# 連載　企業および社会における情報システムの意味を考える
# 第 9 回　アプリケーションの保守（Maintenance）を考える

大島　正善（MBC:Method Based Consulting）

前回の最後に、今月は、開発工程モデルについて書くとしましたが、それは次回にし今回は保守について書くことにします。

## １．保守とは維持改善のこと

システム開発が終わり本番稼動を迎えた後の活動のことを“保守”と呼んでいます。稼働後の作業である“保守工程”は、開発工程とは別のものと考えられています。ライフサイクル・モデルの国際標準でも然りです。国際標準の定義でも、保守工程の主たる活動は、“問題対応”が中心[*1]という位置付けで書かれています。しかしながら、最近、開発と保守の垣根を取り払うDevOps[*2]といわれる“開発手法”が話題を呼んでいます。このことはどのような意義があるのでしょうか? 単なる技術論にすぎないのでしょうか? 今回は、保守とは何かという観点から書いてみます。

保守は、英語のmaintenanceを日本語に翻訳したものですが、「保守」という訳語よりも、「維持」のほうが適切に思えます。英語のMaintenanceを英英辞典で調べると、次のような意味であると書かれています。（ロングマン現代アメリカ英語辞典より）

- The repairs, painting etc. that are necessary to keep something in good conditions

それに対して保守という意味に近いconservativeという英語の意味は次のようなものです。

- Preferring to continue doing things the way they are being done or have been proven to work, rather than risking changes

これらを読んでもわかるように、maintenanceには、「よい状態を保つ」という意味があり、conservativeのように「リスクを恐れて変化しない」という意味ではありません。日本語の保守という言葉は、それこそ「後ろを向いて守る」というイメージを持たせますが、「維持」には、前向きでそのあとに「改善」という言葉が隠されているように思います。しかしながら、稼働後の情報システムの作業を保守工程と名付けたがために、いったん作ったシステムをできるかぎり変えずに置くことが保守だと誤解されてしまった面があるように思うのは私だけでしょうか。
Maintenanceを保守と訳したのは誤訳だと思います。

日本語の持つイメージから保守という仕事はあまりモラルの高い仕事ではないと思われています。たとえば、「情報システムの予算のうち７０％以上が保守に費やされている」というとき、保守に投資するのではなく、新規開発にもっと多くの予算が使われるべきだ、ということが暗黙のうちに語られています。

しかしながら、２０年前とは違い、現在の多くの企業では基幹業務のシステム化は一通り済んでおり、必要なのは既存システムの改善であることが多いのは当然です。もちろん、スマホやタブレット対応、情報の分析と活用などの新規開発案件もありますが、従来から存在する基幹業務システムが不要になるわけではありませんし、ビジネス環境の変化に対応し続けることが必要です。したがって、保守は減ることはないと理解すべきでしょう。つまり問題なのは保守の量が多いということではなく、工数（コスト）が割かれていることと重要な仕事にもかかわらず担当する人たちが意義を見いだせないという心理的な面にあるように思います。

一般に企業におけるＩＴ系の仕事では、新しい技術の適用にチャレンジすることが求められ、そういった仕事を担当し成果を上げると高い評価を得られるという風土があります。従来の技術を扱う仕事は敬遠されがちであり、しかも他の人が開発したシステムの保守という仕事は、モチベーションがあがらないのはある意味理解できる点であります。

しかしながら、Maintenanceという仕事を重要であると位置づけている企業があります。たとえば、東京ディズニーリゾート（以下ＴＤＲと表記）を運営しているオリエンタルランド社がそうです。ご存じのように、ＴＤＲは、東京ディズニーランド開園後３０年がたちました。新しいアトラクションは生まれていますが、施設の多くは基本的に開園当時のまま綺麗に維持されています。このような娯楽施設は、開園後数年は景気がよいものの、数年たつと人気はガタ落ちというのが相場ですが、ＴＤＲが長期にわたって人気を保っているのは、まさにメンテナンス投資を怠らず各種施設が提供する顧客サービスの維持向上を大事にしている結果でしょう。

自動車、家電、PC・携帯電話などの情報機器、洗剤・化粧品などの日用品など、ほとんどの製商品はモデル・チェンジがあります。市場の売れ行きや評価のフィードバックを取り入れて、常に改善を行っています。これは、イノベーションによる新製品開発のようですが、実際の多くの製商品は一度作った製品をもとに、それに手を加えて改善して作り上げたものです。それは、ある意味メンテナンス作業といえます。

また、“カイゼン”を浸透させているトヨタを持ち出すまでもなく、一般に製・商品を作り出している企業は、一度開発した製品を改善しながら新製品を生み出して

います。

企業の情報システムの保守（maintenance）も、このようなカイゼン工程だととらえることが可能なのではないでしょうか。

メンテナンス工程は本来、維持改善工程ともいうべき期間であり、利用へのサービス・レベルを維持あるいは向上させることが狙いの期間です。情報システムの開発は歴史が浅いために、長い間、新たに作ることが目的となっていました。しかし、今では、最低限の業務システムは出来上がっています。今は、考え方をあらため、稼働しているシステムの品質をいかに高めていくのかということも重要目標の一つに挙げる時期に来ているのではないでしょうか。情報システムが業務処理を行うということは、業務の継続的改善は、とりもなおさず情報システムの継続的改善ということになります。メンテナンスという作業を過少評価してはいけないでしょう。

## ２．開発と保守の意味づけを変える必要がある

メンテナンスという作業をいったん開発し終えたシステムをそのまま継続して使い続けると考えるのではなく、維持改善を行う期間だと考えると、ベンダーが参加するシステム開発プロジェクトにおけるいくつかの重大な問題の解決の糸口がみつかるように思います。たとえば、要件の量のコントロールと要件変更のコントロールです。この二つの点は、システム開発プロジェクトにおいて解決できない大きな障害となっています。ベンダーが開発に参加するプロジェクトでは、ベンダーにとっては赤字プロジェクトになるかならないかという死活問題にもかかわるからです。

当初の想定よりも要件が増えて予算の範囲内に収まらない可能性が出てくることはまれではありません。そのために、プロジェクトでは、多くの関係者間の調整作業が行われ、多くの時間とお金が費やされます。要件の変更にいたっては、それが、変更なのか新たな要件なのかという判断の違いもあって、なかなか決着がつかずに時間とお金が湯水のように使われてしまうことがあります。しかも、まずいことに、そのような調整と交渉は、プロジェクト責任者やリーダー・クラスの単価の高い要員が関与するのが通常であるにもかかわらず、その交渉にどの程度の予算が使われたのかを管理している例はまずありません。

そういうことになる原因は、開発段階において、当初開発するシステムにできるだけ多くの要件と取り込んで稼働させたいと思う発注側と、プロジェクト・コストの管理が重要なベンダーに基本的姿勢の違いがあるからです。もし、稼働と同時にメンテナンスを始めるという考え方が浸透すれば、最初の稼働時期をできるだけ早

め、引き続いて追加要件を反映していけばよい、という考え方になるでしょう。それにより、最初の稼働時にどの要件を反映させるべきかという判断が容易になると思います。コスト面の効果も大きく、現状のように無駄に時間とお金を費やすという愚を避けることができるでしょう。

アジャイル方式は技術論だけではなく、こういった開発プロセスの意味への問いかけを含んでいるようです。マイクロソフト社が自社製品をアジャイル方式で開発しているだろうことは想像に難くありません。バージョン1の後、少しずつ改善して機能を追加したり使いやすさを向上させたりできるからです。投資に対する開発コストの管理も容易です。決めたことを請け負ったから無理をしてもすべて開発するというやり方をせずにすむでしょう。

そういう観点からアジャイル手法の考え方を取り入れたDevOpsという開発手法をとらえてみると、単なる手法という以上に本質的な問いかけがあるように思います。ソフトウェアの開発をベンダーに任せず自社で行う企業がアジャイル方式を採用し、開発と保守を一体化させる体制とするのは理屈に合っています。ベンダーにソフトウェア開発を依頼している場合でもこの考え方のほうが妥当であると思えば、DevOpsの仕組みを採用するための準備（組織体制の確立し、人材育成や設計・開発ツールなどの導入など）を真剣に考えるべきでしょう。ベンダーとの契約含めた関係の見直しも大きな課題であることも間違いありません。

## ３．継続的なシステムの維持改善を考慮した要件管理と契約モデル

ソフトウェアの開発があるシステム開発プロジェクトでの大きな課題は二つあります。一つは、開発規模の見積もり精度を低いということと、要件の変更が多いということです。見積もりの精度が低いという点については、今回のテーマではないので別にしたいと思います。

要件の変更にどう対応するのかということは、プロジェクト管理の重要な要素で、この問題に適切に対応できれば失敗は少なくなるのは間違いありません。

システム開発時にどういった要件を取り込み、後回しにしてよいのはどういうことなのかということを前面に押し出してベンダーが作業を実施するのは、かなり困難が伴います。それができるためには、開発プロジェクトの発注サイドのオーナーが最初にそのことを宣言しておくことが絶対の条件です。また、予算のコントロールに関しても、最初から予算すべてを使うことを前提に要員計画を立てますが、考えてみればおかしな話です。貯金を残すことを目指して予算計画をたてるべきでし

ょう。

予算は使うものであるという日本の悪しき伝統から抜け出せないからでもあります。多くの予算を使うプロジェクトが高く評価されるという、これも、悪しき伝統があります。「予算は残すものである」という発想をし、いかに多くの予算を使い切らずに残すことができたかということを評価尺度にすればよいのです。私もベンダーの立場での経験がありますが、ある大規模システムの全面改修案件でそのような計画をたてようとしたときに、「そうであれば開発予算は少なくてよいですね」と詰め寄られたものです。でも、そういう発想は間違いです。

それは、契約方式にも問題があるからです。支援か請負という形態しかありません。予算が使い切らずに完成させると評価する契約にすべきでしょう。余った予算の一定割合を開発企業に支払う契約にすればよいのです。どの程度の割合にするかは、個別に決めればよいだけです。難しい話ではありません。

また、開発とその後の維持改善を同じ土俵で議論することは、どういった要件を最初に取り込むのかという判断に余計な時間を使うことは少なくなるのは間違いないところです。そうすれば、要件の重要性、緊急性、重大性などを短時間で評価し、迅速な意思決定ができるようになるでしょう。

日本のシステム・インテグレーション・ビジネス（ＳＩ）とアウトソーシング・ビジネス（ＳＯ）は大きな転換点にあります。発注サイドにも問題はありますし、受注側にも問題があります。日本の慣行となっている現在の契約形態では、生産性を高めるというモチベーションは決して上がりません。何年も前から言われていますが、相変わらず労働集約的なやり方でシステム開発を行っている方が収益があがるからです。経済産業省が「情報システム・モデル取引・契約書」を発行しています[*3]が、それも抜本的な見直しが必要です。予算を使い切らずに開発を完了させたらインセンティブを出すという契約にすれば、ベンダーは、いかにして生産性を上げるかということを真剣に考えるでしょう。

情報システムの運用が業務の運用そのものになっている時代に、保守という仕事を軽視するのは時代遅れになっているように思います。ビジネスの変化に迅速に対応するのは当然であり、業務改善はすなわち情報システムの改善であるという時代です。今までのような、「情報システムに問題が出たら対応する」という発想ではなく、前向きに維持改善するという発想が必要でしょう。

開発方法論も、生産性を高め早期に開発を終了し、その後連続的に維持・改善をしていく開発・保守モデルを打ち立てることが求められています。それは、ソフト

ウェア開発は約５０年の歴史を経て、ほとんどの企業や行政組織に第一段階の情報システムが導入された今日、開発方法論のイノベーションが求められているということでもあるでしょう。

今回は、ここまでとします。

[参考]

[*1] ソフトウェア保守の国際標準である ISO/IEC 14764 でも、トラブル対応プロセスが最初に記載されています。
(http://homes.ieu.edu.tr/~kkurtel/Documents/IEEE%20Std%2014764-2006%20Software%20Life%20Cycle%20Processes%20%E2%80%94%20Maintenance.pdf)

[*2] DevOps：DevOps (a portmanteau of development and operations) is a software development method that stresses communication, collaboration and integration between software developers and information technology (IT) professionals. DevOps is a response to the interdependence of software development and IT operations. It aims to help an organization rapidly produce software products and services.
(http://en.wikipedia.org/wiki/DevOps)

[*3] 情報システム・モデル取引・契約書
この資料の p2 に目的が書かれている。そこでは、役割分担と取引の可視化などが目的とされていますが、変化の速い時代の要件の変更への対応と品質確保の観点からは何の考慮もされていないことがわかります。ベンダーが瑕疵を負わないことを重点にした契約書のガイドとなっており、発注企業とベンダーが Win-Win の関係を築くためには発想の転換が必要です。
(http://www.meti.go.jp/policy/it_policy/keiyaku/model_keiyakusyo.pdf)

以上