Business Literacy

システム化計画の基礎知識

東京デザインテクノロジーセンター専門学校　講師　石川敢也

Systematization

- システム化計画
  - 業務の内容を分析して、最適な情報システムを設計し、開発、導入を進めための基本方針を計画するプロセス。
- 内容
  - ソフトウェアライフサイクル、企画プロセス
  - 要件定義、見積、検収、品質特性
  - プログラミング、コーディング、デバッグ
  - ブラックボックステスト、ホワイトボックステスト
  - 同値分割、限界値分析、単体テスト、結合テスト
  - 運用テスト、回帰テスト、受入テスト

Software Life Cycle

- ソフトウェアライフサイクル
  - ソフトウェア開発において、ソフトウェアの企画立ち上げから廃棄に至る一連の流れ。
  - [企画]→[要件定義]→[開発]→[運用]→[保守]
- 企画プロセス
  - システム構築において、システム化しようとする対象業務の問題点を分析し、実現すべき課題や優先順位を定義する。
  - システム構想の立案では、経営要求、課題の確認などが行われ、システム化計画の立案では、開発スケジュールの検討、導入の費用対効果の予測などが行われる。

Business Requirements

- 要件定義プロセス
  - システム構築において、システムに関係する利害関係者のニーズや要望、制約事項を定義するプロセス。
  - システムに対する制約条件や業務要件について、関係者の合意を得る。
- 業務要件
  - 要件定義プロセスの初期の段階において定義される内容であり、システム構築やソフトウェア開発において、システム化の対象となる業務の流れを整理したもの。

Procurement

- 開発プロセス
  - システム構築において、システムの応答時間や処理時間の評価基準を設定するプロセス。
  - システム方式の設計と評価を行い、必要な要件に照らして最適化されたソフトウェアの要件が決定される。
- 調達
  - システム構築に必要なものを必要に応じて取りそろえること。
  - 基本的な流れは、[情報提供依頼]→[提案依頼書の作成と配付]→[選定基準の作成]→[ベンダ企業からの提案書および見積書の入手]→[提案内容の比較評価]→[調達先の選定]→[契約締結]→[受入れ・検収]である。

Acceptance Inspection

- 提案書
  - ベンダ企業が、RFP を基にシステム構成、開発手法などを検討して作成し、依頼元に対して提案する文書。
- 見積書
  - システムの開発、運用、保守などにかかる費用を示す文書。
  - 取引先の選定や発注内容の確認にとって重要である。
- 検収
  - ベンダからの納品物が要求した仕様どおりであるかの確認を行うこと。

Functional Requirement

- 機能要件
  - 必ず満たさなければならない機能に関する条件。
  - 例えば、データ構造、種類、出力の方法や形式など。
- 非機能要件
  - システム開発で定義される機能要件以外の要件。
  - 例えば、可用性、性能・拡張性、
    運用・保守性、
    移行性、セキュリティ、
    環境・エコロジーなど。

Quality Characteristic

- 品質特性
  - ソフトウェアの品質を評価する基準。
  - ISO/IECの規格ではソフトウェアの品質特性モデルとして、下記の6つの特性を挙げている。
    - 機能性(Functionality)：必要な機能の実装程度。
    - 信頼性(Reliability)：どの程度機能するかに影響する特性。
    - 使用性（Usability）：利用するのにかかる手間に影響する特性。
    - 効率性(Efficiency)：性能や要するリソース量の影響度。
    - 保守性(Maintainability)：変更を加えるときの労力の特性。
    - 移植性（Portability）：別の環境への移植のしやすさ。

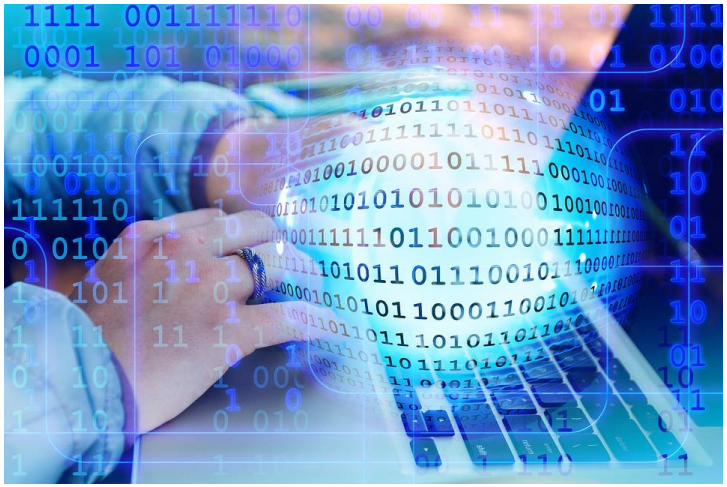
ソフトウェア開発の見積り手法

- プログラムステップ法
  - プログラムを記述するソースコードの行数を基に、ソフトウェアの開発規模を算出する手法。
- ファンクションポイント法
  - 外部入出力や内部ファイルの数と難易度の高さから論理的にファンクションポイントを設けて、開発規模を算出する手法。
- 類推見積法
  - 過去に経験した類似の開発における経験やデータを基にして、開発規模を算出する手法。

Software Design

- 外部設計（基本設計、概要設計）
  - 要件定義に基づいて、ユーザーに提供する機能、ユーザインターフェース、レイアウト、フォーマットなどを決定する工程。

- 内部設計（Internal Design）
  - 外部設計で決定された要件を、どのような構成で、どのように動作させて実現するかを決定する工程。



Programming

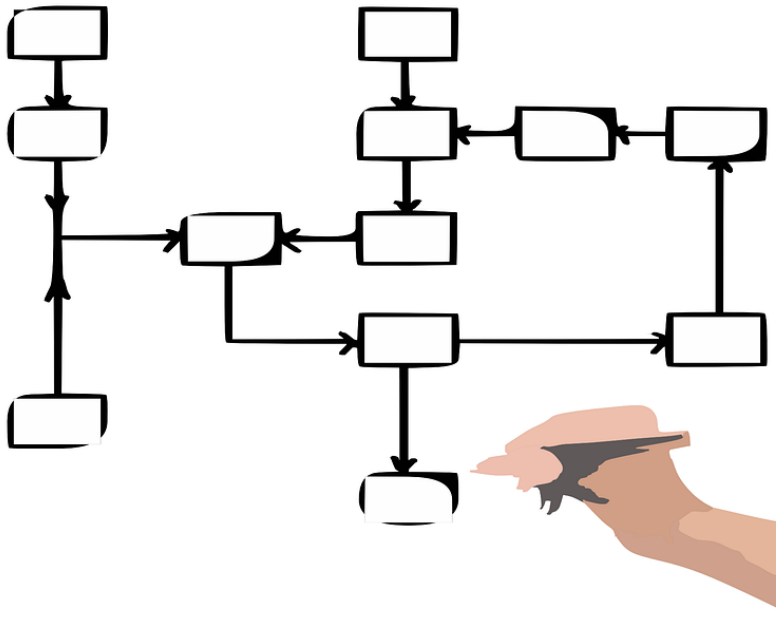
- プログラミング
  - プログラム言語によって、目的とする処理のための
    ソースコードを作成(コーディング)する工程。
  - 動作のテストやデバッグなどの作業を含む場合もある。
- コーディング(Coding)
  - プログラム言語、ハードウェア記述言語、マークアップ言語
    などで、ソースコードを作成する工程。
- デバッグ(Debug)
  - プログラムの欠陥を取り除く修正作業。プログラムの欠陥の
    ことを、虫を意味するバグと呼ぶとことに由来している。

Branch Coverage

- 分岐網羅
  - プログラムの全ての分岐経路を少なくとも1回実行するようにテストケースを作成するテスト方法。
- 同値分割（Equivalence Partitioning）
  - 出力結果が同じであると考えられるデータを入力するテスト省力化の方法。
- 限界値分析（境界値分析、Boundary Value Analysis）
  - 「合格の60点、不合格の59点」など、出力結果が変化するデータの境界であると考えられるデータをテストケースとして入力するテスト省力化の方法。

Functional Test

- ブラックボックステスト
  - プログラム仕様書通りに動作することを検証するテスト。
  - システムへの入力と出力結果のみを検証するテストであり、黒箱試験とも呼ばれる。
- ホワイトボックステスト
  - 内部設計での仕様書通りに動作することを検証するテスト。
  - プログラムやモジュールにおける単体テストとして実施され、白箱試験とも呼ばれる。



Integration Testing

- 単体テスト(Unit Test)
  - プログラムの内部設計書に基づいて、プログラムがモジュール単位で正しく動作するかをホワイトボックステストによって確認する工程。
- 結合テスト
  - 単体テストに合格した複数のモジュールを組み合わせて動作させ、モジュール間のインターフェースを確認する工程。
  - ドライバを使って下位のモジュールから結合する「ボトムアッププテスト」、スタブを使って上位のモジュールから結合する「トップダウンテスト」、モジュールをすべて組み合わせる「ビッグバンテスト」などがある。

Regression Test

- 運用テスト（Operations Test）
  - システムの利用者が、実際の業務に沿った環境でテストを行うことで、実際の稼働か可能か否かを確認する工程。
- 回帰テスト（退行テスト、リグレッションテスト）
  - システムやプログラムを変更・修正した場合に、その作業によって、今まで正常に機能していた部分に不具合などの影響が出ていないかを確認する工程。
- 受入れテスト
  - 完成したシステムが、必要な機能や要求した性能を実際に満たしているか否かを、システムの利用者が確認する工程。

お疲れさまでした！