早稲田大学大学院理工学研究科

博 士 論 文 概 要

論 文 題 目

ソフトウエアの複雑さと品質の関連に関する研究

申 請 者
高橋 良英
Ryouei Takahashi

1997年 2月

1．研究の目的と背景

ソフトウエアコストの大半は保守コストであることが知られている．ソフトウエア保守コストを削減するために，保守性の高いソフトウエア開発が望まれている．ソフトウエアの保守では，流用・再利用するソフトウエアの設計書やソースコード，マニュアルを理解して改造機能や試験項目，試験手順を決定しているので，流用・再利用するソフトウエアの複雑さが，ソフトウエア保守コストに大きな影響を及ぼしていると考える．この複雑さは，プログラマやシステムアナリストの能力・設計手法（構造化分析設計手法等）・言語・保守ツール・開発期間制約・OS(UNIX/MS-DOS）等の環境要因（開発プロセスの複雑さ）とは異なる独立な要因と考える．実際，これらの要因の中で複雑さがコストを規定する大きな要因であることを，Boehm は「Software Engineering Economics」で述べている．しかしながら，その尺度は定性的であり，より定量的・計測可能な複雑さの尺度が望まれる．更に，Boehm や Walston & Felix等の従来のソフトウエア開発モデル（Water-Fallモデルまたは機能改良型スパイラルモデルを想定）では，ソフトウエア保守コスト見積もり時，流用部からの影響を充分考慮したモデルとなっていないため，その影響を考慮したソフトウエアの生産性・品質の評価モデルが必要となる．ソフトウエア保守品質評価メトリック（解析性，変更容易性，試験の容易性）として，具体的に障害件数，障害密度，コード修正改良時間（障害解析時間）を考える．この尺度は，「ソフトウエア品質の特徴づけとその使い方のガイドラインに関する国際標準」ISO9126 における保守性の外部尺度でもある．本研究の目的は，ソフトウエア生産物（ソースコード，仕様書，マニュアル）の複雑さと，ソフトウエア保守品質の関連を，データ分析とモジュール間インタフェースの複雑さの研究により，明らかにし，プロジェクト管理手法や設計手法に反映することである．

2．研究の概要

本研究では，ソフトウエアの複雑さによりソフトウエア品質を計画的に評価・管理・制御するシステムを考える．当評価システムでは，管理対象のソフトウエアシステムを階層的なモジュール階層に分割して評価する．モジュールは論理的な概念であり，最小の単位はC 言語の関数相等の大きさ（数十ステップから数百ステップ位の大きさ）である．関数の機能階層的な集合の要素もモジュールと呼び，この単位は設計時のモジュール分割の結果でもある．当評価システムでは，ソフトウエアライフサイクルの比較的初期の段階で複雑さを計測し，複雑度の高い重点管理モジュールを特定化する．モジュール再分割・統合，試験項目の充実，抽出障害件数の目標値設定と実績値管理を行う．複雑さは C言語のソースコードから測定する．コンパイルが完了した時点から複雑度計測とソフトウエア品質（障害）を管理し始め，ソフトウエア品質と複雑度の関係をデータ分析により明確化する．過去のデータを蓄積し，将来に向けてのソフトウエア品質の予測制御を行う．障害の発生しやすいモジュールとそうでないモジュールの切りわけ制御を，複雑度計測を中心により行う．

現状のソフトウエア品質評価モデルでは，モジュール内とモジュール間の複雑さを測る幾つかの尺度が提案され，ソフトウエア品質との因果関連について研究されているが，プロジェクト毎にその評価結果はまちまちであり，また複雑さの尺度としても充分ではない．本研究ではモジュール内の複雑さのみならず，モジュール間のインタフェースの複雑さを考慮した新しいソフトウエア品質評価システムを提案し，その有効性をデータ分析により検証している．

本研究の概要を，以下に整理する．

（1）モジュール間のインタフェースの複雑さを測る尺度の研究

複雑さの尺度は，モジュール内の複雑さを測るコードメトリックと，モジュール間の構造の複雑さを測る構造メトリックの二つに分かれる．コードメトリックとして，規模の大きさ，条件文数を数えるサイクロマチック数（McCabe），条件文の影響範囲（スコープ）を測定するネスト，オペレータやオペランドの定義数，出現回数等を測定するHalsteadの尺度が知られている．構造メトリックとして，情報の流れの複雑さ（Henry & Kafuraのfan-in/out数）や関数実行条件数(McCabe のS_1)，関数実行条件の制御変数の複雑さ(McClure）が知られているが，ソフトウエア品質(障害密度）との因果関係ではコードメトリックよりも構造メトリックの方が効くことが知られている．最近はコードメトリックと構造メトリックを混合したメトリックも考えられている．しかしながら，現状の構造メトリックでは障害密度との因果関連を充分表現できないことがわかっており，モジュール間のインタフェースの複雑さを測定する新しい尺度を研究する必要がある．本研究では，モジュール間のインタフェースの複雑さを測定する一つの尺度として，モジュール間に渡って影響を与える条件文と，モジュール内にその影響を閉じる条件文を分離カウントする尺度（応用サイクロマチック数と呼ぶ）を考案し，当尺度がソフトウエア品質に大きな影響を与えていることを実験的に検証している．

（2）ソフトウエア品質判別問題への判別分析手法の適用可能性に関する研究

ソフトウエア品質評価モデルとしては，様々なモデルが考案されている．①多変量解析をベースとしてソフトウエア品質を予測制御するモデル（判別分析，重回帰分析（相関分析を含む），分散分析，主成分分析，因子分析，ノンパラメトリック手法，正準相関分析，クラスタ分析），②数値解析をベースとしてソフトウエア品質を予測制御するモデル，③ソフトウエア分類木や複雑度相互依存度等エントロピーベースでソフトウエア品質を特徴づけるモデル，④ニューロコンピュータのバックプロパゲーション手法を応用して，ソフトウエア品質判別関数の重み付けを決定するモデル，⑤情報の流れの複雑さに着目したモデル，⑤Goals& Questions方式で問題解決しようとするモデル，⑥システム成熟度，ソースコード，ドキュメントからシステマチック階層に評価するモデル等多々ある．

本研究では，上記のうち，多変量解析手法をベースとしたソフトウエア品質評価モデルとする．多変量解析の中でも，特に，品質の良いモジュールと悪いモジュールの判別に有効な判別分析，欠陥と複雑度の関連を見いだすのに有効な重回帰分析等をベースとしたソフトウエア品質評価モデルを考える．判別分析モデル，重回帰分析モデルの目的変数は，ソフトウエア品質（障害密度や障害件数）である．重回帰分析モデル，判別分析モデルの説明変数は複雑度である．この時，説明変数の選択方法，すなわち複雑度の主成分の選択方法が課題となる．複雑度の主成分は，通常，変数増減法（F 検定値による方法）や主成分分析や因子分析で抽出される．抽出した主成分自身の線形結合を考え，これを相対メトリック(relative complexity metrics）と呼ぶ．しかしながら，主成分分析や因子分析，変数増減法で抽出された複雑度の主成分の個数について客観的な判断基準は現状ない．この問題を解決するために，判別分析モデルや重回帰分析モデル等における複雑度の主成分を，情報量規準AIC (Akaike Information Criterion)を適用して

決定する方式を採った．情報量規準AIC は，最尤推定法をベースに最小の複雑度のパラメータによりモデルを選択する方法である．AIC = －２*(最尤モデルの平均対数尤度）＋２（自由パラメータ数）で定義される．最小AIC を満たすモデルが真のモデルに最も近いモデルとして選択される．これより，精度の高くかつ安定したソフトウエア品質判別モデルを構築することを可能とした．

３．本書の構成

本書は，以下の構成としている．

（１）第１章「序論」 ―― 研究の目的について整理している．

（２）第２章「研究の位置づけ」 ― 当章では，ソフトウエア品質の特徴づけとその利用方法に関するガイドライン(Software Evaluation: Quality Characteristics and Guide fot their use)ISO9126 [90, 90-1, 90-2]での研究の位置づけ，複雑さの尺度の中での研究の位置づけ，多変量解析手法の中での研究の位置づけを整理している．

（３）第３章「研究の動機」 ― ソフトウエアコスト見積りモデルを解く過程の中で，①流用部と改造部のインタフェースの複雑さを測る具体的な尺度の研究が必要であること，②ソフトウエアの生産性や品質に効く複雑度や規模の主成分を抽出する統計的手法について研究すること，の動機を見いだした．当章では，研究の動機に至った背景を述べている．

（４）第４章「前提条件」 ― 本章では，以降の章を論じる上で前提となるソフトウエア開発プロセスのモデル化とネック項目について述べている．本モデルの特徴は，複雑度による障害の管理を重視している点である．

（５）第５章「ソフトウエア品質判別問題への判別分析手法の適用可能性に関する研究」 ― 本章では，ソフトウエア品質判別問題への判別分析手法の適用可能性について述べている．判別分析手法をソフトウエア品質予測制御問題に応用する場合の大きな課題の１つは，ソフトウエア品質を規定する複雑度の主成分の抽出方法（最適判別モデルの決定）がある．ソフトウエア品質を複雑度から予測する判別分析モデルにおいて，品質を規定する複雑度の主成分を情報量規準AIC （Akaike Information Criterion）を用いて選択する方法の考え方とその妥当性を，Wilks のΛ検定との関連により，理論的に検証している．２つの事例研究を上げて，その妥当性を実験的に検証している．

（５）第６章「モジュール間インタフェースの複雑さを測定する尺度に関する研究 ― 本章では，モジュール（複数のルーチン（ C言語の関数相等）から成る)）間のインタフェースの複雑さを測定する尺度として応用サイクロマチック数を提案し，その背景と考え方を整理している．応用サイクロマチック数は，McCabeの設計時の複雑さを測る尺度integration complexity measure S_1 を，条件文のスコープを考慮して，モジュール内の制御構造の複雑さからモジュール間の制御構造の複雑さに拡張した尺度である．応用サイクロマチック数では，モジュール間のインタフェースの複雑さを，他のモジュール間にそのスコープがまたがるルーチン実行条件と当該モジュール内のルーチンの実行を制御する他のモジュール内で定義された実行条件の和として測定する．２つの研究事例により，当尺度の有効性を検証している．

＜キーワード＞ソフトウエア品質，複雑さ，C 言語，判別分析，重回帰分析，情報量規準AIC ，モジュール間インタフェースの複雑さ，応用サイクロマチック数，ソフトウエア再利用技術