ユースケースを利用した顧客指向の進捗管理手法

近藤隆史　　東 基衛
早稲田大学大学院　理工学研究科　経営システム工学専攻

1　研究背景

現在のソフトウェア開発における進捗管理は，開発者が把握している指標や経験的予測などに基づいており，顧客が容易に理解できるものではない．従って，顧客は進捗度を提示されてもその実体について理解していないのが現状である．

そんな中，近年プロジェクトマネジメントの知識体系(PMBOK)において，顧客に対する進捗報告の重要性が述べられている[1]．そこで，今後ソフトウェア開発の複雑さを表現し，顧客に進捗度を可視化できる管理手法が必要になると予測される．

2　研究目的および概要

本研究は，顧客の視点に基づいた進捗指標を用いてソフトウェア開発工程の可視化を行い，顧客にソフトウェア開発の複雑さの理解を促し，開発プロジェクトに関するリスクの共有を図ることを目的とする．そして，開発者には進捗状況を説明する必要が高まることにより，開発者の責任感が向上すると考えられる．

そこで本研究は，顧客の機能要求を表すユースケースに着目し，その実現度を用いて一貫して進捗管理を行うことにより，顧客に理解可能な進捗管理手法を提案する．

3　研究内容

3.1 各ユースケースの規模測定

本研究では，ユースケースに基づいて規模測定を行うために，機能要求との関係性が高いユースケースポイント(UCP)法を利用する[2]．UCP 法は，各アクターと各ユースケースについて規模を測定している．そのため，各ユースケースについて進捗を把握するために，求められたユースケースについての規模(UCPuc)に，“アクターが機能を使用する”(ユース)頻度に応じて，各アクターの規模(UCPact)を算入する．これを各機能の規模として開発工程を進める(図 1・図 2)．

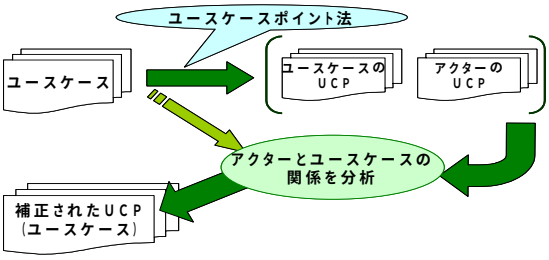

図 1　各ユースケースの規模導出までの流れ

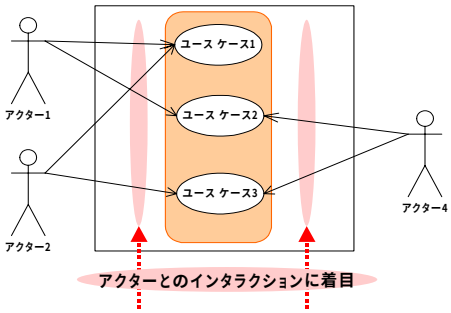

図 2　ユース頻度考慮アプローチ

各ユースケースのユース頻度を考慮した規模(UCPad)を求める手順を以下に述べる．

① 各ユースケースに対するユースの頻度(ACTfre)をアクターごとに算出する．

② 各アクターについて，各ユースケースでユースした頻度の総数($\sum$ACTfre)を求める．

③ 各アクターについて，UCPact をユースした頻度の総数($\sum$ACTfre)で除算することで，ユース 1 回あたりの UCP を求める．

④ 各ユースケースにおいて，各アクターのユース頻度(ACTfre)に応じて，各アクターのユース 1 回あたりの UCP を乗算する．そのユースケースごとの総和を，ユースケース i についてのアクターの複雑度 $UCPcom_i$ とする．よってユースケース i，アクターj について以下の計算式が成り立つ．

$$UCPcom_i = \sum_j \left(ACTfre_{i,j} * UCPact_j \Big/ \sum_j ACTfre_{i,j} \right)$$

⑤ ユースケース i において，$UCPuc_i$ に複雑度 $UCPcom_i$ を加算し，ユースケース i のユース頻度を考慮した規模の未補正値($UUCPad_i$)とする．

$$UUCPad_i = UCPuc_i + UCPcom_i$$

⑥ 開発技術および開発環境に応じて補正を行い，各ユースケース i のユース頻度を考慮した規模 UCPadi を求める．なお，TFactor は技術要因，EFactor は環境要因の係数である．

$$UCPad_i = UUCPad_i \times TFactor \times EFactor$$

A Progress Control Methodology for Client Using Use Case
Takafumi KONDOH & Motoei AZUMA
Dept. of IMSE, Graduate School of Sci. & Eng. Waseda Univ.

3.2 各工程の進捗と規模の関係

ユースケースの実現度を表すためには，各工程とユースケースとの関係を明らかにする必要がある．しかし，要求定義で分析されたドキュメントとオブジェクト指向設計書(クラス図など)とは，構造的な不一致が存在すると言われている[3]．そのため，設計工程以降は顧客の要求を実現するための工程であるにもかかわらず，工程の状態やその成果を要求仕様書から結び付けて判断することは難しかった．各機能要求の実現度を把握するためには，このような構造的な問題を解決する必要がある．

そこで本研究では，進捗度を機能の実現度で表すため，進捗指標として規模を数値化した UCP 単位で表す．

- **要求定義・設計工程**

追加・変更が起きていない場合，どのユースケースについても共通の進捗とする．

- **実装工程**

実装工程は，作業がクラスやモジュールなどを基準として行われるため，機能要求との対応が難しかった．本研究ではユースケースを分析することにより，クラス(またはモジュール)とユースケースの関連度を調べ，関連度に従い各クラスに要する規模を求める．その方法を以下に示す．

(1) 各ユースケースにおける，各クラス(モジュール)の呼び出し頻度(CLASSfre)を算出する．

(例 1：…が C に～する　例 2：C は，～する)

(2) 各ユースケースにおける，各クラスの呼び出し頻度に応じて，実装工程に要する規模 UCP(Coding)，を割り振り，ユースケース i，クラス j における規模($UCPcl_{ij}$)を求める．

$$UCPcl_{ij} = CLASSfre_j * UCP(Coding) \Big/ \sum_j CLASSfre_{ij}$$

(3) ユースケース i のクラス j における規模($UCPcl_{ij}$)を，クラス別に総和し，クラス j の規模 $UCPcl_j$ を求める．

$$UCPcl_j = \sum_i UCPcl_{ij}$$

- **テスト工程**

テスト工程では，ユースケースで記述されたパス(例外パスを含む)が正常に動作するか検証するため，ユースケースごとに進捗を把握する．

3.3 機能別の進捗度測定

機能別の進捗度は，各工程におけるユースケースの実現度の総和に基づいて求める．ユースケース i の実現度は，以下の方法で算出する．

(1) 要求定義･設計工程は，全体の進捗と共通とする．

(2) ユースケース i に関する実装工程の進捗度 P_i(coding)を求める．P_i(Coding)を求めるために，クラス j の規模 $UCPcl_j$ に占めるユースケース i クラス j に関する規模($UCPcl_{ij}$)の割合を求め,それをユースケース i とクラス j の関連度とする．次に，ユースケース i とクラス j の関連度にクラス j の実装工程における進捗度 P_j(coding)を乗算し，ユースケース i クラス j に関する進捗度とする．そして，それをクラス j について総和することで，ユースケース i に関する実装工程の進捗度 P_i(coding)が求められる．

$$P_i(coding) = \sum_j \frac{UCPcl_{ij} * P_j(coding)}{UCPcl_j}$$

(3) テスト工程は，その機能（ユースケース）に関するテストを抽出する．そして，その作業量の総和（実績値）が，ユースケース i におけるテスト工程の進捗度とする．

(4) (1)～(3) の総和が，ユースケース i に関する実現度，つまり機能の実現度となる．

4 検証

実装工程における 3 種類の対面型システムの各機能モジュールに関する規模を，提案進捗指標とファンクションポイント(FP)それぞれで計測し，両者の相関分析を行った．表 1 の相関係数により，現在規模を計測する指標として広く使われている FP との相関係数が高いことがいえる．

このことから，測定技法と求められる規模の関係を考慮に入れる必要はあるが，提案指標もまた適切に規模計測が行えるといえる．

また，提案手法および検証結果によりユースケース i とクラス j の関連度が定まり，工程をユースケースの進捗度として表すことができる．

表 1　相関分析結果

測定したシステム	各機能モジュールの相関係数
Webシステム	0.729
CSSシステム	0.752
ホストシステム	0.874

5 結論

本研究は，顧客に理解可能な機能実現度に基づく進捗管理手法を提案した．これにより，ソフトウェアプロジェクトを顧客に理解可能な形式で伝えることができ，ソフトウェア開発プロジェクトに関するリスクを共有することができる．しかし，提案手法では品質要求の実現度は表すことはできないため，品質要求の実現度を可視化することが今後の課題となる．

[1] PMI，プロジェクトマネジメント知識体系ガイド，Project Management Institute,2003 年 4 月

[2] Gustav Karner, “Use Case Points Resource Estimation for Objectory Projects”, Objective Systems SF AB Rational Software, 1993

[3] C.R.Turner, A Fuggetta, L.Lavazza, and A.L.Wolf, “A conceptual basis for feature engineering,” in Journal of Systems and Software, 49, 1999, pp3-15.