論　文

ユースケースポイント計測におけるアクタとユースケースの自動分類の試みと支援ツールの試作

津田　道夫[†,††]　楠本　真二[†]　松川　文一[†]　山村　知弘[†]
井上　克郎[†]　英　繁雄[††]　前川　祐介[††]

Automatic Type Classification of Actors and Use Cases in Use Case Points Measurement and Its Supporting Tool

Michio TSUDA[†,††], Shinji KUSUMOTO[†], Fumikazu MATSUKAWA[†], Tomohiro YAMAMURA[†], Katsuro INOUE[†], Shigeo HANABUSA[††], and Yusuke MAEGAWA[††]

あらまし　ソフトウェアプロジェクトの計画段階では，通常，ソフトウェアの開発規模を見積もり，その値に基づいて開発工数や開発期間を予測する手法がとられている．開発規模見積りにはファンクションポイント（FP）法がよく用いられているが，ある程度詳細な要求分析が終わらなければ正確な FP を計測することは難しい．そこで，開発初期に作成されるユースケースモデルに基づいて工数見積りを行うユースケースポイント（UCP）法が提案されている．しかし，UCP 法を実際に適用する場合，計測者の主観に依存する部分があるため，同一ユースケースモデルに対する計測でも，計測者によって誤差が生じるという問題点も存在する．本論文では，このような計測者による誤差をなくすことを目的として株式会社日立システムアンドサービス（日立 SAS）で試作したユースケースポイント計測ツールについて述べる．具体的には，いくつかの制限を設けた上で，UCP 計測で行われるアクタとユースケースの自動分類手法を提案し，その手法に基づく UCP 計測ツール U-EST を試作した．更に，U-EST を日立 SAS の五つのソフトウェアプロジェクトで作成されたユースケースモデルへ適用し，その適用可能性を確認した．

キーワード　見積り，ユースケースポイント，ツール，プロジェクト管理

1. まえがき

ソフトウェアプロジェクトの計画段階では，通常，ソフトウェアの開発規模を見積り，その値に基づいて開発工数や開発期間を予測する手法がとられている．近年では，開発規模の見積り方法として，ソフトウェアの機能要件から規模を定量的に計測するファンクションポイント法（FP 法）[1] が普及しつつある．しかし，FP 法を用いて正確に機能量を計測するには，ソフトウェアで扱うデータ項目数等が必要であり，要求分析を詳細に行う必要がある [6]．近年の受注競争の激化や，開発期間の短期化に伴い，開発プロセスの更に早期の段階（上流工程）で見積りをする手法が求められている．

そのための一つの手法として，ユースケースポイント（UCP）法が提案されている [7]．UCP 法は，FP 法をベースとし，近年増えつつあるオブジェクト指向開発において，早期の要求分析の段階で作成されるユースケースモデルをもとに，開発規模，及び，開発工数を見積るための手法である．UCP 法の有用性を評価した結果も報告されている [2], [11]．

一方，UCP 法では，ユースケースモデルに記述される複雑さを考慮するために，アクタ及びユースケースに対して重み付けを行う．特に，ユースケースの重み付けではユースケース中で定義されているトランザクション（原始的な一群のアクティビティ）数を数え上げる必要があり，トランザクションのとらえ方によっ

† 大阪大学大学院情報科学研究科，豊中市
Graduate School of Information and Science Technology, Osaka University, Toyonaka-shi, 560–8531 Japan
†† （株）日立システムアンドサービス，東京都
Hitachi Systems & Services, Ltd., Ota-ku, Tokyo, 143–8545 Japan

て，同一プロダクトに対する計測でも，結果に誤差が生じる可能性がある．より正確な見積りを実施するためには，組織的な見積り教育や訓練，実績データの蓄積が重要になる [5]．この問題に対する一つのアプローチとして，UCP 計測の自動化が考えられる．しかし，筆者らの知る限り，UCP 計測の自動化に関する研究は行われていない．

株式会社日立システムアンドサービス（以降，日立 SAS）では，ソフトウェア開発プロジェクトにおいて FP 法の導入を行っているが，開発期間の短期化に伴い UCP 法を用いた見積りの併用を検討している．UCP 法の開発現場への導入においては，計測のための開発者の負担増を避けること，計測者による誤差をなくすこと，データの統一的な蓄積が重要課題であり，UCP の自動計測ツールの開発が必須であった．

本論文では，見積り経験の浅い者でも見積りが可能となること，及び，計測者による誤差をなくすことを目的として設計・試作した，ユースケースモデルを入力とし，UCP を自動計測するツールについて述べる．具体的には，UCP の自動計測のために，アクタとユースケースの重み付け方法を提案し，そのツール実装について述べる．更に，ツールの有用性を確認するため，日立 SAS で行われた五つのプロジェクトで作成されたユースケースモデル用いて，ツールによる重み付けと経験者による手動での重み付けの結果との比較結果について述べる．この比較から，ツールが有用であるとの結論を得ている．

以降，**2.** で UCP 法について説明する．次に，**3.** でツール化へ向けて提案したアクタとユースケースの自動分類手法について述べる．**4.** で，実装した見積り支援ツール U-EST に関して紹介する．**5.** で，いくつかのプロジェクトのユースケースモデルを対象にして行ったツールの適用実験の結果の分析と考察を行う．最後に **6.** で，まとめと今後の課題について述べる．

2. ユースケースポイント法

2.1 ユースケースモデル

ユースケースモデルは，ユースケース図とユースケース記述の二つの要素で構成されている．

ユースケース図：ユースケースの概念を UML（Unified Modeling Language）で視覚化して表現したものである．アクタを人間の形のアイコンで示し，ユースケースを楕円形のアイコンで表現する．アクタとユースケースとの関連は実線で表す．

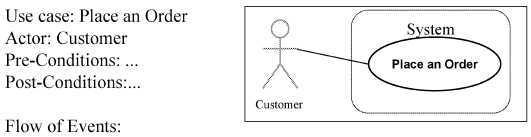


図 1 「注文を出す」ユースケース
Fig. 1 Example of a use case: “Place an order.”

ユースケース記述：ユースケース図に記述されたユースケースそれぞれについて，そのユースケースの機能を実現するために必要な詳細情報を記述したものである．

例として,「注文を出す」というユースケースに関し，ユースケース図とユースケース記述をまとめたものを図 1 に示す．

UCP 計測においては，ユースケース記述の中で，そのユースケースにおけるアクタとシステムとのやり取りを記述した「イベントフロー」に着目する．

2.2 ユースケースポイント法

UCP 法は，オブジェクト指向開発の早期段階における工数及びコスト見積手法が要求される中で，Karner によって提案された工数見積手法である [7]．この手法は，FP 法をベースとし，ユースケースモデルに基づいて見積りを行う．具体的には，ユースケースモデルに記述されるアクタ，ユースケースを重み付けしてその数を数え上げ，計測対象プロジェクトの技術的な要因や，開発環境の要因をもとに調整を行い，工数を見積る．

2.2.1 計 測 手 順

UCP 法での計測は，次の五つのステップで実施される．

Step 1: アクタ，ユースケースの重み付け
Step 2: 未調整ユースケースポイントの算出
Step 3: 技術要因，環境要因による調整
Step 4: ユースケースポイントの算出
Step 5: 工数の見積り

直観的には，アクタとユースケースを重み付けして足し合わせたものが未調整ユースケースポイントであり，それを開発における技術要因や環境要因で調整（補正）したものがユースケースポイントとなる．本研究においては，特に Step 1，2 の部分に着目してい

表 1　アクタのタイプと重み係数
Table 1　Types and weights for actors.

タイプ	説明	重み係数
単純	定義済み API を備えた別システム	1
平均的	プロトコル駆動のインタフェース（別システム），テキストベースのインタフェース（人間）	2
複雑	GUI を介する人間	3

表 2　ユースケースのタイプと重み係数
Table 2　Types and weights for use cases.

タイプ	説明	重み係数
単純	トランザクション数が 3 個以下	5
平均的	トランザクション数が 4 個から 7 個	10
複雑	トランザクション数が 8 個以上	15

るため，ここでは Step 1，2 のみについて説明を行う．Step 3 以降については，文献 [7], [10] を参照して頂きたい．

Step 1 では，まず，ユースケースモデルに記述されているアクタについて，表 1 に基づいてタイプの分類を行う．各タイプには重みが設定されており，各タイプのアクタの個数に重み係数を掛け，合計したものが，アクタの重み（AW: Actor Weight）となる．ここで，単純なアクタというのは，定義済みの API（Application Programming Interface）を備えた別システムを指す．平均的なアクタは，TCP/IP などのプロトコルを介して計測対象システムと相互作用する別システム，若しくはテキストベースのインタフェース（ASCII 端末といったもの）を介して相互作用する人間を指す．複雑なアクタは，GUI を介してシステムと相互作用する人間を指す．

次に，ユースケースについても表 2 に基づいてタイプの分類を行う．ユースケースの場合は，イベントフロー内のトランザクションの数に基づいてその複雑さを決定する．トランザクションは，原始的な一群のアクティビティと定義され，これはすべて完全に実行されるかあるいは全く実行されないかのどちらか一方となる．このトランザクションの定義は，ファンクションポイント法におけるトランザクションファンクションの定義（ソフトウェアに対するデータの出入りを伴う処理，例えば，「画面からのデータ登録」，「帳票出力」，「問合せ応答」）をもとにしている．また，include するユースケースや拡張ユースケースについては，計測の対象とされていない．ユースケースについても，各タイプの個数に重み係数を掛け，すべて合計する．これをユースケースの重み（UW: Use Case Weight）と呼ぶ．

Step 2（未調整ユースケースポイントの算出）では，算出されたアクタの重みと，ユースケースの重みを合計して，未調整ユースケースポイント（UUCP: Unadjusted Use Case Point）を得る．

$$UUCP = AW + UW \tag{1}$$

3. UCP 自動計測の方法

3.1 概　　要

ユースケースポイントの自動計測を行うツールを作成するにあたって，次の 2 点が考慮すべき点となった．

- アクタの分類に必要な情報の抽出
- ユースケースのイベントフローのトランザクションの抽出

そこで，なるべく日立 SAS の環境に特化しないような制限をアクタとイベントフローの記述におき，その上で分類を行うこととした．以降，それぞれの計測方針について述べる．

3.2 アクタの分類方法

Karner の定義では，アクタの分類の基準は，そのアクタが，システムに対してどのようなインタフェースで相互作用するかということである．しかし，モデルに記述されるアクタのもつ情報は名前のみであり，インタフェースに関する情報を明示的にもっているわけではない．そこで，我々はアクタの名前とアクタが関連するユースケースのイベントフロー記述に注目し，以下の Step A1 と Step A2 で分類を行う．

3.2.1　StepA1: アクタ名による分類

表 1 より，アクタが人間であるか，若しくは外部のシステムであるかで，タイプの候補を二つに絞ることができる．すなわち，外部システムであれば単純，若しくは平均的であり，人間であれば平均的，若しくは複雑となる．そこで，第 1 段階として，まず人間か外部システムであるかを判断する．

具体的には，外部システムであると判断できるキーワードを設定し，そのキーワードを名前の語尾にもつアクタを外部システム，そうでなければ人間とする．キーワードの例としては，「システム」「サーバ」などが挙げられる．また，見積りを行う組織，団体等で，ある特定の命名規則を設定する可能性も考慮し，このキーワードはユーザによる変更・追加も可能とする．外部システムに関するキーワードとしたのは，人間である場合の名前には無数のパターンが考えられ，外部

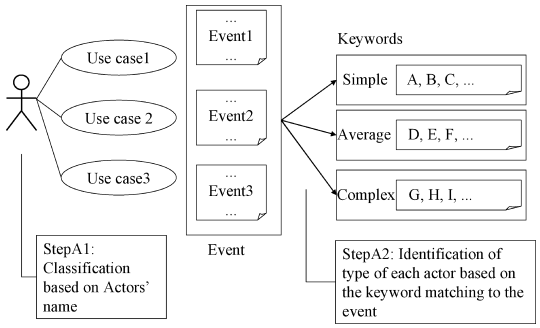


図 2　アクタのタイプ決定プロセス概要
Fig. 2　A process of determining the actor type.

システムに関するそれよりも数が膨大であると判断したためである．

3. 2. 2　Step A2: キーワードによる分類

アクタ名による第 1 段階の分類後，絞られた二つの候補から一つに決定する第 2 段階の分類を行う．ここからは，どのようなインタフェースをもつかに依存するが，先に述べたようにアクタ自身はその情報を陽にはもたない．そこで，対象アクタが相互作用するユースケースのイベントフローに着目し，インタフェースに関する情報をそこから得ることとした．図 2 にアクタのタイプ決定プロセスの概要を示す．

具体的には，まず，インタフェース情報に関するキーワード群を，それぞれのタイプに設定する．キーワードとしては，アクタがシステムに対して行う処理に関するものがある（「入力」,「選択」等）．次に，アクタが関連するユースケースのイベントから，対象アクタが主体となるイベントを取り出し，そのイベント中の単語の中からキーワードと合致するものを探し，当該イベントを合致したキーワードが属するタイプと同じタイプと判定する．しかし，イベント中には，特定のタイプのキーワードだけでなく，異なるタイプに属するキーワードが含まれる可能性がある．このときには，イベント中で最も多く合致したキーワードを含むタイプをイベントのタイプとする．また，このキーワードについてもユーザが変更・追加可能とする．

3. 2. 3　汎化に対する処理

アクタ間には汎化関係が存在するが，Karner はこの汎化関係については特に言及してはいない．我々はこの関係が存在する場合，親アクタの機能を継承した子アクタについては，自分が関連するユースケースを調べるだけではなく，親アクタの関連するユースケースも調べてキーワードの照合をとることにした．また，もし親アクタに関連が全く存在していない場合は，機能を継承した子アクタが既に存在するため，カウント対象とは考えない．

3. 3　ユースケースの分類方法

Karner の定義から，ユースケースの複雑さは，その内部の処理（トランザクション）の数によって決まる．そこで，ユースケース記述内のイベントフローの情報からトランザクションを抽出し，分類を行う．トランザクション数をカウントする最も単純な方法としては，イベントフロー内のイベントの数をカウントすることが考えられる．しかし，ユースケース記述は定まったフォーマットが存在せず，作成者が自由に記述できる．したがって，例えばトランザクションが二つ認識できるような処理が，一つのイベントとして記述される可能性もある．

前述のように，トランザクションは原始的な一群のアクティビティと定義されている．言い換えるとトランザクションは，ユーザあるいはシステムにとって一つの意味のある処理である．Jacobson は，ユーザとシステム間の相互作用について代表的な四つの処理として以下を挙げている [4].

- 主アクタがシステムに要求とデータを送る．
- システムが要求とデータを確認する．
- システムがその内部状態を変更する．
- システムがアクタに結果を返す．

このような一連の処理内容を一つのトランザクションとして認識することは有効であると考えられる．その場合，イベントを分析して「主アクタとシステム間でデータをやり取りしている可能性があるもの」を抽出することがトランザクション抽出の基本となり,「要求とデータを送る」処理を記述しているイベントがトランザクションの開始となる．

トランザクションの開始部分を識別するための方法として，イベント記述の自然語文章から主語や述語を認識し，アクタやシステムが主語であり，かつ，上述した処理が述語として現れているものを抽出する方法をとる．

そこで，以下の StepU1〜U3 でトランザクションを抽出する方法を用いた．

Step U1: ユースケースシナリオ内のイベントに対して形態素・係り受け解析を行い，主語と動詞のセットをトランザクションを構成するイベントの候補とする．

Step U2: 抽出された候補から，主アクタとシステム間でデータをやり取りしている可能性があるものを

取り出す.

Step U3: アクタの動作から，システムの応答までの一連の動作をまとめて，一つのトランザクションとする.

以降，各 Step について説明する.

3. 3. 1 Step U1: 形態素・係り受け解析によるイベント候補の抽出

イベント記述に対する形態素解析及び係り受け解析を行うために，日本語係り受け解析器「南瓜」[12] を利用する.「南瓜」は，SVM（Support Vector Machines）に基づく日本語係り受け解析 [8] 器であり，統計的な日本語係り受け解析器として最も精度が高い (89.29%) とされているツールである.

南瓜の出力から，イベント中の名詞や動詞など，文の成分の情報を得ることができる．ただし，主語と目的語については名詞というカテゴリーで分類されているため，更に助詞などをチェックして主語や目的語であるか判定を行う必要がある．最終的に，主語がアクタとシステムのものをトランザクションに含まれるイベント候補とする.

3. 3. 2 Step U2: イベントに対する動詞の分類

StepU1 で抽出されたイベントに対して動詞の分類を行う．主語と述語となる動詞の対応がとれていても，必ずトランザクションになるとは限らない．トランザクションは主アクタとシステム間のデータのやり取りから認識するため，主語がどのような処理を行っているか，動詞を詳しく調べてみる必要がある．動詞の特徴を調べるために，以下の五つのカテゴリーを設けた.

(1) アクタ入力 A
(2) アクタ入力 B
(3) システム出力 A
(4) システム出力 B
(5) その他

「アクタ入力」とは，主アクタからシステムに対するデータの入力要求の可能性がある動詞であり,「入力（する）」や「検索（する）」などの動詞が挙げられる.「システム出力」とは，システムから主アクタに対する応答の可能性がある動詞であり,「出力（する）」や「表示（する）」などの動詞が挙げられる．主語とは独立して，抽出された動詞がそれぞれどのカテゴリーに当てはまるかチェックする.「アクタ入力」と「システム出力」に大別されるが，それぞれの項目について，処理の違いによって A と B に更に詳細に分類している．アクタ入力，システム出力のどちらにも当てはまらないものはその他となる.

主アクタの入力操作を表す動詞は大別して二つのパターンがある．操作の段階を表す動詞か，一語で応答までを確定できる動詞かである．前者を A，後者を B として定義する．例えば「検索する」という操作は,「（検索キーを）入力する」（操作の段階を表す）と「（システムに）送信する」（一語で応答までを確定できる）という 2 行のイベントで表現される可能性もある.「削除する」という操作は,「（削除すべき項目を）選択する」と「（削除キー等を）押す」というイベントで表現される可能性もある．このように，システムの応答部分が記述されずに，アクタの動作が 2 行続けて記述された場合にこれらの区別をするために A と B の分類をする.

システム操作を表す動詞にも同様に二つのパターンがある．主アクタへの応答の動詞と更に別のシステムへの入力を想定させる動詞である．前者を A，後者を B と定義する.「表示（する）」などは，主アクタからのデータによって確認を行い，処理の結果を返す応答を表す動詞であると考えられる．他方,「認証（する）」などは，対象システムを経由して，更にデータベースやパスワード認証システム等の外部システム（アクタ）へと要求を渡すことが想定される動詞である．これは応答とは別のトランザクションとしてカウントしなければならない.

3. 3. 3 Step U3: トランザクションの判定

Step U2 で得られた主語と動詞のセットの情報からトランザクションの判定を行う．基本原則は，主アクタからシステムに対して要求とデータの送信を行うイベントをカウントし，それ以外の処理を，それに付随したシステムからの応答として纏める.

具体的には，主アクタからシステム方向のデータの送信が行われたイベントを検出し，システムからの応答までを一つのトランザクションとして認識する．ただし，人間が書く文章のすべてがこれに当てはまるとは限らない．主アクタからのデータ送信のイベントのみが書かれる可能性もある．このとき，主アクタの動作が，本当に一つのトランザクションを表しているかを，確認しなければならない．前述したとおり，主アクタからシステムにデータ送信を行う方向の動詞には二種類ある.「入力」という動詞がシステムに対するデータ要求までを意味している場合も,「入力」と「送信」で一つのデータ要求を表している場合もある．これらを区別するために，対象としている前の行の動詞

を調べ，一つのデータ要求として独立させるか，複合させるかを決定する．具体的には，対象としているイベントがトランザクションの開始の可能性がある場合（アクタ A あるいはアクタ B を含んでいる場合），その前の行の動詞がアクタ入力 B あるいはその他であれば，対象としている行はトランザクションの始まりと判断し，アクタ入力 A であればまだユーザからの入力処理の途中であると判断する．

また，システムから外部システムへのデータ要求を行う場合もトランザクションと数える必要がある．この場合は，対象となるイベントがの主語がシステムであり，かつ，動詞がシステム出力 B の場合，そのイベントを一つのトランザクションであると判定する．

3.3.4 包含や拡張に対する処理

Karner の提案した手法では，包含や拡張ユースケースについては考慮しないとされている [7] が，このようなユースケースに，システムの本質的な処理が含まれる場合もあることが指摘されており [2], [9], 機能規模計測において無視できないと考えた．

包含や拡張の関係が存在するとき，それを一つのユースケースとして処理を記述し，その処理を含むユースケースは，自身のイベントフロー内で包含（拡張）ユースケースを参照するイベントとして記述を行う．このことから，包含または拡張するユースケースは，一つのユースケースとして複雑さを決定し，計測対象とした．また，それらを参照するユースケースは，イベントにおけるその参照処理をトランザクションに含めないようにした．

3.4 アクタ・ユースケース記述の制限

これまでに述べてきたように，提案する計測手法ではユースケースモデルに対していくつかの制限を設けることになる．具体的には，以下の (1)〜(2) にまとめられる．

(1) ユースケース記述のイベントの文章構造は単純にする．

(2) 利用する組織のコンテクストに合わせて，アクタ名やトランザクション分類のための動詞キーワードを設定する．

(1) は，イベント記述では，主語，修飾語，目的語，述語（少なくとも主語と述語だけでも）をきちんと記述するということである．これは形態素解析，係り受け解析を行う上で必要な制限である．(2) の制限は，アクタ分類，ユースケース分類におけるアクタ入力 A と B，システム出力 A と B に対応するキーワード，動詞の設定を決めることである．

いずれの制限についても，ある程度のユースケースモデル開発経験のある組織ではそれほど負担の大きい制限ではないと考えている．実際に，**5.** で述べる評価実験では，特に問題とはならなかった．

ただし，**3.2**，**3.3** で述べたアクタとユースケースの分類に関して，分類不可能な場合も現実には考えられる．その場合には，基本的には過去に計測されたプロジェクトの様々な情報を用いた支援が考えられる．以降，それぞれの分類について方針を述べる．

3.4.1 アクタ分類不可能時の処理

3.2 で述べた方法によりアクタのタイプを決定するが，関連するユースケースのイベントフローに，用意したキーワードに全く一致しない場合も考えられる．このような場合，ユーザを支援する手法として，過去の情報を利用することを考える．

具体的には，過去に計測されたプロジェクトの様々な情報をデータベースとして蓄積する．その中のアクタの情報から，分類が不可能なアクタに類似したアクタを検索し，見つかった情報をユーザに提示する．検索はアクタ名の部分一致による検索を行い，アクタ名，そしてそのアクタに対して過去に決定されたタイプ等の情報を表示する．ユーザはそれを受けて，タイプを手動で設定，修正する．

もし，過去の情報からも類似したアクタが見つからなかった場合は，デフォルトのタイプとして，外部システムであれば「単純」，人間であれば「複雑」とする（平均的とは判断しない）．その理由は，前者に対しては，最近のソフトウェア開発の現状を考えると，開発者がすべての処理を記述するより，開発環境のサポートによって，効果的に定義済み API を使用して開発を行うことが多いと考えられるからである．また，後者に対しては，ユーザとシステム間のインタフェースも GUI を介する場合が多いためである．

3.4.2 ユースケース分類不可能時の処理

Karner の定義に基づいてユースケースの複雑さを決定するには，イベントフローが記述されていることが前提となる．しかしながら，実際の開発では，ユースケース図が記述されていても，そのイベントフローまでは記述されないこともある．このような場合，アクタの場合と同様に，過去の蓄積された情報を利用することを考える．蓄積されたユースケースの情報から，対象となるユースケースと類似したユースケース（例えば，同じ名前をもつ）を検索し，見つかった情報を

ユーザに提示する．ユーザはそれを受けて，複雑さの設定，修正ができる．

もし，過去の情報からも類似したユースケースが見つからなかった場合は，複雑さを「平均的」とする．

4. 工数見積り支援ツール U-EST

4.1 ツール概略

3. で述べた方法に基づいて工数見積り支援ツール U-EST（Usecase-based Estimation Supporting Tool）を試作した．U-EST は XMI（XML Metadata Interchange）形式で記述されたユースケースモデルを入力とし，**3.** で述べた方針に基づいて，ユースケースポイントの計測，及び工数の見積りを行う．開発言語は Java であり，規模は約 5KLOC である．なお，今回は UML モデリングツールとして「Describe」[13] を利用した．

4.2 システム構成

本ツールのシステム構成図を図 3 に示す．システムは四つのサブシステムから構成されており，以下に説明を記す．すべての処理はユーザが GUI を通して操作する．

4.2.1 XMI 解析部（XMI Analyzer）

XMI 形式で記述されたモデルファイルを解析し，記述されたアクタ，ユースケース，イベント等，計測に必要な情報を抽出する．

4.2.2 複雑度測定，UUCP 計測部（Scenario Analyzer）

XMI 解析部において抽出した情報をもとに，**3.** で説明した方針に従い，各アクタ，ユースケースのタイプ分類を行う．またその結果から，未調整ユースケースポイントを算出する．各アクタ，ユースケースに対する分類結果，及び未調整ユースケースポイントは，GUI を通してユーザに提示する．

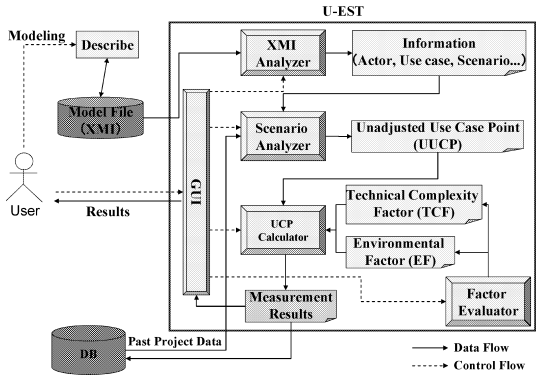


図 3 システム構成
Fig. 3 Overview of U-EST.

また，表示される結果に対し，ユーザは手動にて分類を再設定することも可能であり，変更による各計測値への更新は随時自動で行われる．

4.2.3 調整要因評価部（Factor Evaluator）

技術要因，環境要因それぞれに対して，ユーザがダイアログ上で評価を行い，評価値を入力する．入力された結果をもとに，各調整係数を算出し，画面に表示する．

4.2.4 UCP 測定部（UCP Calculator）

複雑度測定部，及び調整要因評価部の結果を受けて，ユースケースポイント値を算出し，画面に表示する．また，その結果見積られる工数（人時で算出）も同時に画面に表示する．ユースケースポイント値から工数値への変換係数は，ユーザによる変更が可能である．

5. ケーススタディ

5.1 概　　要

U-EST が行う分類が妥当かどうか，そしてツールの適用可能性を評価するため，実際のプロジェクトにおいて作成されたユースケースモデルに対して，ツールを適用した．評価に用いたプロジェクトは，小中規模の Web システム開発プロジェクトであり，全部で五つある．表 3 に，その概要について記す．また，表 4 に U-EST 使用上設定したキーワードやアクタシステムの動詞のリストを示す．

これらのキーワード等の設定は，文献 [10] を参考の上，日立 SAS の複数の技術者と相談して行ったが，特に手間はかからなかった．

5.2 複雑さ分類の結果

対象となるプロジェクトのユースケースモデルにおけるアクタ，ユースケースの複雑さに関して，U-EST によって決定された複雑さと，一人の日立 SAS の技術者（UCP 計測熟練者）の手動によって決定された

表 3 プロジェクトデータ
Table 3 Project data.

Project	開発言語	アクタ数	ユースケース数
A	Java	5	15
B	Java	5	14
C	Java, VB.NET	2	20
D	Java	5	28
E	Java	8	13

表 4　キーワードリスト
Table 4　Keyword list.

アクタ名キーワード	システム，サーバ
単純アクタ用	リクエスト，要求，通知，認証
平均的アクタ（システム）用	メッセージ，メール，送信
平均的アクタ（人間）用	コマンド，テキスト，入力
複雑アクタ用	ボタン，押す，選択
アクタ入力 A 用	入力，選択，指定，決，承認，判定
アクタ入力 B 用	押す，押下，確定，送信，要求，検索，変更，修正，追加，問い合，削除，絞り込
システム出力 A 用	出力，表示，閉じる，返す，送信，更新，求める，削除，検討，修正，参照，登録，検索
システム出力 B 用	認証，チェック，記録

表 5　アクタの分類結果
Table 5　Results of actor types.

Project	手動分類			U-EST 分類			複雑度一致率
	単純	平均	複雑	単純	平均	複雑	
A	1	0	4	0	1	4	0.80
B	3	0	2	2	1	2	0.80
C	0	0	2	0	0	2	1.0
D	1	0	4	1	0	4	1.0
E	0	0	8	0	0	8	1.0

表 6　ユースケースの分類結果
Table 6　Results of use case types.

Project	手動分類			U-EST 分類			複雑度一致率
	単純	平均	複雑	単純	平均	複雑	
A	13	2	0	13	2	0	1.0
B	10	4	0	10	4	0	1.0
C	14	6	0	12	8	0	0.90
D	27	1	0	27	1	0	1.0
E	2	8	3	2	8	3	1.0

複雑さとを比較する．これにより，日立 SAS 内における適用可能性を評価する．なお，U-EST を用いた計測は，筆者の 1 名（五つのプロジェクトの内容についてほとんど知らない者）が行った．

アクタ，ユースケースについて，U-EST と手動での分類結果について表 5, 表 6 に示す．

各表は，U-EST による分類と，手動での分類での，各複雑さに分類された要素の数を記している．複雑度一致率という項目は，U-EST で決定された複雑さと手動で決定された複雑さが一致した要素の数の，全体数に対する割合を示したものである．

5.3　分類結果の考察

アクタの複雑度について，一致した割合が 0.8〜1.0 となっており，高い一致率を達成できている．プロジェクト A と B の誤りはアクタ五つ中の一つのずれである．誤りが生じたのはいずれも外部システムのアクタであった．技術者の手動による分類では，ユースケース記述の情報には現れない技術者のもっている情報や経験が利用できる．しかし，当然ながら自動分類では，ユースケース記述の情報とキーワード以外の情報は利用することはできない．キーワードリストを改善することで完全一致させることができる可能性もあるが，他の方法により計測精度を向上する方策の検討が必要である．例えば，プロジェクトでは開発環境，ユーザインタフェースは一貫したものを使うケースがほとんどであるので，ユーザに「定義済み API を使用するか否か」「GUI を介したプログラムであるか」の 2 点について問うだけで，十分な精度が得られる可能性もある．

表 7　総トランザクション数
Table 7　Number of transaction.

プロジェクト	手動での合計	ツールでの合計	合計値の比率
A	85	85	1.0
B	90	90	1.0
C	130	140	1.08
D	145	145	1.0
E	135	145	1.07

ユースケースの複雑度について，高い割合で複雑度が一致している．表 7 に，ユースケース複雑度分類の基になる，トランザクション数を示す．表 7 に示すようにトランザクション数についても，ほぼすべてのユースケースについて手動との差が 1 以内に収まり，熟練した計測者とほぼ同じ結果が得られた．

ユースケース複雑度分類で差の出たプロジェクト C は，トランザクション数の一致した割合が 0.90 と他のプロジェクトと比べて低くなっているが，トランザクション数自体は手動と比べてユースケース 20 個中 17 個が完全一致し，20 個中 2 個が 1 のずれ，20 個中 1 個のみが 2 のずれを出していた．他のプロジェクトについてもトランザクション数の 1 程度のずれはあるものの，通常は 3 段階の分類に丸め込まれて，複雑度の判定結果ではずれは検出されていない．なお，プロジェクト C の差の原因は，トランザクション数が「単純（3 個以下）」と「平均的（4 個から 7 個）」の境界線を挟むユースケースが二つ検出されたためであった．このように，トランザクション数がタイプの境界値をもつユースケースについては，その情報をユーザに呈示して，ユーザに結果の調整をゆだねる方法が現実的であると考えている．

また，今回の適用事例では，Actor の汎化やユース

ケースの包含や拡張に関する部分が含まれていなかった．したがって，**3.2.3** や **3.3.4** で述べた手法についての有効性の評価は行われていない．これらについての評価は今後の課題の一つである．

6. むすび

本論文では，いくつかの制限を置いた上で，UCP 計測で行われるアクタとユースケースの自動分類手法を提案し，その手法に基づく UCP 計測ツール U-EST を試作した．また，U-EST を日立 SAS で実施した五つのプロジェクトに適用し，U-EST による計測値と熟練者による手動での UCP 計測値が極めて近い結果を得た．

今回の結果より，利用環境は限定されるが，当初の目的である，計測のための開発者の負担増を避けること，計測者による誤差をなくすこと，データの統一的な蓄積の基盤の確立が達成でき，効果的な見積り支援が可能となると考えている．

今回評価したプロジェクト以外にも，人間が書く様々な様式のユースケースシナリオを考慮して，より人間に近い複雑度の分類ができるシステムを構築する必要がある．ただし，結果としてツールの一般性は失われ，ある特定組織の特性に依存したツールになってしまう可能性が高い．今後は，より多くのケーススタディを行い，判定のために用いるキーワードリストの再考，アルゴリズムの改良を行い，アクタについてもより精度の高い複雑度の分類を行って，工数見積りのより効果的な支援を目指していく予定である．また，ツールによる計測精度を向上させるためには，ユーザとのインタラクションは不可欠である．ユーザとツール間のインタラクションに要する工数評価も現場での利用における重要な課題である．

謝辞　本研究は一部科学研究費補助金基盤研究（C）(課題番号：17500022）の補助を受けている．

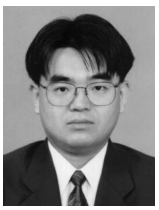
文　献

[1] A.J. Albrecht, “Function point analysis,” Encyclopedia of Software Engineering, vol.1, pp.518–524, 1994.
[2] B. Anda, H. Dreiem, D.I.K. Sjoberg, and M. Jorgensen, “Estimating software development effort based on use cases - Experiences from industry,” Fourth International Conference on the UML, pp.487–504, 2001.
[3] B.W. Bohem, Software Engineering Economics, Prentice Hall, 1981.
[4] A. Cockburn（著），山岸耕二ほか（訳），ユースケース実践ガイド — 効果的なユースケースの書き方，翔泳社，2001.
[5] ソフトウェア・エンジニアリング・センター，IT ユーザとベンダのための定量的見積りの勧め，オーム社，2005.
[6] C. Jones, Estimating Software Costs, McGraw-Hill, 1998. 富野　壽（訳）：ソフトウェア見積りのすべて：規模・品質・工数・工期の予測法，共立出版，2001.
[7] G. Karner, “Use case points - Resource estimation for objectory projects,” Objective Systems SF AB (Rational Software), 1993.
[8] 工藤　拓，松本裕治，“Support Vector Machine による日本語係り受け解析,” 情処学自然言語処理研報，NL-138, 2000.
[9] K. Ribu, Estimating Object-Oriented Software Projects with Use Cases, Master of Science Thesis, University of Oslo, 2001.
[10] G. Schneider and J.P. Winters（著），羽生田栄一（訳），ユースケースの適用：実践ガイド，ピアソン・エデュケーション，2000.
[11] J. Smith, “The estimation of effort based on use cases,” Rational Software white paper, 1999.
[12] CaboCha : Yet Another Japanese Dependency Structure Analyzer, http://cl.aist-nara.ac.jp/ taku-ku/software/cabocha/
[13] http://www.embarcadero.com/products/describe/index.html

（平成 19 年 2 月 6 日受付，7 月 21 日再受付）

津田　道夫　（正員）

昭 45 同志社大・工・電気卒．同年日立製作所入社．平 11 より日立システムアンドサービス（株）所属．平 17 大阪大学大学院情報科学研究科博士後期課程入学．プロジェクト管理，見積りに関する研究に従事．情報処理学会会員．



楠本　真二　（正員）

昭 63 阪大・基礎工・情報卒．平 3 同大大学院博士課程中退．同年同大・基礎工・情報・助手．平 8 同学科講師．平 11 同学科助教授．平 14 阪大・情報・コンピュータサイエンス・助教授．平 17 同学科教授．博士（工学）．ソフトウェアの生産性や品質の定量的評価，プロジェクト管理に関する研究に従事．IEEE, JFPUG，PM，情報処理学会各会員．

松川　文一

平 14 阪大・基礎工・情報卒．平 16 同大大学院情報科学研究科博士前期課程了．同年（株）東芝入社．ビジネスコミュニケーションシステム開発業務に従事．

山村　知弘

平 18 阪大・基礎工・情報卒．同年東京大学大学院情報理工学系研究科知能機械情報学専攻入学．神経インタフェース，有機 EL など MEMS 分野に関する研究に従事．電気学会会員．

井上　克郎　（正員）

昭 54 阪大・基礎工・情報卒．昭 59 同大大学院博士課程了．同年同大・基礎工・情報・助手．昭 59〜61 ハワイ大マノア校・情報工学科・助教授．平元阪大・基礎工・情報・講師．平 3 同学科・助教授．平 7 同学科・教授．工博．平 14 阪大・情報・コンピュータサイエンス・教授．ソフトウェア工学の研究に従事．情報処理学会，日本ソフトウェア科学会，IEEE，ACM 各会員．

英　　繁雄

昭 60 名工大・土木卒．同年日立システムアンドサービス（株）入社．生産技術部所属．Web システム関連の生産技術に関する研究に従事．

前川　祐介

平 12 名大・理・数理卒．同年日立システムアンドサービス（株）入社．生産技術部所属．Web システム関連の生産技術に関する研究に従事．