応を可能にすること
・【補足】スクラム：計画から実装までをできるだけ小さな単位に分割したもの（スプリント）を、チームに割り当てて短期間で完成することを継続的に行う手法。
・【補足】プロダクトバックログ：実現が必要な要素のリストのこと（※アジャイル以外では、バックログというと「積み残し」「作業の山」と悪いイメージだが）
・【補足】スプリントバックログ：各スプリントで行う作業のリストのこと（スケジュールと同じ）
p.107 4-2-4 開発プロセスに関するフレームワーク
・SLCP＝ソフトウェア・ライフサイクル（生成から再利用／破棄への繰返し）プロセス：企画から開発・運用・保守までの一連の作業のことで、保守を得て前方の工程に戻るのでサイクル構造になるか、利用されなくなって廃棄される。
・【補足】共通フレーム：SLCPの用語と定義を標準化したガイドライン
・【補足】SLCPにおける開発プロセスの特徴：外部設計を「システム要件定義」と「システム方式設計」としている。内部設計を「ソフトウェア要件定義」「ソフトウェア方式設計」「ソフトウェア詳細設計」としている（※「ソフトウェア詳細設計」はプログラミングとする場合もある）。プログラミングと単体テストを「プログラム構築」としている。結合テストを「ソフトウェア結合」「ソフトウェア適格性確認テスト」、システムテストを「システム結合」、運用テストを「システム適格性確認テスト」としている。また、テスト後に「導入」「受入れ支援」がある。
・【補足】V字型モデル：同じ高さになる工程が対になり、同じ文書を作成・利用する。
・CMMI＝ケイパビリティ（成熟度）マチュリティ（成長）モデル・インテグレーション（統合）：主にソフトウェア開発チームや工程を改善するために、現在の段階と、今後の目標を示すもの。5段階あり、常に問題の分析・防止・改善が可能な最適化状態が最上位。

第5章 プロジェクトマネジメント

p.116 5-1-1 プロジェクトマネジメント
・【補足】 プロジェクト：既存の組織では対応が難しい、一時的な任務に対応する特命チーム。定常業務には対応せず、対応完了後には解散する（永続しない）。
・【補足】プロジェクトマネジメント：既存の組織における管理手法とは異なる「プロジェクト向けの管理手法群」
・【補足】プロジェクトマネージャ：予算管理などを行う管理者で、通常、プロジェクトリーダとは兼任しない
・PMBOK＝プロジェクトマネジメント・ボディ・オブ・ナレッジ（知識体系）：プロジェクト管理の技法や用語をまとめたもので、プロジェクト管理の国際標準になっている。プロジェクト管理を5つのプロセス群にまとめている。「終結プロセス群」があるのがポイント。

・【補足】プロジェクト憲章：プロジェクトの目的・背景・概略を記述したもの。立ち上げ時に作成。
・【補足】ステークホルダ：利害関係者。プロジェクトの場合、プロジェクトの影響を少しでも受ける全ての人や団体のこと。プロジェクト管理の対象。
・【補足】スコープ：元の意味は「視野（見える範囲）」。プロジェクトの場合、作業対象のことで。成果物や対象項目で示す。プロジェクト管理の対象。
・WBS＝ワーク（作業）ブレイクダウン（分解）ストラクチャ（構造）：スコープ（成果物や対象項目）を分析するために作成する階層図。
・ワーク（作業）パッケージ（単位）：WBSにおける最下層の作業単位で、管理可能（作業者への割り当てやスケジュール化など）が可能なレベルまで分割した結果。（※必要以上に分割しないことがポイント）
・【補足】統合マネジメント：すべてのプロセス群にかかわるマネジメント項目群。
・【補足】タイムマネジメント：主にスケジュール管理と作業時間の見積りのこと。
・【補足】資源マネジメント：主に人的資源＝プロジェクトチームの管理
・【補足】マイルストーン：元の意味は一里塚で、プロジェクトの進捗管理の目安となる地点＝イベントのこと。例：設計説明会開催日、テスト開始日（※期間のことではない）
・【補足】アクティビティ：マネジメント項目のこと（例：プロジェクト憲章の作成）
・PERT＝プログラム（工程表）エヴァリューション（評価）＆レビュー（確認）テクニック（技法）：アローダイヤグラムにより、工程の流れを評価・確認する技法。
・ガントチャート：アローダイヤグラムでは表現しづらい予実対比（予定と実績の差の対比）を明示できる図法。時系列の横棒グラフで項目ごとに予定と実績を示す。
・PMO＝プロジェクト・マネジメントオフィス（管理部門）：プロジェクトの終結時に成果物などを受け取る組織で、成果物の散逸を防ぎ、次のプロジェクトに活かす。プロジェクトのサポートも行う。

今日の1問：成熟度モデルは？ ア CMMI イ PMO ウ WBS エ RAD
次回予告：第6章「サービスマネジメント」から