98-01080

オブジェクト指向ソフトウェアに対する機能量計測に関する研究

楠本　真二　大阪大学大学院基礎工学研究科助教授\\

1　研究調査の目的

近年、コンピュータシステムへの依存度が高まるにつれて、その主要な構成要素であるソフトウェアはますます大規模化・複雑化・多様化してきている。さらに、ソフトウェアの需要量も増大し、開発期間の短縮化が求められるようになってきた。このような背景から、高い品質を持ったソフトウェアを効率良く開発するために、明確な開発計画の下で開発プロセスの全工程を系統づけて管理する必要性が高まってきている。

通常、開発工数や開発規模を予測するのに、まずソフトウェアの規模を見積もり、これに基づいて開発工数と開発期間を予測する手法がとられている[12]。従来、ソフトウェアの規模を表すのにプログラム行数(SLOC)が使われていた。しかし、最近になって、ソフトウェアの機能要件だけを抽出して定量的に計測する「ファンクションポイント法」が世界的に普及しつつある。ファンクションポイント法は、Albrechtによって1979年に提案された[1]。その後、これをベースに種々のファンクションポイント法が提案されている。現在では、IFPUG (International Function Point Users Group)法[2]が事実上の標準となっている。

しかし、ファンクションポイント法には、測定における一般的なルールが述べられているにすぎず、同一プロダクトに対しての計測であっても、計測する人間によって誤差が生じてしまうという問題が指摘されている[5]。例えば、同じ組織内の人間が同じプロダクトに対して測定した場合は12%程度、違う組織の人間が測定した場合は30%以上の誤差が出る[3]という報告がある。

また、要求仕様書から設計仕様書への詳細化を行う際に用いるモデリング言語としてUnified Modeling Language(UML)[11]が提案されており、近年広く用いられるようになったが、UMLで作成された設計仕様書からファンクションポイントを計測するための手法は確立されていない。

そこで、本調査研究では、先ず、UMLで記述された設計仕様書に対してファンクションポイントを計測するための手順を検討し、その手順に基づいて計測ツールを試作することを目的とする。具体的には、UMLで設計仕様書を作成する際に標準として用いられているRational Roseで作成された設計仕様書からファンクションポイントの計測を行うツールの試作を行う。

2　方法\

2.1　概要

ファンクションポイント法は、利用者要求のうちの機能要求仕様の大きさを定量的に計測する手法であり、求められる計測値は、機能量または機能規模と呼ばれる。機能量の計測では、計測対象ソフトウェアの機能のうち、画面や帳票、ファイルなどを通じた情報の入出力に着目し、それらを種類別に数え上げ、種類数を加重合計した値を機能量とする。ただし、実現方法の違いによる影響を除くために、画面などの数は物理設計結果ではなく論理設計レベルで数える。

こうして得られた機能量の規模尺度としての最大の長所は、

●規則にしたがって計測される値(推定値でなく誰が計測しても同じ値が得られる)

●機能仕様にだけ依存(開発環境や開発言語などの技術要件に左右されない)

という二点である[2]。

本研究では、数多くのファンクションポイント法の中から、ファンクションポイント標準化の中心的組織であるIFPUGが定めており、日本においても主流技法として用いられているIFPUG法を使用してファンクションポイントを計測する。

2.2　IFPUG法

IFPUG法は、Albrecht版のファンクションポイント法に対して複雑さの評価の客観化やルールの精密化・適正化などの変更を行なったバージョンである。

IFPUG法は次のStep1～Step7でファンクションポイントを計測する[2]。

Step1　算出種類の選択：算出種類を、アプリケーションファンクションポイント(アプリケーションソフトウェアの大きさを表す)、新規開発プロジェクトファンクションポイント(新規にアプリケーションを開発するプロジェクトの規模を知るために使用するファンクションポイント)、機能改良プロジェクトファンクションポイント(すでに存在するアプリケーションを改良するプロジェクトの規模を知るために使用するファンクションポイント)、の3種類の中から選択する。

Step2　計測境界の設定：ファンクションポイントを計測する対象(範囲)を明確にするため、計測境界の設定を行う。計測境界によって、ファンクションポイントを計測したい対象アプリケーション自体を境界の内部、他のアプリケーションおよびユーザを境界の外部となるように設定する。

Step3　データファンクションの抽出：データファンクションとは、アプリケーション中にあり、ユーザが認識できる論理的な意味でのデータのまとまり(論理ファイル)のことである。データファンクションにはファンクションタイプと複雑さの要素があり、それらを用いてデータファンクションのファンクションポイントが決定される。データファンクションには、内部論理ファイル(計測対象のアプリケーションによってデータが更新される論理的な関連を持ったデータの集合)と外部インタフェースファイル(計測対象のアプリケーションによってデータが参照される論理的な関連を持ったデータの集合)の2種類がある。次に、それぞれのデータファンクションをレコード種類数(データファンクションが持っている、異なる意味合いを持つデータのまとまりの個数)、データ項目数(データファンクションを構成する、ユーザが認識できる論理的なデータ項目の数)という2つのパラメータによって、低・中・高の3段階の複雑さとして重み付けする。ここで、複雑さを決定するために表1を使用する。

表1　データファンクションの複雑さ

RET \ DET	1-19	20-50	51-
1	低	低	中
2-5	低	中	高
6-	中	高	高

RET: レコード種類数　DET: データ項目数

Step4　トランザクションファンクションの抽出：トランザクションファンクションとは、次の(1)～(3)の項目を満たす処理のことである。(1)計測境界外と、状況によって値が変わりうるデータの入出力を行う処理、(2)ユーザにとって意味があり最小作業単位となる処理、(3)処理ロジックおよびデータ項目が、他のトランザクションファンクションと異なる。トランザクションファンクションにも、ファンクションタイプと複雑さの要素があり、それらを用いてトランザクションファンクションのファンクションポイントが決定される。トランザクションファンクションには、外部入力(計測境界外からデータが入力され、データファンクションが更新される処理)、外部出力(計測境界外へのデータ出力を含む処理のうち、出力データに、データを加工した物を含むもの)、外部照会(計測境界外へのデータ出力を含む処理のうち、出力データに、データを加工した物を含まないもの)、の3種類がある。次に、それぞれのトランザクションファンクションを関連ファイル数(対象となるトランザクションファンクションの処理中にデータが更新または参照されるデータファンクションの個数)、データ項目数(計測境界を出入りするデータ項目の個数)、という2つのパラメータによって、低・中・高の3段階の複雑さとして重み付けする。複雑さを決定するためにファンクションタイプ毎に用意されている複雑さ決定表を使用する(表2参照)。外部照会については、処理を入力側と出力側に分け、そのそれぞれについて関連ファイル数およびデータ項目数を算出し、入力側は外部入力の表、出力側は外部出力の表を用いて複雑さを決定し、複雑さの高い方をその外部照会の複雑さとする。

表2　トランザクションファンクションの複雑さ

外部入力の複雑さ

FTR \ DET	1-4	5-15	16-
0,1	低	低	中
2	低	中	高
3-	中	高	高

外部出力の複雑さ

FTR \ DET	1-5	6-19	20-
0,1	低	低	中
2,3	低	中	高
4-	中	高	高

FTR: 関連ファイル数　DET: データ項目数

Step5　未調整ファンクションポイントの算出: Step3, Step4の結果をもとに、種類別重み別に個数を数え、表3を用いて算出した結果が未調整ファンクションポイント(Unadjusted Function Points)となる。

表3　未調整ファンクションポイント算出表

ファンクションタイプ \ 複雑さ	低	中	高	合計
内部論理ファイル	□ × 7= □	□ ×10= □	□ ×15= □	
外部インタフェースファイル	□ × 5= □	□ × 7= □	□ ×10= □	
外部入力	□ × 3= □	□ × 4= □	□ × 6= □	
外部出力	□ × 4= □	□ × 5= □	□ × 7= □	
外部照会	□ × 3= □	□ × 4= □	□ × 6= □	
			未調整 FP	

Step6　調整係数の算出: 未調整ファンクションポイントは「データのまとまり」と「データの出入り」のみに着目した値で、性能、信頼性、ユーザインタフェースなどについては考慮されない。そこでシステム特性をファンクションポイントに反映させるために、表4に示すシステム特性の14項目を6段階で評価し、その結果から調整係数を算出する。調整係数は、ファンクションポイント算出の際に未調整ファンクションポイントを補正する役割がある。

表4　システム特性の14項目

1	データ通信機能	8	オンライン更新
2	分散データ処理	9	複雑な処理
3	性能条件	10	再利用性
4	高負荷構成	11	インストレーションの容易さ
5	トランザクション率	12	運用の容易さ
6	オンラインデータ入力	13	複数サイト
7	エンドユーザの効率	14	変更の容易さ

$$\text{調整係数} = 0.01 \times \text{影響度の合計} + 0.65$$

Step7　最終ファンクションポイントの算出:未調整ファンクションポイントと調整係数を用いて最終ファンクションポイントを算出する。Step1の算出種類によって算出の方法が異なる。

- ●アプリケーションFP = 未調整FP + 調整係数
- ●新規開発FP = (未調整FP + 移行分未調整FP)×調整係数
- ●機能改良FP = (変更部分の新未調整FP + 追加部分の未調整FP + 移行分未調整FP) × 新調整係数 + 削除部分の未調整FP × 旧調整係数

2.3　UML

UMLは、オブジェクト指向開発のためのモデリング言語である。1994年に、代表的なオブジェクト指向開発技法であるBooch法[4]を提案したGrady BoochとOMT法[4]を提案したJames Rumbaughらが当時乱立していたオブジェクト指向方法論の統合を開始したのが始まりであり、1997年9月にUML1.1が発表された[6]。UMLでは、ユースケース図、シーケンス図、コラボレーション図、クラス図、オブジェクト図、ステートチャート図、アクティビティ図、コンポーネント図、配置図、の9種類の図が成果物として生成される[7, 9]。

本研究ではUMLを用いたオブジェクト指向分析、設計ツールとして標準的に用いられているRational Rose(Rational社)で作成された設計仕様書に対してファンクションポイントの計測を行う。

以下に、9種類の図のうち、ファンクションポイント計測に有用と思われるクラス図、シーケンス図について説明する。

2.4　シーケンス図

シーケンス図(sequence diagram)は、相互作用を時系列に並べたものである。特に、相互作用に関係しているオブジェクトを生存線によって示し、オブジェクトが交換するメッセージを時系列に並べてあらわす。オブジェクトとはクラスの特

定のインスタンスである。

シーケンス図では、システムの外側に存在する実体をあらわすアクタが、スティックマン(線で人物を単純化したアイコン)として表される。また、その他のオブジェクトはオブジェクトの名前に下線をつけたものを長方形で囲んだものとして表される。また、オブジェクトの役割は、生存線と呼ばれる垂直な破線で示され、アクションを実行している期間を活性化と呼ぶ。活性化は縦に細長い長方形で示され、長方形の上部は開始時に、下部は完了時に合わされる。水平方向の矢印は、メッセージの送信であり、矢印はメッセージを受け取るオブジェクトに向かって指される。

図1は、学生が受講科目を登録するシーンにおけるシーケンス図の一例である。アクタである学生および受付、その他のオブジェクトである登録システムと学生情報DBが存在する。学生が受講科目等を書いた登録票を提出し、受付がシステムに対してその学生の情報を登録し、結果を通知するまでが時系列順に示されている。

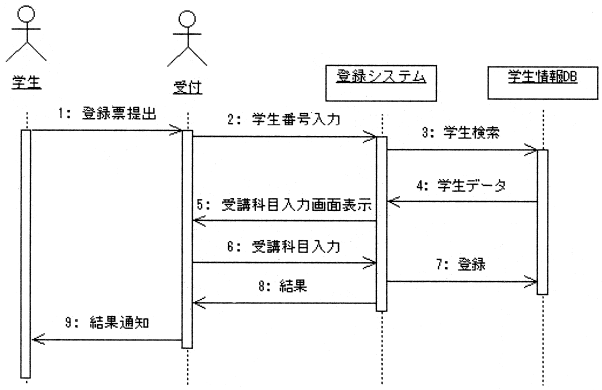

図1　シーケンス図

2.5　クラス図

クラス図(class diagram)は、モデルの静的構造、特に、そこに存在するクラスやタイプ、その内部構造、およびほかの事項との関連について示したものである。クラスは、実線の枠をもつ長方形で書き、水平線で分割した3つの区画からなる。上部の区画にはクラス名、中間の区画には属性のリスト、下部の区画には操作のリストを記述する。

図2は、例としてスーパークラスPersonおよびサブクラスのStudent、Teacherの関係を示した図である。Personクラスには属性として名前、住所が示され、また、それらの属性を設定/取得する操作が示されている。

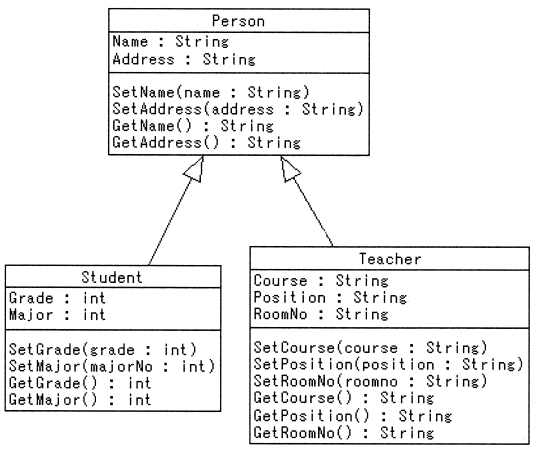

図2　クラス図

2.6　基本方針

IFPUG法では、先に挙げた7つのステップ(Step1：算出種類の選択、Step2：計測境界の設定、Step3：データファンクションの計測、Step4：トランザクションファンクションの計測、Step5：未調整ファンクションポイント算出、Step6：調整係数の算出、Step7：最終ファンクションポイント算出)でファンクションポイントを計測する。ここでは計測ツールが採用している方針に沿って説明する。

Step1 (算出種類の選択) : 算出種類は新規開発プロジェクトファンクションポイントのみとする。
Step2 (計測境界の設定) : 設計仕様書の図に表れるオブジェクトの型によって境界内であるか境界外であるかを決定する。アクタであるオブジェクトは計測境界外、それ以外のオブジェクトが計測境界内であるとする。
Step3 (データファンクションの計測) : 指定したモデルファイル中のデータファンクションを、3.1で説明するルールに従って抽出する。
Step4 (トランザクションファンクションの計測) : 指定したモデルファイル中のトランザクションファンクションを、3.2で説明するルールに従って抽出する。
Step5 (未調整ファンクションポイント算出) : Step3、Step4の結果をもとにファンクションタイプ別複雑さ別に個数をカウントし、それぞれに対して重み付けをし、加え合わせて未調整ファンクションポイントを算出する。
Step6 (調整係数の算出) : システム特性の影響度をユーザが入力し、調整係数を算出する。
Step7 (最終ファンクションポイント算出) : 未調整ファンクションポイントと調整係数から最終ファンクションポイントを算出する。

3 結果

3.1 データファンクション計測方針

データファンクションの抽出にあたっては、アクタ以外のオブジェクトのうち、属性を持ち、かつ他オブジェクトとデータの入出力を行うオブジェクトを候補とする。これらのオブジェクト以外がデータファンクションとなることはないが、その逆(データファンクションに該当しない場合)があるので、ユーザが適宜除外するものとする。ツールは、上記に該当するものを候補として提示するにとどめ、その中からユーザが選択するものとする。

ファンクションタイプは、シーケンス図、コラボレーション図において、対象となるデータファンクションの属性を変更するような動作を、一ヶ所でも行っている(すなわち、データファンクション中のデータを更新している)ならば内部論理ファイルとし、そうでなければ外部インタフェースファイルとする。

複雑さの判定は、レコード種類数とデータ項目数で決定される。レコード種類数は、該当オブジェクトのクラスの属性が二つ以上のサブグループに分けられるならばそのグループ数とし、分けられなければ1とする。

データ項目数は、該当オブジェクトが属するクラスの属性の数とする。派生クラスの場合は、基本クラスの属性も含めて計算する。

最後にレコード種類数とデータ項目数を元に複雑性判定マトリックスに基づきファンクションの複雑性を決定する。

3.2 トランザクションファンクション計測方針

IFPUG法でのトランザクションファンクションの抽出ルールに従い、シーケンス図あるいはコラボレーション図内において、データファンクションとして指定されたオブジェクトが相互作用(メッセージのやり取り)を行っている部分を抽出し、それらをトランザクションファンクションの候補とする。メッセージに引数がない場合は、データの入出力は無いと考え、EI,EO,EQのいずれにも該当しないのでトランザクションファンクションとしない。また、データファンクションでないオブジェクト同士の相互作用はトランザクションファンクションではないと判断する。また、コラボレーション図から計測する場合はすべてのメッセージの時間的順序が記述されている必要がある。

まず、ファンクションポイントを求める際に必要な記述面での制約を列挙する。

- ●メッセージを送信したときに、それに対する処理結果などで意味のある応答がある場合、それをメッセージとして送信する。
- ●データのやり取りはメッセージの引数で表現する。
- ●属性と同じ名前の変数がメッセージの引数として出力されている場合はそのデータは無加工で出力されているものとする。
- ●引数およびメッセージ名がいずれも等しいメッセージは同じ操作であるとみなす。

ファンクションタイプの判別は、データファンクションの更新の有無および出力されたデータ項目を比較することによって判別できる。ここでは6つのルールを定義し、それに基づいてファンクションタイプ、データ項目数および関連ファイル数を判別した。

(1)ルール1(アクタからDFへメッセージが送られている場合) : 外部入力とする。引数がなければ入力とはみなさない。データ項目数はそのメッセージの引数の数とし、関連ファイル数はメッセージを受け取っているDFのみ、すなわち1とする。
(2)ルール2(DFからアクタへメッセージが送られている場合) : そのメッセージの引数(複数の場合はそのすべて)がDFの属性そのものであれば外部照会とし、そうでなければ派生データが含まれていると判断し外部出力とする。データ項目数はそのメッセージの引数の数とし、関連ファイル数は1とする。

⑶ルール3(アクタからDFへメッセージが送られ、そのDFから同じアクタへメッセージが送られている場合) : DFからアクタへのメッセージに着目し、メッセージの引数(複数の場合はそのすべて)がDFの属性そのものであれば往復のメッセージ二つをまとめて一つの外部照会とし、そうでなければ派生データが含まれていると判断し、往復のメッセージ二つをまとめて一つの外部出力とする。すなわち、入力側のメッセージについては参照のためのキーの入力等であるとみなす。ただし、DFからアクタへのメッセージに引数が無い場合は外部入力と判断する。データ項目数はそのメッセージの引数の数とし、関連ファイル数は1とする。

⑷ルール4(あるアクタ1からDFへメッセージが送られ、そのDFから別のアクタ2へメッセージが送られている場合) : アクタ1からDFとDFからアクタ2であると考える。すなわちルール3は適用しないでルール1とルール2の組み合わせであると考えて、別々に処理する。

⑸ルール5(Actor1→DF1→DF2…→DF1→Actor1とメッセージが遷移している場合) : 各オブジェクトに対して入出力の組となるメッセージ対に対して、ルール1〜3を適用する。

⑹ルール6(メッセージが遷移しているが、直接Actorと関与していないDFが存在する場合) : メッセージが遷移しているDFについて、関連ファイル数として算出する。データ項目については他のルールに準拠する。

3.3 ツールの概略

Windows95以後のWindows系OSで動作するツールをMicrosoft Foundation Classライブラリを用いてVisual C++で実装した。プログラムサイズは約7000行である。ツールの入力はRational Rose 4.0およびRational Rose 98で作成された設計仕様書である。まずはじめにRational Roseで作成された設計仕様書を指定すると、データファンクションの候補を提示するので、その中からデータファンクションを指定する。その結果、ファンクションポイントの計測結果(未調整ファンクションポイント)とその計測の元となるトランザクションファンクション、データファンクションおよびそれらのファンクションの詳細な情報を出力する。

また、未調整ファンクションポイント出力後、調整係数入力画面で調整係数を入力すると最終ファンクションポイントが算出される。

3.4 システム構成

システム構成を図3に示す。

- **解析部**：Rational Roseの出力ファイルを構文解析し、ファンクションポイント計測のために必要であるデータを抽出する。
- **解析DB**：解析部で抽出したデータを格納しておく。
- **計測部**：ファンクションポイントを計算する。
- **計測DB**：計測結果(ファンクションポイント、データファンクション、トランザクションファンクション)を格納しておく。
- **インタフェース部**：計測結果(ファンクションポイント、データファンクションの候補、トランザクションファンクションの候補)を表示したり、ユーザが変更することができるインタフェースを提供する。

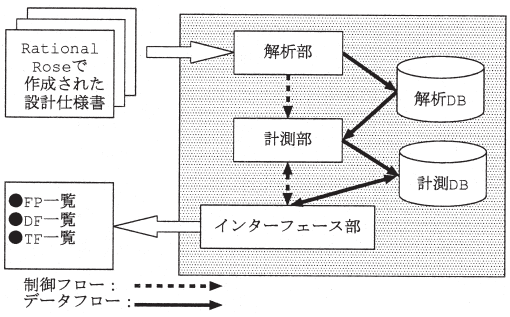

図3　システム構成

3.5 適用結果

ツールの計測したファンクションポイント値が妥当なものであるかを評価するために、計測ツールを数種の設計仕様書に対して適用した。具体的には、Rational Roseで作成された、購入業務、発注業務、在庫管理業務を対象とし、ツールの計測したファンクションポイント値とファンクションポイント計測の熟練者が測定したファンクションポイント値との比較を行った。

なお、ツールを用いてファンクションポイントを計測する際には、データファンクションを一部指定しなければならないが、今回は、計測ツールが提示したものをそのまま使用した。また、計測は未調整ファンクションポイントの算出までとした。

購入、発注、在庫管理業務について、それぞれのシステムの設計仕様書に対してツールの計測したファンクションポイント値と熟練者が計測したファンクションポイント値を表5に示す。表5の各ファンクションの計測結果を比較すると、いずれもツールの算出した値のほうが熟練者の算出した値と比べて若干低い値となった。

そこで、トランザクションファンクションおよびデータファンクションの一つ一つについて、熟練者が計測したものとツールが計測したものを比較し、これらの差異の原因について考察を行った。一つ目の原因としては、従来ファンクションポイントは、ユーザから見える部分の処理のみをトランザクションファンクションとして算出するものであるが、実際の現場では、納期や規模等を見積もる必要性から、内部処理(アクタが関与しないデータの受け渡し)に関してもトランザクションファンクションとして算出したほうが都合がよいと判断し、内部処理についてもトランザクションファンクションとして抽出していた。次に、熟練者は、設計仕様書の記述にシステム化する際に必要と思われる処理を考慮して計測していた。

何れの原因も、ツールに依存するものではないため、これらの原因の対策を行い、再度ツールに適用した。その結果、熟練者の計測結果と設計仕様書変更後のツールのFP値は一致した。

表5　適用結果

	熟練者の FP 値			ツールの FP 値		
	購入	発注	在庫	購入	発注	在庫
DF の FP 値	14	29	26	14	29	26
TF の FP 値	18	63	33	15	47	30
FP 値の合計	32	92	59	29	76	56

3.6　まとめと今後の課題

UML(Unified Modeling Language)で記述された設計仕様書に対してファンクションポイントを計測する手順を提案し、その手順に基づいてファンクションポイントを求めるツールを試作した。さらに、実際の開発現場で作成された設計仕様書に対してツールを適用し、ツールが計測した値と、ファンクションポイント計測の熟練者が測定した値との比較を行った。

今後の課題としては、より多くのサンプルについて熟練者が測定した値と比較し、本手法の有効性をより確かなものにするように改良することがあげられる。

参考文献

[1] A. J. Albrecht : *Function Point Analysis*, in John J. Marciniak, editor, Encyclopedia of Software Engineering, vol.1, John Wiley & Sons, pp.518-524, (1994).

[2] IFPUG :“ Function Point Counting Practices Manual, Release 4.0 ”, (1994).

[3] B. A. Kitchenham :“ The problem with function points ”, IEEE Software, Vol.14, No.2, pp.29-31, (1997).

[4] Lockheed Martin Advanced Concepts Center :“ Succeeding with the Booch and OMT Methods : A Practical Approach ”, Addison-Wesley Publishing Company, (1996).

[5] G. C. Low and D. R. Jeffery :“ Function points in the estimation and evaluation of the software process ”, IEEE Transactions on Software Engineering., Vol.16, No.1, pp.64-71, (1990).

[6] T. Quatrani :“ Visual Modeling with Rational Rose and UML ”, Addison Wesley Longman, (1998).

[7] Rational Software Corporation :“ UML Notation Guide, version 1.1 ”, (1997).

[8] Rational Software Corporation :“ UML Notation Guide, version 1.3 ”, (1999).

[9] Rational Software Corporation :“ UML Semantics, version 1.1 ”, (1997).

[10] Rational Software Corporation :“ UML Semantics, version 1.3 ”, (1999).

[11] S. Zamir : Handbook of Object Technology, CRC Press, (1999).

[12] 西山 茂 :“ ソフトウェア規模の見積り技術の最近の流れ－行数による評価から機能量による評価へ ”, 情報処理学会誌、Vol.35, No.4, pp.289-298, (1994-04).

＜発 表 資 料＞

題　　名	掲載誌・学会名等	発表年月
A function point measurement tool for UML design specifications	Proc. of Sixth International on Software Metrics, pp.62-69（Florida, Nov.,1999）	1999年11月
オブジェクト指向開発におけるフォールト発生早期予測手法の一提案	情報処理学会オブジェクト指向 '99シンポジウム論文集, pp.145-152	1999年7月