品質と保守

2022/12/5

海谷 治彦

品質特性

- 品質は単にバグが無いというだけでなく，多様な視点から議論されている.
- 現在は，ISO25000シリーズ(SQuaRE)での定義が主流であるが，いままで，そして次頁で紹介するISO9126 等と大きな変化はない.

利用時の品質 (ISO9126の場合)

- 有効性
  - 利用者が明示された目標を達成するときの正確さと完全さの度合い
- 効率性
  - 利用者が目標を達成するために正確さと完全さに関して使用した資源の度合い
- 満足性
  - 利用者の満足度
- リスク回避性
  - 製品やシステムが利用環境に存在するリスクを緩和できる度合い.
- 利用状況網羅性
  - 想定および想定外の状況で, システムが上記の度合いを満たして利用できる割合.

製品の品質 (ISO9126の場合)

- 機能適合性: 要求にあっているかどうか？
- 性能効率性: 時間，資源，容量等の視点から.
- 互換性: 共存や相互運用
- 使用性: いわゆるユーザビリティ
- 信頼性: 障害許容や回復性能も含まれる.
- セキュリティ
- 保守性: 再利用，解析，酒精，試験性など
- 移植性

品質の測定

- 後述のソフトウェアメトリクスを用いることがほとんど.
- 全ての特性ではないが，メトリクスによって測定可能な品質もある.

開発プロセスの品質

- 各種標準において，開発プロセス（開発チームの能力といってもいいかも）の格付けが行われている.
  - 格付けは語弊があるが，実質，格付け.

- CMU(カーネギメロン大)のSEIによるもの
  - CMM および CMMI 開発組織全体
  - TSP と PSP 開発チームや開発者個人
- ISO 9000 シリーズ
- SPICE

保守

- まず，多くのソフトは作って売って終わりにはならない.
- 何年か運用される，その間に，

✓運用の支援が必要な場合がある.

✓運用後に機能追加等の変更が必要な場合がある.

- 上記に対応する活動を保守と呼んでいる.

保守の現状

- ハード（自動車等）ならリコール沙汰になるような修正でも，現在では，ネットワーク経由で，ソフトウェアの修正や機能拡張が行われている.
  - OS, アプリ等
  - 実は，テレビやゲーム機器などの機能でも同様.

- 機能拡張といっても，無限に行われるわけでなく，どこかで，全く新しいソフトといれかえる必要がある.
  - ソフトウェアの構成上の問題.
  - ビジネスの問題，新製品買ってもらわないと商売にならない.

保守の分類

- 修正保守
  - 名前の通り，運用中に仕様通りに動作しない場合に対処する．いわゆるバグフィックス．
- 適応保守
  - 動作環境への適用．例えばOSのアップデートへの追従等．
  - 要求されるパフォーマンスの変化に追従等の要求変更への対応も含まれる．
- 改善保守
  - 一般的な機能拡張や性能改善．
  - 将来の改善に対応しやすくするようにソフトの構造を変更するのも含む．（以下の予防っぽいけど）
- 予防保守
  - 将来，起こりうる障害に対して事前に対処すること．
  - CAPECを見て新規の攻撃にいち早く対応する等．

保守技法

- 保守は基本的に稼働中のソフトの理解を前提とする.
- よって，技法は既存ソフトの理解を助けるものが多い.

1. 構成管理，版管理
   - 本講義の最終回にて.
2. インパクト分析
   - 次回.
3. 回帰テスト (regression test)
   - もとからある機能もちゃんと動くかテストする.
4. 可視化，Visualization
   - プログラム等をグラフ表現で可視化する.

Reengineering, Restructuring

- Reengineering
  - 既存ソフトを構成しなおすこと.
  - 設計の見直し，アーキテクチャの置き換え等，異なる抽象度の見直しも行われる.
- Restructuring
  - コード内，設計内等，同じ抽象度内で，構造の見直しを行うこと.
  - 代表的な手法としては，リファクタリングや，プレファクタリング等.
  - リファクタリング Refactoring
    - クラスやメソッド等の名前の変更
    - 同じ機能をもつメソッド等の統合
    - 同じ処理(コードクローン)を認識して，メソッドとして共通化する.

プログラム理解

- 制御フロー解析
  - ほぼホワイントボックステストに同じ
- データフロー解析
  - データの利用や定義を追跡する.
- プログラムスライシング Program Slicing
  - ある役割をもつ変数に関係する部分だけ，プログラム全体から，部分的に抜き出す技法.
  - 巨大なプログラムでも，幾分，小さくなるので，理解しやすくなる.
  - 入力データを限定して，より小さいスライスを抜き出す手法もあり，無論，そのほうが理解が楽になる. (Dynamic slicing)

DevOps

- 最近，流行りの開発技法.
- 開発(Development)と運用(Operations)を連携して行いましょうという考え方.
- 素人から見れば当たり前に見えるが，従来は，開発現場と運用現場の間で連携が十分にとれてなかった.
- 開発側からの運用での支援だけでなく，運用でのログ等から，次期開発のヒントを得るところが特徴.
- いわゆる，アジャイル手法のカテゴリ.

別紙に続く