日本大学大学院総合社会情報研究科紀要　No.5, 254-259 (2004)

ソフトウェア開発トラブルの発生原因と解消策

―IT ベンダーとユーザとの関係改善に向けて―

松村　泰夫
日本大学大学院総合社会情報研究科

To reduce Software-troubles in Project-teams

－Changing in a right direction of mutual relations between customers and IT-vendors－

MATSUMURA Yasuo
Nihon University, Graduate School of Social and Cultural Studies

IT-vendors develop new-systems by customer’s requests. It is necessary for IT-vendors to build up new business application systems with customer’s support. Although deep knowledge of business application areas are possessed by customers, IT-vendors only provide a system design methodology.

To develop new-systems, customers and IT-vendors should corporate for mutual benefits. Goals of project-teams will be completion within the limit of an estimated cost and that of a target date.

This article suggests a solution to reduce troubles of software development.

はじめに

昭和 37(1962)年 4 月、筆者が就職した当時、国産コンピュータメーカ各社は、米国に遅れること約 10 年、ようやく汎用電子計算機の試作を始めたところであった。国産メーカは見込み先に売り込みを図ったが、ユーザニーズを満たすためには、ソフトウェア[1]開発要員を一時的に大量に必要とすることが分かり、その要員を外部から求める方策を採った。

そこにビジネスチャンスが生まれたのである。こうして日本の情報サービス産業は、コンピュータメーカへの要員派遣業としてスタートしたのであった。

当時のメーカはハードウェア（金物）の代金は請求できても、ソフトウェア開発代金は、ユーザに請求することなく、金物の代金に含めてしまっていたので、“ソフトはおまけ”の発想が定着してしまった。

その後、米国の先進メーカが、ハードとソフトの価格分離（方式）[2]を打ち出し、ユーザにも徐々に理解が進み始め、日本の情報サービス産業（IT[3]ベンダー）に、業としての自立意識が芽生えた。

しかし、計画された予算とスケジュール内でシステムが完成した例は現在でも少なく、IT ベンダーのソフトウェア開発部門は赤字経営が続いている。

IT ベンダーが商売としてうまく成行かない原因は、ソフトウェア開発能力やプロジェクト管理能力の未熟さばかりでなく、ユーザの真の協力が得られないからでもある。

ユビキタス時代[4]を迎えた今、IT ベンダーはユーザとの関係改善に向けて歩まないと、健全な発展は望めないし、真の高度情報化社会は訪れない。

本稿は、筆者が定年退職を迎えた平成 14(2002)年 6 月に至るまで、ソフトウェア開発業の中で得た体験から、業界の問題点でもあるソフトウェア開発トラブルの発生原因を分析し、その解消策を提言するものである。

第１章　ソフトウェア開発工程別に見た問題点

ソフトウェア開発業務は、IT ベンダーの主要な業務の一つであり、大勢の PG（プログラマー）、SE（システムエンジニア）、プロジェクト管理者等が従事しており、今後も需要は急増する傾向にある。

IT ベンダーは、発注者であるユーザの開発構想に基づいて、ソフトウェアを構築するのであるが、長い開発期間中には、様々な問題が発生する。

ソフトウェア開発トラブルは、次の７つの段階の中で起きている。各段階の問題点については、著者の修士論文[5]の中でも述べてきた。

（1） 提案~受注の段階

まず最初に、ユーザの経営トップは、予め経営目的を達成するための戦略課題に優先順位を付け、新システムの全体構想を練り、社内のコンピュータ導入委員会に詳細を検討させ、提案書を作成させる。

ユーザには、IT 専任者が不足しているので、IT ベンダーの助けを借りて提案書を作成することになる。提案書には、ユーザを取り巻く経営環境の変化、競業他社のシステム構築状況、ユーザの経営課題、システムの全体構想図、システム化対象範囲、ソフトウェア開発工数見積、システム化予算案、開発スケジュール等が含まれる。

IT ベンダーは、類似システムの開発経験をベースにして、ユーザの理想により近い新システムを、ユーザと協力しながら構想し提案する。

ここで問題となるのは、ユーザが、開発経験に基づくノウハウの結晶である提案書は無償であると考えていることである。IT ベンダー側も、無償で提供しないと受注は取れないと考えている。

体力の無い IT ベンダーにとっては、上級 SE を何ヶ月も対価も無く提供し続けることはできない。

また、システム概要設計の段階でソフトウェア開発工数を見積るので、その精度は低く、この時点で受注（契約）を取れば、IT ベンダーは大きなリスクを負うことになる。システム詳細設計が終了した段階（次に述べる（２）の後半の段階）で受注（契約）するのが本来あるべき姿であるが、競合する IT ベンダーに出し抜かれる恐れから、IT ベンダー側には早い時点での受注獲得の焦りが生じるのである。

しかし、プロジェクトがスタートした後で開発スケジュールが遅延したり、予算規模が大きく膨らむことによる不利益を被るのは、他ならぬユーザ自身なのである。ユーザによる IT ベンダーに対する現状分析支援の重要性が無視されるケースが多い。

（2） 現状分析とシステム概要設計、検討と承認、システム詳細設計の段階

この段階では、プロジェクト管理者、上級 SE、SE、PG 等の大勢のメンバーが業務を分担してプロジェクトチームを編成し、共同作業を行う。開発規模が大きくなれば、１人のプロジェクト管理者が１人１人の担当内容を熟知したり、更にシステム全体の流れを完全に掌握することは困難を極める。業種別・業務別の開発経験をいくら積んでも、分析モレは生じるし、連携ミスも起きるので、完璧な設計仕様書を作ることは困難である。

そこで、プロジェクト会議を何度も開き、設計レビューを行っている。分析モレや設計モレを防止するのが目的で、複数の目で見てもらって、適切であるかどうかを点検するわけである。

概要設計が終了してから改めて詳細設計を行うのであるが、設計担当者が代わることもあり、再び分析モレ、設計モレ等が起き易い。

また、業務を細かく分割して、１人１人の SE に分担させるので、SE 個人の技術レベルの差が成果物に現れるし、詳細設計仕様書を高品質に保つことは難しい。

更に、設計の最終段階では、細かく分割した単体を寄せ集めるので、単体の組立てミスが起きるし、単体同士の相互干渉も起きる。

このような開発手法の特殊性から生じる、“人間の起こすミス[6]”の撲滅は困難であり、コンピュータでテストをしてみて初めてミスに気づくことが多い。

しかしながら、詳細設計仕様書は、家を建てる場合の設計図に相当するので、この段階で再度ユーザの承認印をもらっておく必要がある。当初考えていたシステムは、細かく分析していく内に機能モレが続々と発見され、システム規模が大きく膨らんでくることがよくある。大きく膨らんでも開発するのかどうか、ユーザの意思を確認し、請求根拠になることを理解しておいてもらわないと IT ベンダーとし

ては安心して先には進めない。

（3） 製造（プログラミング）の段階

製造（プログラミング）は、詳細設計仕様書に基づいて行なわれるが、その品質は、１人１人の PG の技術水準次第であり、品質に差が生じる。システム全体として高品質を維持することは並大抵のことでは達成できない。

単体テスト、結合テストを行っても、テストモレは生じるし、テスト仕様書の作成ミスもあり得る。場合によっては、（２）の設計工程にまで逆戻りすることもある。

また、製造（プログラミング）の段階まで来ると、詳細設計仕様書を精読しても気づきにくい分析モレ、設計モレ等が顕在化してくる。新システムで処理できること、できないことがユーザの現場サイドにまで知れわたってくるので、各種の改善要望が出され、詳細設計仕様変更が頻発する事態となってくる。

詳細設計仕様書の不完全性が、この時点で顕在化するが、その原因がユーザ側の説明不足であったとしても、基本的には IT ベンダー側の分析ミス、設計ミスであり、また、文書化技術・プレゼンテーション技術の未熟さが主な原因であるとされてしまう。

この問題は、ユーザとの質疑応答時間が十分に取れていたり、日頃からの緊密な連携が取れていれば、ある程度は解消できるものである。

（4） 総合受入テストの段階

この段階では、テストデータの不完全性が問題となってくる。日常業務処理のテストが優先され、細かい事務処理の中での例外処理事項や危機管理対策処理[7]等にまで、テストの範囲が及び難いからである。

また一方では、日常業務処理が一通り出来た段階で、“ソフトは 95%完成しました”と報告し、ユーザを安心させたい気持ちが IT ベンダー側に起きて、残り５%の作業を後回しにすることが多い。社会問題化する料金計算ミスやシステム障害[8]等は、残り５%の作業量の中のソフトウェアの品質問題である。

IT ベンダーは、開発が予定通りに進捗していれば、この時点で開発代金の最終的な請求書をユーザに提出するが、スムーズな支払いがしてもらえるかどうか緊張しながら待たされることになる。

この段階で不幸にして、システム開発が凍結される場合もある。その原因は、残り５%部分を“後で直します”と返事はしたものの、ユーザ・クレームを調べていく内に、システム設計にまで遡って大改造する必要性が判明したりするからである。

また、ユーザが IT ベンダーの技術力を過信していたり、現状分析支援が十分でなかったり、ユーザ自身の勉強不足等で（３）の製造の段階における単体テスト、結合テスト等の評価が十分でなかった等の原因で開発トラブルが起きることがある。

（5） 移行本番（立会い）の段階

この段階では、旧システムと同じデータを使って、新システムでも同じ結果が出るかどうかを点検する。

この移行本番中に、出てくる問題点としては、処理時間が旧システムと比較すれば遅い、データファイルの容量が不足する、出力帳票が見難い、入力伝票の処理が煩雑だとかの、エンドユーザ[9]からの改善要望も含め、各種のクレームが出てくる。ユーザと共同で対応策に追われるが、改善要望やクレームは、詳細設計仕様書の作成ミスなのか、設計・製造工程中のミスなのか、その判断によっては、再作業のために IT ベンダーが多大な出費を背負うことになる。

IT ベンダーによる移行本番（立会い）期間が終了し、初期トラブルが終息すれば、その後の運用はユーザの責任となる。

（6） 運用・保守の段階

オペレータ教育も終われば、IT ベンダーは総引揚げして、常駐者は居なくなるが、その間に細かな改善要望やシステム変更要求が出てくる。IT ベンダーは、ユーザと定期的な協議会を開くと共に、「保守契約」を結ぶ必要がある。システム上の問題が発生すると、プログラム瑕疵[10]であるとして、際限もなく無償修補を要求されることがあるからである。

（7） 評価・見直しの段階

運用・保守期間が３年も経つと、一部システムの

改廃や情報処理方式の変更等の要望がまとまって出てくることが多い。ユーザを取り巻く経営環境が激変する場合もあり、新機種の開発や新しいビジネスソフトウェアが普及し始めるとユーザの改造要求が高まるからである。どこの世界でも、安い・早い・便利さは絶え間なく追求されるが、評価・見直しの結果によっては一部の改造に留まらず（１）の提案~受注の段階に戻り、次期システムの開発計画がスタートすることが多い。

第２章 業界独特の商習慣、開発手法の特殊性、保守の困難性

以上のことから、IT ベンダーを苦しめているソフトウェア開発トラブルの発生原因を分析すると、IT ベンダー業界独特の商習慣、開発手法の特殊性、保守の困難性が浮かび上がってくる。要約すれば、次の通りである。

①“ ソフトはおまけ ”という考え方が根強いのでユーザに対する“ ノウハウを買う ”考え方の啓蒙活動が必要である。提案段階で、膨大な聞き取り調査・分析・概要設計業務があり、その工数は受注獲得の営業努力と見なされ、請求し難い。

②ユーザ側に、業務の全体像を示す文書が整備されていないので、一つ一つ話を聞きながら文書化していくしかない。その上に、ユーザに SE 並みの文書化技術力を期待するのは無理であり、これをユーザ責任と決め付けることも難しい。

③詳細設計仕様書の記述の不完全性が後になって明らかになるので、設計のやり直しが起きる。詳細設計仕様書を書く技術と（これをユーザが）読んだ上で承認を与える技術力が必要とされるが、ユーザをそこまで教育するのは困難である。

ユーザは、記述されていないことがあっても、“ その位のことは常識として分かっているはずだ ”との思い込みがあるし、一方の SE は“ 書いてないからやらなくて良い ”と解釈してしまう。そこに設計モレが発生する原因が潜んでいる。

④詳細設計仕様変更が、開発の途中で頻発するので、その都度開発作業の手戻りが生ずる。開発工数が増えれば、開発スケジュールがキープできなくなる。開発要員の調達は“ 余人を以って代え難し ”の面があるので、途中からの要員の追加投入が困難である。期待される技術水準と業務経験を持つ SE が常時待機しているわけではない。

⑤工数見積精度が低い。その原因は、不完全な概要設計仕様書に基づく見積であるので、正確な工数が把握できないためである。その結果、予定通りの日程・工数で作業が完了しない。しかし、ユーザは納期遅延にはペナルティを課すことが多いので、IT ベンダーは泥沼に引きずりこまれることになる。開発要員に残業指示を出し、スケジュールに追いつこうとするが、開発期間が長くなれば要員に疲労が蓄積し、生産性は低下し、効率や品質の悪化を招く。こうなると、要員追加申請か開発期間の延長依頼しかないが、ユーザは見積（契約）をたてに開発金額の増額には応じようとはしない。

⑥企業活動と共に情報システムも成長するので、絶えずシステム手直し(更新)が必要となってくる。

以上のことから、納期通りに作業が完了する確固たる自信がない限り、「請負」契約方式は採れない。

詳細設計仕様が固まらず、開発規模が膨らんだり開発期間が延びそうな場合は、「委任」契約（要員派遣契約）の方が IT ベンダーにとってはダメージは少ないし、経営は安定する。

しかし、要員派遣を主業務とする IT ベンダーは、真のソフトウェア開発業とは言い難くなり、理想と現実の食い違いに経営者はその割切り方に悩むことになる。要員派遣事業を表看板にする企業では、学生の採用活動で苦労している現実があるからである。学生自身も就職に当っては業界研究に取り組み、夢と希望の見える企業を選択したいからでもある。

ユビキタス社会は、もうそこまで来ているのに、主役たるべき IT ベンダーに光が見えて来ないのは、どうしたことであろうか。

第３章 ソフトウェア開発トラブル解消策

ソフトウェア開発トラブルの解消に向けて、どの IT ベンダーも一応の地道な努力はしてきたが、短期間で効果が出るものはなかった。解消アイディアは

あっても、抜本的な改善はなされていない。IT ベンダーの企業規模や技術水準に合わせたトラブル解消策を模索しなければならない。

従来から言われている伝統的な解消策の実施がまず第一であるが、ある程度の企業規模があれば、人材面や技術の蓄積により実施できるもの、更にその上のレベルへと段階的に進まざるを得ない。

（1）従来から言われている伝統的な解消策

IT ベンダー業界は基本的には人材次第である面が強い。したがって、極論すれば SE 適性のある人材を集めれば事足りるとの考え方もある。

しかし、人材の発掘・確保・育成に苦労している IT ベンダーは多い。これは、この業界の永遠の課題でもあるが、先ず、人材に関しては、次の５項目を挙げることができる。

①職業適性検査手法の活用により、SE 適性のある人材を発掘する。その後は、適切な人事処遇を考えないと人材は流出すること。

②人材の素質・資質・能力を向上させる訓練コースを企業内で定期的に実施して、育成する。

③資格取得を奨励し、本人のヤル気を促す。報奨金を出し給与やボーナスにも反映させること。

④自己啓発を継続的に促進させる。本人から計画を提出させ、毎年上司がチェックして観察育成記録を残し、本人の意欲を持続させること。

⑤プロジェクト管理を徹底させる。これが最重要課題であるが実際上は困難である。その理由の第一は、作業の進捗状況が建設業のように、目で確認できないからである。成果を可視化する工夫はなされてきたが、プロジェクト管理者は SE 本人からの成果報告を聞き、エビデンス[11]の提供を受けて、それらを評価するしかない。理由の第二は、プロジェクト管理者の育成が需要に追着いていないことである。

その育成のステップとしては、上級 SE にマネジメント教育を施し、実地に大小いくつかのプロジェクト管理をやらせてみることである。しかし、うまくいかなかったら、マネジメント適性がなかったとして SE に戻すしかない。

部下の評判も良く、上司や横の管理職ともうまくやれて、しかもユーザからお呼びが掛かるようなプロジェクト管理者は、多くは居ない。

⑥ユーザから現状分析支援をもらうこと。

⑦ユーザにも設計レビュー会議やプロジェクト会議に出席してもらうこと。

（2）1,000 人以上の規模の IT ベンダーであれば人材面・技術力の蓄積で実施が可能な解消策

解消策の第２弾として、次の４項目がある。

①詳細設計仕様書を書く文書化技術、プレゼンテーション技術の社内標準化を推進する。

ここでの留意点は、“ 知った ”、“ 分かった ”、“ 身に付いた ” の三段階の内、今どの理解レベルにあるのかを SE 自身に認識させ、意図的に次のレベルへ一歩を進めさせることである。

②図解技法の活用によりドキュメントの可視化努力を徹底する。視覚に訴えた方が理解が早い。

③ドキュメント用紙の色分け区分により、バージョン管理を徹底する。机上を見れば、担当者毎に作業の進展状況が一目で分かる。 例：白（初版~改版）、黄（最終版）、青（納入物件）。

④「詳細設計仕様変更審査会」の開催。仕様変更の内容をユーザ立会いのもとで、プロジェクト内でオーソライズする。担当者名、変更内容、工数見積、優先度、他システムに与える影響、納期等の開発情報を明記することにより、プロジェクトチーム内での情報の共有化を図る。

用紙の色分け区分　例：赤（詳細設計仕様変更）。

（3）先進的な IT ベンダー向けの解消策

この解消策は、ユーザに余程の理解がない限り実現は難しいし、IT ベンダーにも相当な覚悟が必要である。要約すれば次の３点となる。

（3－1）開発契約方式の見直しを図る

従来からの「一括請負契約方式」を改め、契約を開発工程に合わせて大きく二分割する。

①上流工程（提案~受注、現状分析とシステム概要設計、検討と承認、システムの詳細設計）については、工数が予測し難いので共同研究開発契約（「委任」契約）とする。

②下流工程（詳細設計仕様書が完成し、製造（プ

ログラミング)、総合受入テスト、移行本番(立会い)）については、「請負」契約とする。

この共同研究開発契約方式のメリットとデメリットは次の通りである。

★メリット

①詳細設計仕様書の完成度が高まる。
②分析ミス、設計ミスの事前防止につながる。
③工数見積精度が高まる。
④開発スケジュールの変動が少なくなる。
⑤開発コストの縮減につながる。
・総発注金額の内、ユーザのシステム担当者の工数金額分は減額できるからである。
⑥ユーザとの協力体制が一層強化される。
・IT ベンダーの常駐している一室に、ユーザのシステム担当者も常駐するのが条件である。
・打合せが緊密になり、コミュニケーションが円滑になる。
・開発情報の共有化に寄与できる。
・詳細設計仕様変更の件数が減少する。
・将来的にはユーザの SE 育成に貢献できる。

★デメリット

①ユーザにもシステム開発責任が生じる。
②従来の「丸投げ方式」は取れなくなる。
③IT ベンダーの受注金額の総額が減る。

（3－2）ユーザとの役割分担を明確にし意識改革に取組む

ユーザはあくまでも業務の専門家であり、どんなに頑張ってもユーザの知識量を超えることは出来ない。一方の IT ベンダーは、システム化の専門家であるので、両専門家が力を合わせて新システムを構築するのが望ましい姿である。しかし、現状はユーザからの「丸投げ方式」が多い。また、ユーザに戦略情報化企画部門が組織化されているケースは少ないし、CIO[12]すら居ないので、その辺の意識改革が必要である。

（3－3）外部のシステム監査業者を導入する

システム監査を行う外部の業者を導入することにより、ユーザと IT ベンダーから独立した立場でソフトウェア開発プロジェクトの全工程を監査させるものである。建設業でいえば、設計監理企業に相当するが、IT ベンダー業界では実績はなく、その発想すらない。類似した形態として、ユーザが IT ベンダーとの間に経営コンサルタントをはさむケースはあったが、その身分は税理士であったり公認会計士であったりした。ユーザの経営トップの推薦で割り込んできたのが実態で、ソフトウェア開発工程の知識もなく、プロジェクト管理の素養すら具備していないため、プロジェクト会議に出席しても、その役割を果たせなかった。

日本ではまだ、有償でシステム監査業者を導入する動きはない。先ずは、システム監査能力を具備した後で実績を示し、ユーザに評価されない限り、定着はしないであろう。しかし、近い将来にシステム監査企業が出現することを期待したい。

---

[1] ソフトウェア；コンピュータのプログラムや運用技術の総称。実際に目で見る形は文書のみである。
[2] 価格分離（方式）；Unbundling 方式は IBM 社によって 1969 年 6 月から始められた。
[3] IT；Information Technology の略。情報通信技術と訳す。
[4] Ubiquitous 時代；いつでも、どこでも、誰とでもネットワークに繋がる時代。
[5] 修士論文；松村泰夫『IT 革命時代におけるコンピュータ関連トラブルの現状と対策』日本大学大学院総合社会情報研究科、2003 年 3 月、参照。
[6] 人間の起こすミス；ソフトウェア開発業が抱える悩みと同様に、世間には多種多様な人為ミスが起きているが、その防止対策には、何か共通したものがあるに違いない。人為ミスの例として、医療過誤、交通機関の運転・操縦ミス、管制指示ミス、大学等の入試問題出題ミス、採点集計ミス、金融機関のシステム障害、官公庁のデータ処理モレ・金額計算ミス・重複請求、個人情報の紛失・漏洩事件等が起きている。
[7] 危機管理対策処理；操作ミス防止対策、過負荷異常対策、通信回線異常対策、ウィルス対策、電源・空調異常対策、災害対策等の通常では起こりえない異常時の対策。
[8] システム障害；2002 年 4 月 1 日、みずほ銀行で障害が発生した。原因はプログラムミスと発表された。
[9] エンドユーザ；ここでは、ユーザの情報処理部門から提供されるコンピュータ出力結果を利用する現場部門のこと。
[10] プログラム瑕疵；IT ベンダーの責に帰する設計仕様書とプログラムの不一致については、IT ベンダー側の瑕疵担保責任期間内であれば、瑕疵修補責任があるとされている。
[11] エビデンス；Evidence 証拠書類、文書（ドキュメント）を中心とするソフトウェア成果物。
[12] CIO；Chief Information Officer の略。情報戦略統括役員。

（Received:December 31, 2004)
（Issued in internet Edition:January 31, 2005)