一般論文

産業用計算機システムにおける新たな開発プロセスの適用

トータルソリューション事業所　技術第2部　工業第2課
藤原 宏文

1.まえがき

三菱電機(株)と当所は、1997年に産業用計算機向けにソフトウェア開発支援ツールであるEZprogramming(注1)を共同開発し、1998年から2005年にかけて、同ツールを適用したシステムをユーザーに納めてきた。

産業用汎用計算機は、ハードウェアのダウンサイジングが進み、汎用計算機ではなくワークステーション、パーソナルコンピューターが主流となり、ソフトウェア開発においてもワークステーション、パーソナルコンピューターが提供するプラットフォームで開発が行なわれるようになってきた。さらに近年では、産業システム向けに開発された専用ソフトウェアである「産業用ソフトウェア」を使用してシステムを開発することが標準的となっている。

EZprogrammingの開発から約20年が経った現在、同ツールを適用して開発したシステムは、ハードウェアの老朽化・生産中止を迎え、システム更新案件の受注が増加してきた。しかしながら、システム更新においては、開発プラットフォームの違いにより、既設システムからのアプリケーションの移植に多くの労力を費やしていた。

このような背景からEZprogrammingを用いて構築した既設システムから、産業用ソフトウェアを用いたシステムへの更新案件をターゲットとして、高効率且つ標準的に利用可能な開発プロセスを確立する活動に着手した。

本稿では、この活動概要について述べる。

2．既設システム更新における課題

2.1　既設システムの特徴

EZprogrammingを開発した背景は、計算機の汎用化と共に、ユーザーのシステム開発費用抑制、かつ高品質なアプリケーションを短期間で納入するニーズが高まってきたためである。

このようなユーザーニーズへの対応策として、アプリケーションの流用を推進するために、Ezprogrammingを開発した。

EZprogramimingを用いたシステムの概略構成は、図1のとおりである。

図１．EZprogramming概略システム構成

EZprogrammingの特徴を以下に述べる。

(1)プラットフォームに依存しない関数の提供

プラットフォームの違いにより提供されるミドルウェアが異なっても、EZprogrammingの提供する関数が、プラットフォームの差異を吸収する。

(2)外部通信手段に依存しない関数の提供

定義ファイルの設定項目の選択によって通信手段の切り替えが可能であり、アプリケーションは、外部入出力方法の違いを意識しない。

(3)ソフトウェア仕様書から関数を自動生成

一部のアプリケーションは、ソフトウェア設計仕様書から自動生成できる。

2.2　産業用ソフトウェア適応時の問題点

2003年以降 EZprogrammingに替わって産業用ソフトウェアが導入され、アプリケーションの記述言語も、ANSI準拠のC言語から専用のスクリプト言語に改められた。これにより発生した問題点および背景は次の通りである。

(1)プラットフォーム相違に関する問題

産業用ソフトウェアの導入により新規にアプリケーションを製作する場合の効率が飛躍的に向上した反面、既設のシステムを更新する場合はスクリプト言語で再開発を要するというデメリットも持ち合わせていた。そのため、既設システムの更新を行う案件では、ユーザーから短納期かつ更新後、数時間以内に既設同様の製品品質の精度を求められており、新たに専用のスクリプト言語で再開発するには工期も厳しく、かつリスクも大きかった。

(注1) アプリケーションの標準化とミドルウェア改善計画に基づき、ソフトウェアの製作工期の短縮化を目的としたソフトウェア開発支援ツール。
(注2) ソフトウェアが情報を互いにやりとりするためのインターフェースの仕様。

(2)実績収集機能のインターフェース相違に関する問題

システム更新においては、計算機とともに制御コントローラーも同時更新する場合が多く、制御ネットワークも専用プロトコルからEGD(Ethernet Global Data)(注3)プロトコルに変更される。既設システムでは、TCP/IPで構築した専用プロトコルを用いており、大量のデータを高速で処理する必要がある場合は、制御コントローラーが高速でデータを蓄積し、計算機は制御コントローラーから送信された蓄積データを長期間保存する仕組みが実績収集機能の主流であった(図2参照)。しかし、近年では、計算機に用いるハードウェアの高速化およびハードウェアリソースが潤沢になり、長期間保存を行うデータを計算機が直接収集する機能構成となった。そのため実績収集機能は、膨大な項目のデータを扱うため、新システム向けにアプリケーションを改造するには改造量が多く工期および、移植後の品質確保にリスクがあった。

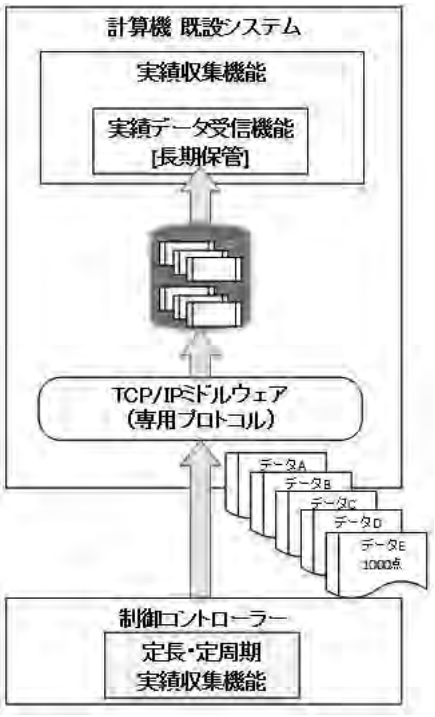

図2．定長・定周期実績収集（既設システム）

3．開発プロセスの適用

プラットフォーム、実績収集機能のインターフェース差異を吸収することで既存アプリケーションを活用し、効率的に開発できるプロセスに見直し、適用した。

3.1　プラットフォームの相違への対応

(1)経緯

産業用ソフトウェアを適用したプラットフォーム上で更新する場合、アーキテクチャや使用上の作法が異なる。

特に、既設アプリケーションは手続き型プログラミングであるのに対して、産業用ソフトウェアはオブジェクト指向で作成されており、EZprogrammingへの組み込みは関数の差分開発では既設の機能を踏襲できなかった。

そこで、新プラットフォーム+産業用ソフトウェアのシステム上で、EZprogrammingで稼働していた手続き型プログラミング言語の既設アプリケーションを活用できる仕組みを検討した結果、Ezprogrammingと産業用ソフトウエアとのインターフェースを吸収できるAPIの開発が必要と判断した(図3参照)。

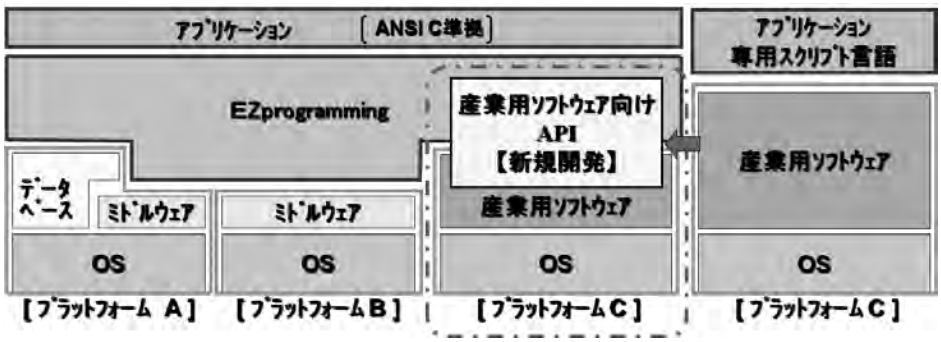

図3．ソフトウェア構成イメージ

(注3) 制御コントローラーの内部メモリーをイーサネット経由で通信し、グローバルメモリーのように読み書きできる情報。

(2)課題

図4にEZprogrammingを適用した既設システムの手続き言語と、産業用ソフトウェアを適用した新システムのオブジェクト指向言語でのデータベースアクセスの相違を示す。

EZprogrammingを適用したプラットフォーム間で改造･更新する場合、汎用計算機に付属している手続き型言語用のミドルウェアを使用していたため、仕組みは類似しており大幅な修正にはならないが、産業用ソフトウェアはオブジェクト指向であり、相互に起動を行うオブジェクトの集まりであるため、アプリケーションは処理要求と処理結果受け取りを別処理とする必要がある。また、この仕組みの場合、マルチCPUシステムでは、結果を受け取るまでの間に別の処理要求を受け付け、処理の順番が変わるため、既設システムと同様に処理の途中で他の処理の割り込みを抑止する必要があった。

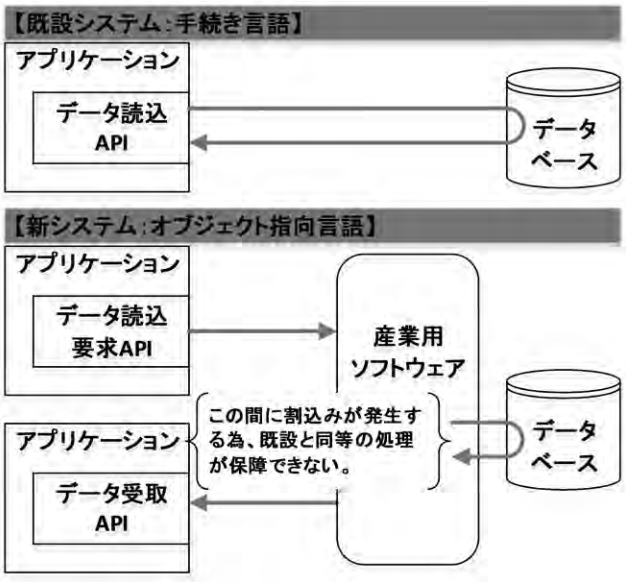

図４．データベースアクセスの相違

(3)解決策

図5に産業用ソフトウェア適用環境下における新規開発APIによるデータベースアクセスの例を示す。

新規開発APIでは、まず既設と産業用ソフトウェアが提供する関数の差分を抽出し、提供されていない制御ネットワークのアクセス関数を追加開発した。次に開発において課題となっていた処理の途中で割り込みを発生させないようにデータ読み込み要求を行ったアプリケーションに結果が返ってくるまで処理を待つと共に、C言語の既設アプリケーションから呼び出し可能な関数を新規に開発した。

最後にEZprogrammingの提供する関数群に産業用ソフトウェアが提供する関数および、追加開発した関数群を組み込むことで、既設アプリケーションを修正することなく流用することが実現できた。

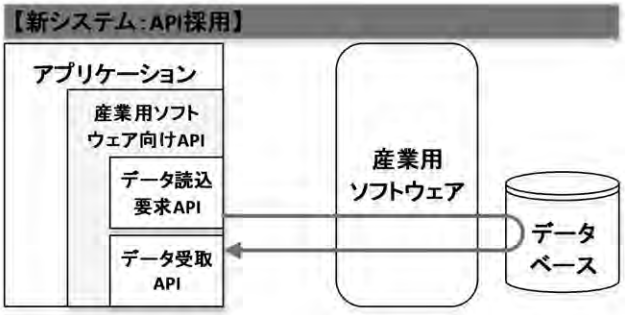

図５．新規開発API 採用時のデータベースアクセス

3.2　実績収集機能のインターフェース相違への対応

(1)経緯

プラットフォームの相違を吸収するため、前節のAPI開発を行ったが、計算機と制御コントローラーのインターフェースが既設と異なることから、API開発だけでは実績収集機能の既設アプリケーションを活用することができなかった。実績収集機能は、一項目あたりの収集点数が1000点を越えるデータを扱っており、また、収集項目が多くアプリケーションを移植するには、工期、品質にリスクがあることから既設アプリケーションを流用できるようにする必要があった。また、今後の更新にも使用できるように汎用性を持たせるための開発も必要と判断した。

(2)課題

実績収集機能の開発における課題は、既設の制御コントローラーが行っていた処理を最小限の開発量で計算機のアプリケーションに組み込むことであった。

更に、データの保持に関しては、産業用ソフトウェアがオブジェクト指向で設計されているため、アプリケーションが自発的にコモンメモリーを使用する機能を開放していない。しかし、既設の機能を実現するためには、複数のアプリケーションから共有データを参照する必要があった。

既設のアプリケーションを変更せず、既設の制御コントローラーの実績収集機能を計算機で実現するために産業用ソフトウェアが提供しているEGD通信サービス、状態変化監視サービス及び、データ収集サービスを使用し既設の実績収集機能を実現する必要があった。

(3)解決策

機能を実現するにあたり、EGD通信サービスが制御ネットワーク上の情報を取込み(図6①)、データ収集サー

ビスがメモリー上にデータを管理し(図6②)、新規追加したアプリケーションがデータ収集サービスから情報を受け取る(図6③)仕組みとし、産業用ソフトウェアが標準提供するサービスで制御コントローラーが行っていた処理を代用したことで、アプリケーションの新規開発を最小限にできた。また、データ収集サービスがメモリー上で情報を管理するため、数百項目×1000点のデータも数百msecでの処理を実現しており、システム全体の処理速度に影響を与えることはなかった。

次に、収集したデータは、状態変化監視サービスでタイミングを監視し(図6④)、定長･定周期実績収集機能が新規開発APIを経由(図6⑤、⑥)してアプリケーションに渡す仕組みとすることで、既設アプリケーションを修正することなく流用することが実現できた。

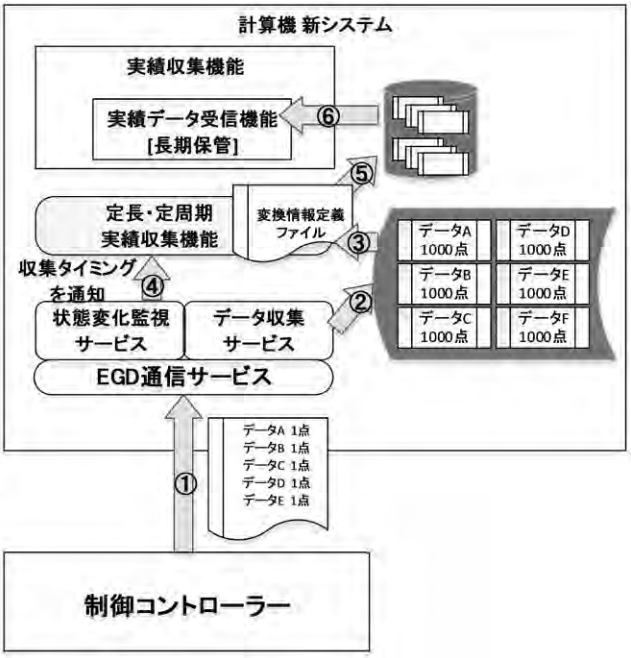

図６．定長・定周期実績収集（新システム）

更に、収集データの項目追加や今後続く更新案件への適応を考慮し、ユーザーが容易にメンテナンスできるようにメモリー上の情報を既存データのレイアウトに変換する様に定義ファイルで設定できるようにした。また、収集データの単位とアプリケーションが扱う単位の相違も考慮し、容易な設定パラメーターで単位変換できる仕組みにした。

４．まとめ

産業用ソフトウェア向けに新規ＡＰＩを開発し、EZprogrammingの関数群にAPIを組み込み、さらに実績収集機能のインターフェース相違をなくしたことで、既設アプリケーションを変更することなく流用が可能となり、短納期で既設と同等の品質を確保する新たな開発プロセスを確立することができた。

５．今後の展望

EZprogrammingは多くのシステムに適応されているが、ユーザーのニーズにより機能拡張が行われてきた。今後も、更新案件をスムーズに遂行するために、拡張されている機能の調査および開発を行い、開発プロセスの適応範囲拡大に努める。

６．むすび

既設アプリケーションの流用率向上を実現し、新規開発を無くすことを最終目標とした開発プロセスを推進することが、更なる工数削減および高品質の維持につながり、当所の強みになると考える。

最後に、本システム開発において社内外の開発支援をいただいた関係各位に深く感謝申し上げる。

執筆者紹介

藤原　宏文　　フジワラ　ヒロフミ
1989年入社。主に鉄鋼システムのソフトウェア開発に従事。現在、トータルソリューション事業所技術第2部工業第2課。