試験項目によるプロジェクト工数見積りに関する研究

NTTコムウェア　OSS推進部

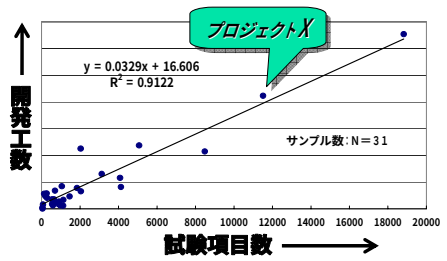
開発技術部門

アジェンダ

0. 提案内容と意義
1. 研究の背景
2. 研究の目標
   - 課題の設定
3. 研究内容
   - 調査対象
   - 調査方法
4. 研究結果とその分析
5. 今後の展望
   - ユースケースの利用
6. まとめ

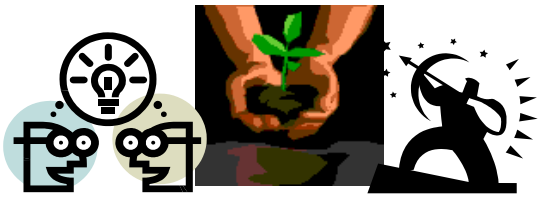
提案内容

- 試験項目と開発工数との間の回帰分析をしてみました．
- 開発工数の見積りに使えるかも知れません．



提案の意義

- 試験項目をプロジェクト工数の見積りに用いる意義
  - データの有効利用
  - 直感に訴える単純なアイデア
  - プロセス改善の追求（テストファーストに導く）



研究の背景

- 規模見積りは，プロジェクトの成功の鍵を握る重要な作業
- 矛盾を含んだ困難な作業
  - 過小に見積ると必要な工期やリソースが十分に準備できず実行時の失敗リスクが増加する．
  - 過大に見積ると，競争相手に負けて契約を得ることができない．
  - 見積り精度を向上するには，詳細な検討が必要となる．
  - 初期段階では検討の前提条件さえ不明確

⇩

- 様々な見積り技法が提案されている

各種見積り技法

- ソースコード量（SLOC）による見積り
  - 計測が簡単で数多くの既存プロジェクトのデータが存在
  - 見積り過程が感覚的
  - ソースコードを書かないタイプの開発には適用できない．
- ファンクションポイント法による見積り
  - 見積り方法がルール化されている
  - 実践したプロジェクトのデータが少ない
- ユースケースポイント法による見積り
  - 開発方法論に組み込まれている
  - ユースケースは規模の概念ではない
  - 実践したプロジェクトのデータが少ない

研究の目標

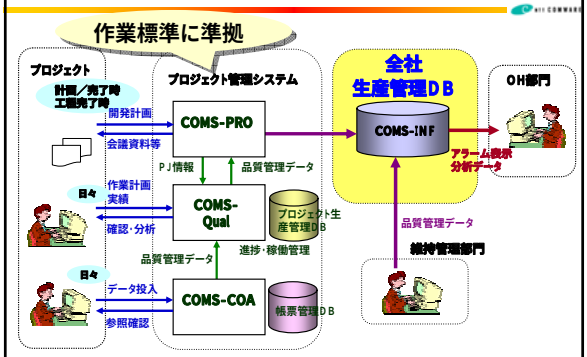
- 既存の技法は問題を含むため，そのまま適用できない．
- 新たな見積り技法の開発が必要．

- できる限り，見積りのために開発プロセスに特別なタスクや特別な成果物を追加しない．
- 既存プロジェクトのデータを用いて独自の見積りモデルを構築する．

試験項目数に注目する理由

- ＳＬＯＣの特徴（廃れない理由）
  - 自動計測が簡単
  - 経験データ数が多い
  - プロジェクト規模との結び付きが直感的に受け入れ易い
- 機能量の課題（拡がりが鈍い理由）
  - 自動計測が困難
  - 実践した経験データ数が少ない
  - 概念的で計測のためのトレーニングが必要
- 見積りから見た試験の特徴
  - **計測の自動化が簡単**
  - **試験は開発にとって欠くことができないタスク**
  - **プロジェクト規模の根拠として受け入れ易い**
- 試験から見積りができるのか調査が必要

調査対象（生産管理ＤＢ）



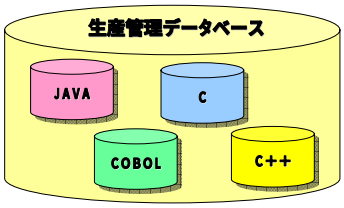
調査対象（試験の種別）

- ＮＴＴコムウェアの作業定義
  - ３段階の試験を定義してデータを残している

工程名	呼　称		概　　要	ＳＬＯＣ ISO-12207
単体試験	ＵＴ	UnitTest	各モジュール単位の機能確認を行う	5．3．7
結合試験	ＩＴ	IntegrationTest	各モジュールを通して、モジュール間インタフェース、及びモジュール間機能確認を行う	5．3．8 5．3．9
総合試験	ＰＴ	ProductTest	システムとしての総合的な機能確認を行う	5．3．10 5．3．11

調査方法

- **成功プロジェクトの残したデータを分析対象**
- **オーバーオールの開発工数を目的変数に，試験項目を説明変数として回帰分析を実施**
- **回帰テストの影響を避けるため新規開発プロジェクトを選択**
- **プロジェクト特性（開発言語）によってデータを層別**



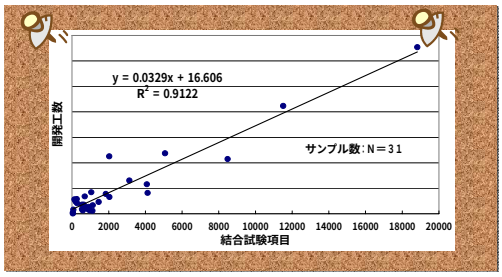
調査結果の分析１

- **決定係数の大きさで各説明変数を個別に評価した**
  - **言語別では試験項目の方がＳＬＯＣより工数との相関が強いものがある．**

開発言語	決定係数（R^2）			データ数
	ＩＴ	ＰＴ	ＳＬＯＣ	
ＶＢ＆ＶＣ＋＋	0.91	0.71	0.79	31
Ｊａｖａ	0.75	0.77	0.68	34
JAVA script	1.00	0.99	0.98	4
Ｃ＋＋	0.73	0.19	0.64	36
Ｃ	0.28	0.2	0.55	139
ＣＯＢＯＬ	0.55	0.50	0.94	17
ＰＲＯ-Ｃ	0.95	0.11	0.99	5
ＬＯＴＵＳ	0.87	0.53	0.87	5
組み合わせ	0.69	0.41	0.52	25
全体	0.41	0.23	0.60	296

調査結果の分析2

- VBとVC＋＋の回帰分析結果
  - ツール利用開発では試験項目の方がSLOCより工数との相関が強い



調査結果の分析3

- 開発言語によって回帰式の傾き（生産性）が異なる
  - ある試験項目数に対し，工数大＝生産性小，工数小＝生産性大
  - 使用言語はシステム特性を代表していると分析

傾き小

傾き大

開発言語	回帰式 傾き	回帰式 切片	データ数
VB＆VC＋＋	0.03	17	31
Java	0.05	35	34
JAVA script	0.02	0	4
C＋＋	0.22	148	36
C	0.06	164	139
COBOL	0.10	158	17
PRO-C	0.07	14	5
LOTUS	0.03	3	5
組み合わせ	0.09	27	25
全体	0.09	92	296

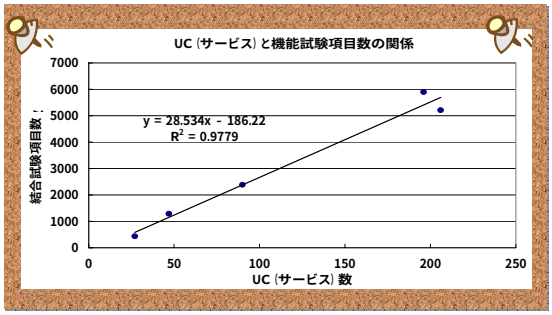
注：生産性＝ 試験項目数 / 開発工数

試験項目の見積り

- 試験から工数を見積る見通しはついたが．．．．
- しかし，試験を見積る方法論がなければ結果を活かせない
- しかも，開発のできるだけ初期段階で正確な試験項目数が欲しい

- 【ユースケースの利用】
  - システムが果たすべき機能をユーザレベルで記述したドキュメントで，システムの外部仕様として要件定義に利用される．
  - RUP®をはじめとするオブジェクト指向開発手法において開発の極初期段階で着手するアクティビティーとして定着している．
  - システムの果たすべき責務が記述されており試験項目の抽出根拠として適当

ユースケースと試験項目の相関分析結果



まとめ

- 試験項目数と開発工数の間に強い相関が認められる
- 直接コードを書かない開発の見積りにも適用可能
- 回帰式はプロジェクト特性毎の生産性の違いを映している

- 【結論】
  - 試験項目数は，開発工数見積りにおけるSLOCの代替手段として有望
- 【課題】
  - ユースケースを利用した試験項目の見積り方法