ソフトウェア工学 77−22
（1991. 2. 8）

大規模ソフトウェア開発環境の研究に求められるもの

上原 三八
大規模ソフトウェア開発環境プロジェクト
(株) 富士通研究所

現実の大規模ソフトウェアシステムに基づいて研究問題を抽出すること無しには、研究成果の適用は期待できない理由を述べる。すなわち事務処理大規模ソフトウェアの開発および保守作業の本質は、巨大なソフトウェアを作り上げることのみならず、常に成長すること、大勢の人間がコミュニケーションしながら行う共同作業をはじめ他の多くのことにもあるからである。さらに、理想的な研究のありかたと現実のシステムの問題点を発見することがどの様に難しいかを述べる。本論文は、事務処理分野を対象とする開発環境の研究のありかたについて議論するものである。

A Requirement for Research of a Large-scale Software Development Environment

Sanya Uehara
Large-scale Software Development Environment Project
FUJITSU LABORATORIES LTD.
Kamikodanaka 1015, Nakahara-ku, Kawasaki, 211 JAPAN

This paper discusses the reason why the result of a software development environment research is not applicable to existing large-scale business application systems, without intensive investigations of the existing systems. The potential problems of developing and enhancing business application systems are associated with not only system construction, but also system's rapid evolution, collaboration among a large number of people, and many others. This paper argues the research requirement of a software development environment for large-scale business application systems, and difficulties in satisfying the requirement.

1 まえがき

本論文は、事務処理分野を対象とする大規模ソフトウェア開発環境の研究が社会的に重要であることに加え、ソフトウェア工学研究の題材を豊富にもつことを指摘した後、実存する大規模ソフトウェアシステムに基づいて研究問題を抽出すること無しには研究成果が期待されない理由を述べる。すなわち事務処理大規模ソフトウェア開発保守作業の本質は、巨大なソフトウェアを作り上げることのみにあるのではなく、常に成長すること、大勢の人間がコミュニケーションしながら行う共同作業をはじめ他の多くのことにもあるからである。とりわけ、急速に社会に応じて変化しなければならないシステムの業務中心性と人間中心性を理解することが重要であり、それが考慮されていない開発方法論や難解な記述言語といった研究結果では適用性が見出せないであろう。最後に、理想的な研究のありかたと実存するシステムの問題点を発見することがどの様に難しいかを述べる。

本論文は、我々研究グループが富士通および大規模ソフトウェア開発企業の協力の下で研究の際に得た知見に基づいて、**事務処理分野を対象とする開発環境の研究のありかたについて議論する**ものである。

2 研究の観点からの対象分野の違い

ソフトウェア開発環境を研究するにあたり、「本当に良い技術はどこでも使える」そして「共通技術こそが研究に値する」と暗黙のうちに認めていないだろうか。本節では2つのポイント、(a) システムの対象分野が異なると研究すべき問題が異なる(言い換えると、共通の技術にはあまり重要なものはないかも知れない!)、(b) 事務処理分野の大規模ソフトウェアはソフトウェア工学の研究対象として多くの題材を提供する、を指摘する。

OSやコンパイラなどの設計法や開発環境については多くの研究がなされているが、数メガステップからなる金融システムなどの大規模事務処理ソフトウェアを対象としたものは多くない。後者と前者を較べるとアルゴリズムは単純だが出現するデータ数が膨大であるということは良くいわれるが、ソフトウェアの性質や開発形態がかなり異なることはあまり良く知られていない。2〜3の例をあげながらどの様な研究課題の差があるかを述べる。

(1) 業務知識の必要性

事務処理分野のシステムを作る上では、仕様獲得フェーズだけでなく、詳細な設計からテストまでの**かなり広い作業フェーズで分野知識が必要である**。実際、詳細な仕様決定は設計フェーズで行われることもある。この説明は[1]にも述べられている。これはOSやコンパイラでは開発当初に仕様や外部インタフェースが比較的明確に定められることに較べて大きな違いである。この事実は開発方法や開発環境のあり方に関して多くのことを意味する。例えば、設計者は業務を熟知した人物が好ましいために、プログラマでなくともシステム開発が可能なように業務の言葉で記述し開発できるしかけが極めて強く望まれる。また、リスクを少なく、不明確な仕様を明確にしながら開発するためのプロトタイピング技術への要請も大きい。

(2) 大規模性

小人数の優秀な人達でOSやコンパイラを作り上げることは良くあるが、事務処理分野では極めて稀である。理由は、業務に加えてシステム設計、プログラミング言語、データベース、データ通信、そしてハードウェア等に関する**多くの知識が必要であること**、またシステムが社会という毎日変化膨張している中に組み込まれるため大小さまざまな**仕様拡張が実に多い**ことである。大量のソースコードやドキュメントをさまざまな役割を持つ**数百人の人間が常時作業する**環境では、レビューや障害対応等において全く異なる形態をもつことになる。また特に社会的役割を担う大規模システムでは、ダウンすることまして情報損失は決して許されない。システムを止めることなくソフトやハードを入れ換えることも求められる。

(3) 拡張性

社会に組み込まれる金融システム等は社会に歩調を合わせ膨張する。そのスピードは大きい。例えば、10メガプログラムステップのシステムが年2割の割合で進化している。そこで、システムテの開発記述を分かり易くしながら少なくする技術や拡張を容易にする技術が求められている。例えば、金融システムでは似た商品に対する処理の追加などがあるため、ソフトウェア・モジュールの部品化、差分開発技術の適用による効果が期待できる。

以上の(1)～(3)からでも分かるように、大規模事務処理ソフトウェアはOSやコンパイラ等の基本ソフトウェアとは研究すべき問題が異なっていることが伺える。中でも、**機械中心でなく人間と業務を中心とすべき点が重要である。**多人数が共同作業する場では人間の間の連絡や承認処理の他、業務の言葉による分かり易い表現が設計・開発の中心となるべきである。いかに開発記述量が少なくともコミュニケーションを阻害するような分かり難い記述言語や計算機(実現)寄りの言葉は採用されることはない。しかし別の見方をすると、多くの計算機や人間が関与することから、**分散、統合、コミュニケーション等に関して多彩な研究問題があること**も伺えよう。ここでは、開発環境とOAの研究題材が豊富である。

3 大規模ソフトウェア研究の重要性

90年代のソフトウェア工学で最も重要な問題は何であろうか。社会生活の中で重要な役割を担うシステム、あるいは企業戦略の中枢となる情報システムを開発する企業が最も欲しい技術は何であろうか。それは数メガステップ級のシステムを高い信頼性と安全性を保証しつつ効率良く開発し、より重要なことは、拡張(enhance)できる技術である。それ以下のサイズであれば、経験的に数名の経験豊かな人達を中心としたプロジェクトチームと時間とお金を投入すれば開発可能であることが分かっている。しかし大規模ソフトウェアの新規開発はもはや人とお金の問題ではなく、可能かどうか、という危機感がある。繰返しになるが注意すべき点は、問題はソフトが巨大というだけではなく、それが急速に社会経済と共に成長し、多人数の人間によって開発し拡張されること、そしてシステムの仕様が商品やサービスの変化、顧客数と地域の拡大、使用するデータベースやコンピュータの拡張に伴い常に変化することである。

80年代に構造化分析法と設計法を広めた中心人物の一人であるEd. Yordonは、その技術は成長型の大規模ソフトウェアの開発には不向きであると認め、スパイラルあるいは成長型の開発保守が可能なオブジェクト指向開発の研究に移行している[2]。スペース・シャトルのシステムは10メガステップ級の大規模システムであるがその生産性は極めて低い。新しい要件を持つシステムの設計には未知なことが多く、完全な仕様が当初与えられないシステムであることを考えると完成したこと自体驚くべきことなのかも知れない。

我われが分析した幾つかの大規模な金融システムでは、開発にあたり共通モジュールの洗い出しやモジュール機能の整理に過去の経験に基づいた工夫を行っており、設計方法論ひとつをとってみてもこれらより確実に優れたものを開発することはチャレンジングなテーマである。例えば、オブジェクト指向開発は90年代のソフトウェア開発およびデータベースのパラダイムとして有望視されているが、我々は大規模ソフトウェアを分析した結果、単にオブジェクト指向を採用するだけでは不十分で、分野別方法論、設計言語や支援ツールに研究すべき問題が多いことを認識している。

4 従来研究について

大規模ソフトウェアを対象とした開発環境を対象とする論文の中で我われが今まで述べたような問題を扱ったものを見るのは稀である。そもそも対象分野や問題が明確に選択されずに2節で述べた様な"共通技術"が研究目標となっていることがある。現状ではプロセスプログラミング、ハイパーテキスト等のキーワードが中心となり、適用できる問題をどこからか切り出してくる、あるいは製造してしまうことがある。そのような問題への

解決策を現実へ適用しようとすると、現実の問題の本質を含んでいないため、多くの障害が生じてしまう。

実際の問題への解を研究した後、それを適用すると新たな問題が見えてくることを我々は経験し
ている。

実際の問題を抽出→研究→研究成果適用
→新たな問題を発見！

この発見された新たな問題が実は求められている技術と関係していることが多い。これを比喩で説明する。簡単な問題を解くことのできるエキスパートシステムを作ったとしよう。これを本格的な問題を扱えるように拡張すると多くの問題が出てくる。この様子は例えば[3]の評価の分析部分が参考となる。また改良はしばしば不可能で最初からやり直さなければならないこともある。最初の単純化された問題は実は表面的な問題であまり技術的なイノベーションを必要としなくとも、後で出てくる問題は極めて複雑で、新しい技術あるいは種々の技術を巧妙に組合わせるといった新しい解決法が必要である。

5 研究のありかた

5．1 理想：現場とのタイアップ

4節までに、大規模ソフトウェアの開発と拡張を対象とした研究を行う上で、実際の問題を扱うことの重要性について議論した。出来れば、**実際の大規模ソフトウェアの開発ドキュメントだけでなく、開発保守の現場を目の当たりにし、現実の問題を分析し抽出することが望ましい。**そして、研究成果である開発技術を適用し現場で評価することでより本質的な問題がクローズアップされ、これを新たな研究目標とすることで革新的な技術の芽となるのではないだろうか。

現場での要求は厳しい。第一に、難しい技術の説明より、実際にどう改善されるかが明確でなければならない。何しろ、何百の人間と何百億ものお金を投入して行っている作業を変革させるための検証の重さは測り知れない。第二に、2節で述べたように、難解な方法論や言語は拒絶される。計算機の専門家からの観点ではなく、業務からの観点から素直な考え方が基本となってなければならない。これは計算機を中心に考えているソフトウェア工学研究者にとってカルチャーショックとなるに違いない。また、新技術の導入に必要な教育期間は短かくて済む方が良い。

5．2 困難への対処

今までの議論に対して、「それなら、何が大規模ソフトウェアの問題か明確にして下さい。それを研究しましょう。」と言いたくなるに違いない。ところが、**問題を捉えることは極めて難しく、それが出来たら半ば問題は解けたといっても過言ではない**こともある。以下にそれが難しい理由を述べる。

まず問題を正確に捉えるためには現行システムを理解しなければならない。それには、どの要求や制約をもとにどう設計したか(上流の設計から、モジュール分け、領域の使い方の他、データベース設計、データ通信などへの多岐にわたる)に始まり、開発保守の体制、ドキュメント管理、障害調査の方法のあらゆることが係わってくる。これらについて熟知しかつ説明できる人物を探すことは難しい。**多くの人にヒアリングして回り、誰がどの情報を持っているかを知ることが必要だ。**

そのような人物の一人に幸運に会えたとしても、「作業のプロセスを教えて下さい。」や「何が問題ですか。」と質問しても必要な答えは返ってこない。一緒にドキュメントやデータベース設計を見て、糸を解きほぐすように理解する努力が必要である。しかし、このような人物は通常忙しいため、かなりの時間に見返りする利益を相手が得られないことには応じてもらえない。

最後に、多分最も困難で研究者の多くが見落とし勝ちな点が、企業の壁である。膨大な金と時間と経験を注ぎ込んで構築した企業のしかも稼働中の中枢システムの内容を無制限に公開することはあり得ない。無数のハードとソフトを応用し、安全性を高めるため工夫している点や、各種のセキ

ュリティ用のプログラムのしかけや作りは流出することは許されない。この障害を乗り越えるには、**信頼関係と双方のgiveとtakeを納得行くものにする**必要がある。

6 まとめ

本論文では、大規模ソフトウェアの開発保守の研究がソフトウェア工学の90年代の研究として重要であること、その研究にあたり現実の大規模システムを知り、そこから問題を把握することが必要不可欠であることを述べた。また、開発企業側から簡単に協力できない理由と、研究機関と企業との相互信頼と相互利益を伴ったタイアップが必要であることを述べた。以上のことは、我々が大規模ソフトウェア開発環境の研究プロジェクトを起こして以来、開発保守の現場を調査し、研究を進めていく上でますます強く感じていることである。

欧米では研究機関と企業との契約が結ばれることが多いが、日本企業も経済摩擦の解消のためもあり欧米の研究機関との交流が今後盛んになると予想される。しかし、日本のソフトウェア工学の分野では残念なことにさまざまな障害が有りこのような兆しは見られない。当研究グループは、日本の大学のソフトウェア工学研究グループにも企業的野心（！）を持って活躍して頂き、優れた研究者を育てながら、社会的要請に応えるべくシステム設計の技術を企業と協力して開発していただければ、と考えている。本稿は技術論文ではないが、そのような議論を活性化するとっかかりになれば幸いである。

謝辞

貴重な時間をさいて多くのことを教えて頂いた各企業のシステム部門の方々、ならびに多くの助言をして頂いた富士通のシステム本部の方々に深く感謝します。また、日頃有益な議論をして頂く園部室長をはじめ富士通研究所の諸兄に感謝します。

参考文献

[1] Charles Rich and Richard C. Waters : Automatic Programming : Myths and Prospects, IEEE Computer, Vol. 21, No. 8, pp. 40−51 (1988).

[2] Peter Coad and Edward Yordon : OBJECT−ORIENTED ANALYSIS, YordonPress (1990).

[3] 上原三八 他：LISP−PAL：プログラミング支援のための自然言語による質問応答システム、情報処理学会論文誌、Vol. 30, No. 11, pp. 1413−1423 (1989).