氏名（本籍）	金井（かない） 敦（あつし） （新潟県）
学位の種類	博士（情報科学）
学位記番号	情 第24号
学位授与年月日	平成14年2月14日
学位授与の要件	学位規則第4条第2項該当
最終学歴	昭和57年3月東北大学大学院工学研究科博士課程前期2年の課程
論文題目	ソフトウェア開発アーキテクチャに関する研究
論文審査委員（主査）	東北大学教授 中村維男　東北大学教授 金子誠 東北大学教授 静谷啓樹

論文内容要旨

1．緒論

リソースを多量に消費するソフトウェア開発においては、効率的にソフトウェアを開発する技術が重要である。その具体的方法として、開発期間や開発工数を削減する開発プロセス、手法、品質管理技術など幅広く様々なアプローチが試みられている。これらの技術は、プログラミング言語、ソフトウェア方式やコンピュータ方式などソフトウェアそのものの技術ではなく、それらの利用の仕方や手順あるいは評価に関する技術であると言える。このような技術を利用することにより、闇雲にソフトウェアを開発するのではなく、ソフトウェアの開発思想に基づいた体系的な開発が可能となり、大幅な生産性向上が期待できる。本論文は、このようなソフトウェア開発方法の設計思想を取り扱う研究であり、ソフトウェアそのものの技術であるソフトウェアアーキテクチャ技術と区別して、この技術をソフトウェア開発アーキテクチャ技術と呼び、本論文のテーマとした。

本論文では、開発プロセスおよび工程管理の基礎となる工数モデルや進捗モデルなどを提案した。具体的には、まず、平行開発法について開発期間の特性の評価を行い並行開発の一般的な傾向について明らかにした。続いて、開発期間を短縮化するために新しい概念による平行開発手法を提案した。また、ソフトウェア開発において、多く行われているにも関わらず的確な工数モデルが存在しなかったソフトウェア移植について工数モデルを提案し、さらに、リソースが多量に消費されるにも関わらず進捗管理などを的確に行うモデルが存在しなかった試験工程進捗モデルについて精度の高いモデルを提案した。これらモデルにより、ソフトウェア開発期間の短縮と的確な管理が可能となる。

2．開発プロセスと開発期間評価

開発プロセスを複数の開発アクティビティに分割しその開発アクティビティを並行に実行することにより開発期間を短縮させる並行開発における開発期間などの特徴について一般的な評価を行った。ソフトウェアの並行開発法は、古くから研究されている計算機アーキテクチャ分野における並列処理による処理速度向上技術と対応させて考えるとわかりやすい。計算機アーキテクチャの分野では、ベクトル計算やパイプライン手法など多様な並列計算手法があるが、より汎用的な手法としてデータフローアーキテクチャが提案されている。データフローアーキテクチャはベクトル計算やパイプライン手法を包含した一般的手法であり、どのような局面でも最大限の並列性を引き出すことができる。一方、ソフトウェア開発では、厳密な開発期間はあまり評価されずに、経験に頼って並行開発が行われているのが実情であった。このような観点から、ソフトウェアの並行開発について、計算機アーキテクチャの研究成果を応用して開発プロセスを並行に実施した場合の開発期間評価と特性の分析を一般的に明らかにすることは、ソフトウェアの並行開発を行う場合に大きな指針となる。

具体的には、ソフトウェア開発を並列計算の様子に対応づけ単純な２進木にモデル化し、それぞれのノードでの処理時間分布を与えることにより全体の処理時間を求めることにより分析した。処理時間分布としては、処理時間の確率分布が連続な場合と２種類の処理時間を持つ離散的な場合について一般式を示した。さらに、離散系の場合について、具体的な数値例を用いて総実行時間と平均実行時間を評価した。

その結果、ソフトウェアの並行開発においては分割した開発アクティビティの中でも最も遅い開発アクティビティが全体の開発時間を大きく左右することを示した。これは、闇雲に開発期間を短縮しようとして開発アクティビティを細かく分割しても遅い開発アクティビティが混在していると短縮効果が出にくい事を示しており、開発プロセスを設計する時の重要な開発アクティビティ分割指針を与えている。

3．畳込開発法

従来は目的機能を独立なモジュールとして小機能に細分化し、それらモジュールについて並行開発を行い開発期間を短縮させる手法が主に試みられていた。しかし、この方法は独立したモジュールとしてソフトウェアの塊を切り出せない場合には適用できずまた切り出せる単位も大きいため、開発の効率化に限界があった。そこで、本開発手法では、機能的に完全に独立でない場合でも並行開発を可能にし、さらに開発期間を短縮する手法を提案した。これは、仕様レベルで開発工程を分割し仕様設計時点から並行に開発する事が可能である新しい開発方法論である。この開発手法について、開発手順を明確に規定するとともに、開発期間および開発工数についてモデルを構築した。さらに、実際のアプリケーションについて開発期間および工数を評価した。

具体的には、仕様を部分仕様に分割する基準や条件などを示し、次に、それに基づいて、分割された仕様がその分割された仕様毎に設計、コーディング、試験と開発され、それを合成して一つの機能とする手法について詳細に規定した。また、本開発法について、その

開発期間と工数変化を理論的に求めた。さらに、実際の音声蓄積サービスを例にした場合、正常系、準正常系、異常系と3ステップで仕様分割すると旨く行くことを示し、具体的に開発期間と工数変化を評価した。

その結果、理論的には、理想的な場合で開発期間を 50%程度まで短縮できることを示した。また、音声蓄積サービスの場合は16%程度開発期間を短縮可能であることが分かった。さらに、部品化効果および工数変化の模様についても考察し、工数については、ステップを分割した損失は比較的少なく、工数変化も少ないため、工数予定が立てやすいことが分かった。

この手法は、従来のモジュール分割による平行開発と共存が可能な手法である。このため、この成果は、従来の並行開発手法で限界であると思われていた開発期間をさらに短縮することが可能であることを示したものであり、この成果の適用によりさらなるソフトウェア開発効率の向上が期待される。

4．変換項目を考慮した移植工数モデル

ソフトウェア移植プロセスを解明し、移植工数モデルを提案した。ソフトウェア開発は、新規開発、すでに存在するソースプログラムや部品の再利用を伴う開発、バージョンアップ、ほとんどの機能はそのままにＯＳなどの走行環境を変更する移植など様々な開発形態がある。これらの中で、ソフトウェアの新規開発とソフトウェア移植ではその開発方法がまったく異なる。これに対して、再利用を伴う開発形態や既存のソフトウェアの機能拡張であるバージョンアップは、新規開発と移植の中間的な開発スタイルと見ることができる。従来、最も基本である新規開発時の工数見積もりモデルは多く提案されて来たが、ソフトウェアの移植は見落とされがちであった。新規開発と移植は両極にあり移植における工数モデルを検討しておくことは、中間的な開発スタイルにも応用が利くため重要である。このソフトウェア移植作業について、移植プロセスを解明し、移植工数を見積るモデルを実測データを元に提案した。

具体的には、移植プロセスを詳細に分析し、移植プログラム規模に依存する作業と、移植プログラムに内在する変換項目数に依存する作業に分類した。それぞれの作業について、実際に、それぞれの要因に依存していることを実データの統計分析により証明した。さらに、実データを分析することより、プログラム規模依存度および変換項目数依存度を定量的に求め、プログラム移植工数をプログラム規模と変換項目数をパラメータとして求めるモデルを完成させた。本モデルを実データに適用し、本モデルを検証するとともにプログラム規模から単純に工数を見積もる場合と精度を比較評価した。

その結果、プログラム規模のみから予測する手法に比べて41%程度正確な工数の評価を可能にすることが分かった。この成果により、従来プログラム規模のみをよりどころとして見積もられていた移植工数をより正確に見積もることが可能になり、結果として的確な工程管理とコスト削減が可能になる。

5．プログラム品質を考慮した試験工程進捗モデル

試験工程はソフトウェアの開発工程の中で特にリソースが多く必要になる工程であり、

リソースを有効に生かす工程管理が重要となる。しかし、試験工程に関しては、品質管理の観点からソフトウェアの信頼性がどのように向上していくかといった観点からの研究はあるが、試験工程の進捗や工数を直接取り扱えるモデルの研究はほとんどなかった。本研究では、試験工程における進捗を直接評価できる新しいモデルを実データに基づいて提案した。

具体的には、ある期間に着目すると投入される工数は一定であり、その工数の範囲内で試験項目を処理する作業と発見したバグを処理する作業が行われるという基本的性質に着目してモデルを構築した。バグが多く発見されるということは、プログラム品質が悪いということであり、結果として、試験進捗がプログラム品質に左右されるというモデルである。本モデルでは、工数分布曲線がベースとなりその形がプログラム品質に依存して変化し試験進捗となる。本コンセプトに基づき定式化し、実際のソフトウェア開発のデータを用いて本モデルの検証を行いモデルが妥当であることを示した。さらに、検証に用いたデータとはまったく異なるデータを用いて、本モデルの適用性を評価するとともに、試験工程期間を予測する手法を提案した。

その結果、プログラム品質を考慮した本モデルは考慮しない場合に比べて 21%程度正確に試験進捗状況を表現できることを示した。本成果は、試験工程の進捗および工数管理をより的確に行うためのベースとなる基礎理論を与えている。この成果を応用することにより今後様々な、試験工程管理手法を生み出すことが可能となる。

6．結論

本研究により、ソフトウェア開発の効率化に貢献する重要な開発手法およびモデルを提案することができた。実際のソフトウェア開発現場では、場当たり的に闇雲に開発する傾向があるが、開発プロセスを明確な思想（ソフトウェア開発アーキテクチャ）に基づいて進めることにより、ソフトウェア開発効率を飛躍的に向上することが可能となる。

WWW の登場によってソフトウェア開発のやり方は従来の開発と比較して大幅に異なって来ており、ソフトウェア開発アーキテクチャは混沌とした状態である。従来型の開発は依然として重要ではあるが、今後は WWW アプリケーションなどの比率がさらに増して来ると思われる。このため、これまでの従来型の成果をベースに、新しい開発スタイルに対応したソフトウェア開発アーキテクチャの構築が望まれている。本研究はこのような新しい開発スタイルを生み出す原動力にもなることを期待するものである。

本研究がこのようなソフトウェア開発の効率化に貢献し、ひいては多くのソフトウェアが生み出され、人類の生活がますます豊かになることを期待するものである。

論文審査の結果の要旨

ソフトウェア開発には膨大な労力が必要とされ、生産性を向上させることはコスト削減に大きな効果がある。生産性を大きく向上させるためには、裏付けのある理論に基づいた開発思想を持って体系的に開発を行うことが重要であるが、十分に理論構築が行われていないのが現実である。本論文は、開発の効率化を可能とする開発プロセスおよび評価モデルを提案しその検証を行ったもので、全編6章からなる。

第１章は緒論である。

第２章では、複数の開発アクティビティを並行に実行する開発プロセスについて、一般的な観点からその特性を論じている。その結果、遅い開発アクティビティが全体の開発時間を大きく左右する特性があることを明らかにしている。これは、開発アクティビティを設計する際の重要な指針を与えていると考えられ、実用上有用な成果である。

第３章では、サービス仕様を部分仕様に分割することによりそれぞれの開発工程を重ね合わせ、開発期間を短縮する畳込開発法を提案している。本手法を用いた開発プロセスは、ウォータフォールモデルに比べて、理論的には50％まで開発期間を短縮可能であり、音声蓄積サービスにおける具体例では、16％開発期間を短縮可能であることを明らかにしている。この成果は、従来の限界を超えてさらに開発期間の短縮を可能にする重要な成果であり、高く評価される。

第４章では、ソフトウェア移植の場合について、プログラム規模に加えてプログラム内に潜在する変換項目数が移植工数に重要な役割を果たすことを解明し、それに基づいたソフトウェア移植工数モデルを提案している。実プロジェクトのデータを用いて本モデルを検証し、従来の手法に比べて見積もり誤差が41％減少することを示している。これは、実用上有用な成果である。

第５章では、プログラム品質を考慮することにより、試験工程の進捗度合いを正確に表現できる試験進捗モデルを提案している。実際の開発データを基に検証を行い、プログラム品質を考慮しない場合に比べて試験進捗率を21％少ない誤差で見積られることを示している。本成果は、試験工程の進捗および工程管理をより的確に行うためのベースとなる理論を与えており、非常に重要な成果である。

第６章は結論である。

以上要するに本論文は、ソフトウェア開発の効率化に貢献する開発プロセスと評価モデルを提案し、ソフトウェア開発効率を飛躍的に向上させる有力な指針を与えたもので、情報基礎科学並びに計算機科学の発展に寄与するところが少なくない。よって、本論文は博士（情報科学）の学位論文として合格と認める。