プロジェクトの状況にあわせたプロセスの調整

プロセスの現状と日本ユニシスに見る実際

尾島 良司
OJIMA, Ryoji
日本ユニシス株式会社
.NETビジネスディベロプメント

Technology Tools

- ❑ Visual Basic .NET
- ❑ Visual C# .NET
- ❑ SQL Server 2000
- ❑ Oracle 9i
- ❑ Access 2002
- ❑ ASP.NET
- ❑ Internet Information Services
- ❑ Other:

Level

★ ★ ★ ★ ★

Samples

はじめに

「プロセスなんかいらないよ。無駄なドキュメントを強要されて生産性が落ちるだけだもの」

プログラマ代表の声が聞こえてきました。ごもっとも。私も、プログラマの人権を無視した無茶苦茶なプロセスの下に、どう考えても誰も読まないドキュメントを徹夜で書かされたことがあります。あの日会議室の椅子を並べてとった仮眠で、悔し涙で椅子を濡らしながら「私は絶対に偉くなってプロセスのない世界を作る」と誓いました。

……えーっと、現在バリバリの平社員で、偉くなれる見込みもゼロだったりします。毎日が肉体労働天国で、プロセスのない世界は遥か彼方でまったく見えません。

でも、プログラマを苦しめた既存プロセスの問題点が明確になり、実際に役に立つプロセスが出てきたので、私が偉くならなくても（なれなくても）もう大丈夫。冷静に考えてみれば、悪いのは“既存の”プロセスであってプロセス“そのもの”ではないわけですし。今の私は、「適切なプロセスは我々プログラマにとっても有効」だと考えています。

というわけで、本稿では、既存のプロセスが拠りどころとしているソフトウェア工学の問題について考察し、ソフトウェア工学以外の解決策を模索し、日本ユニシス株式会社（以下日本ユニシス）の解答である開発方法「LUCINA for .NET」のプロセス部分の概要を説明します。

ソフトウェア工学という幻想

最初に、IEEEによるソフトウェア工学の定義を。

> ソフトウェア工学とは、ソフトウェアを開発／運用／保守するために適用される、体系的かつ規則化された定量的アプローチのこと。すなわち、ソフトウェアに対して工学を適用することである

実にもっともらしく聞こえます。そして、ソフトウェア工学が成果を上げた事例も豊富に存在しているのです。

たとえば、1969年から1975年にかけて開発されたSAFEGUARD弾道ミサイル防衛プロジェクト。64,484人月の大規模プロジェクトで、低水準言語のコードを人月あたり34.8ステップ作成しました。

たとえば、NASAのスペースシャトルのソフトウェア開発。年あたりの予算は3,500万ドル（1$＝120円で42億円）でソースコードは420,000行。直近11回の改訂で発見された欠陥はわずか17個でした。ちなみに、この規模だと通常は5,000個程度の欠陥が出ると言われています。

「ちょっと待て。数字がおかしくないか？」

おっしゃる通り。これらのプロジェクトでの成功の定義は「安全なシステムを構築することができた」というものなのです。数万人月とか年あたり数十億円とか、こんな常識はずれの数字が並んでいることは問題とはしていません。つまり、期間やコストなどのリソースは無制限という前提を立てて、安全性を追求しているのです。

人命がかかわっている軍や政府のプロジェクトであれば、この前提は正しいでしょう。そもそも税金使い放題なわけですし。しかし、商用目的のソフトウェアではこんな前提は通用しません。ビジネスで利益を上げるためにソフトウェアを作るわけで、利益とはつまるところプロフィットからコストを引いたものなのですから、コストに直結するリソースを無制限につぎ込める場合など存在しないのです。

整理しましょう。ソフトウェア工学では、以下のトレードオフが必要になります。

- 十分な品質を得るには、膨大なリソースが必要
- リソースを抑えると、品質は大きく下がってしまう

商用アプリケーション開発にソフトウェア工学を適用する場合は、このトレードオフを調整しなければなりません。しかし、調整の結果は、ちょっとは安くなったけどやっぱり高額で低品質なアプリケーションとなってしまいます。ソフトウェア工学を採用するなら、リソースを制限するわけにはゆかないのです。

「つまり、ソフトウェア工学は商用アプリケーション開発には適用できないの？」

おっしゃる通り。ソフトウェア工学は一部の洒落にならないプロジェクトでは有効でしょうが、それ以外のプロジェクトでは意味を持たないと考えます。軍隊のやり方を民間にそのまま持ってくるのは無理があります。そもそも、ソフトウェア工学という単語は1967年にNATO（北大西洋条約機構）の研究会で発明されたものです。生まれも育ちも軍隊。会社に来たら隣の席にランボーが座っていて、銃口を突きつけられて仕事をするなんて嫌でしょ？

「それだけ問題があるということは、出発点が間違っているのでは？」

おっしゃる通り。一般の工業とソフトウェア開発は大きく異なるので、工学を単純に適用することは難しいと考えます。

一般の工業の“製造”に相当するソフトウェア開発の工程は“ビルド”なのです。工場のベルトコンベアと同じやり方が通用するのは、「Visual Studio .NETで［ビルド］-［ソリューションのビルド］を選択してしばし待つ」という部分だけなのです。

一般の工業では製造の自動化を推し進めています。しかし、ビルドを自動化するなどというのはソフトウェア開発では当たり前の話。一般の工業と比較してソフトウェア開発は非常に進んでいると結論できます。よって、通常のプロジェクトに工学を追加する必要は少ないと考えます。

そう、ソフトウェア開発という作業は、工場の製造ラインの設計に相当するのです。製造ラインの設計は単純労働では不可能ですよね？ ソフトウェア開発も同じです。ソフトウェア開発をベルトコンベア式の誰でもできる単純作業に分解することはできないのです。

アジャイルという反論

ソフトウェア工学が忘れていること、それは我々プログラマが人であるということです。

プログラマなら誰でも知っているように、優れたプログラマはそうでない