廃棄物管理ソフトウェアの要件の進化

デイビッド・W・ジェームズ
（David W. James）

放射性廃棄物のソフトウェア市場は、発電所、処理業者、仲介業者を含め、全体で数百人のユーザしかいない。しかし長期的には、ユーザとデベロッパの協力によって、より堅牢な製品が登場するであろう。

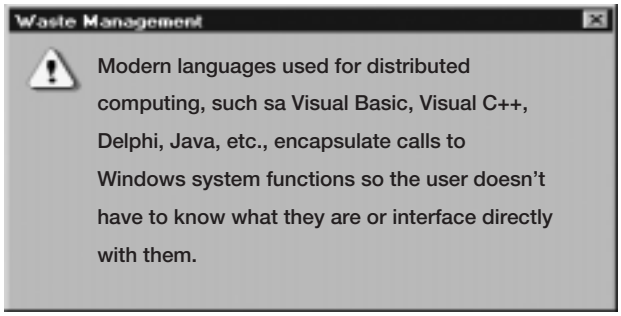
パソコンがオフィスに最初に導入され始めたのは、ちょうど20年前であった。当時筆者は、ベクテル社で働いていた。部署全体にあてがわれた1台のコンピュータは、IBM社のPC-XTであり、部署の全員が利用できる場所に置かれた。登録用紙を作成し、使用時間を区切ってコンピュータを共有した。ソフトウェアとして、dBaseと表計算プログラムLotus 1-2-3があった。ハードディスク容量は、数メガバイト程度だった。自分で作成したファイルは、各自フロッピーディスクに保管した。コンピュータが初めて導入されたとき、全員が集まって感嘆の声を上げた。

当時コンピュータは、まだコンピュータ部門の管理下にあった。コンピュータ部門（と会社）は、コンピュータ使用料をとることで多くの金を稼いだ。ほとんどのプログラムは、バッチ・プログラムであった。入力カードのセットアップ・ファイルが提供され、プログラムはルーチンを実行後に停止した。たとえ時間がかかっても（一晩かけてもよかった）、そのようなプログラムを安価に実行できるようになって、プロジェクト・コストが大幅に低減し、コンピュータ関連の所得は激減した。コンピュータの購入規則と使用法をめぐって、争奪戦が盛んであった。しかし、方向は定められた。

システムとOS

当初のコンピュータは、16ビットのDOSオペレーティング・システム（OS）によって制約された。その仕組みを以下に説明する。コンピュータ・メモリ内のすべてのデータには、アドレスが割り当てられる。アドレスは整数値であり、そのサイズは整数値を規定するレジスタの数によって決まる。16ビット・システムでは、各整数値に16個のバイナリ・レジスタがある。生成できる最大の数字は、2の16乗（約64,000）である。これによって、実行できるプログラムのサイズ、取り扱えるグラフィックス、およびメモリで処理できるデータ量が制約された。全セグメントが64Kのアドレス空間を使い果たすと、コンピュータは最果ての地をさまよい始め、再起動する必要があった。実際に起きていたことは、整数値が32,768を超えると、−32,768に移行し、そこから0に向って増えるように計数された。

Windows 3.0および3.1は、DOS OS上で作動したので、やはり64Kのアドレス空間の制約を受けた。1990年代初めに、最初の32ビット・コンピュータが登場した。32ビットになり、アドレス空間は400万Kほどに拡大した。インテル社の80386プロセッサがベースになった。われわれは、このプロセッサのために多くの資金を投じたが、OSとほとん



どすべてのソフトウェアは、16ビットの制約のままだったので、支出に見合う価値が実際に得られなかった。32ビット対応システムとしてWindows 95が登場したとき、われわれはすでに386型コンピュータを買い替えていた。Windows 95とその後のWindows 98は、いずれも32ビットに対応したが、16ビット・プログラムも作動するようになっていた。

今日普通に使われているWindows NT、Windows 2000、Windows XPなどのOSは、完全な32ビット・システムである。OSのRAM容量とハードディスク容量は、劇的に増大した。かつては、メモリ管理に注意が必要だったが、今では二次的な考慮事項になっている。RAMとハードディスクの容量拡大を促しているのは、ほとんどがグラフィックス関連であるが、それは放射性廃棄物ソフトウェアには過大に組み込まれていない。放射性廃棄物関連のさまざまな機能のためにわれわれが使うほとんどのソフトウェアにおいて、OS関連の問題が起きるとすれば、おそらくソフトウェアが壊れたか、他に作動しているプログラムが多すぎることによるであろう。

1991年に最初のSample Analysis Programを導入したとき、筆者はそれを、Windows 3.1で作動するC++プログラムとして作成した。筆者は、数カ所の発電所を訪問し、多くの発電所がまだWindowsに移行していないことを知った。実際、訪問先の部署には、専用のコンピュータがないところもあった。プログラムをインストールするコンピュータがないので、筆者は、ソフトウェアのデモンストレーションだけの目的で、最初のラップトップ・コンピュータを購入した。発電所で使用されているOSとデスクトップ構成によって、どのソフトウェアを購入するかが決まったので、Windowsプログラムに対する抵抗もあった。しかし、1993年にはWindows 3.1が一般に使われるようになり、個別のデスクトップが標準になり始めた。

その当時、プログラムにバグが発見された場合（通常は未処置の例外）、データ・ファイルを入手し、データの破損かプログラムの問題かを判定し、問題を修正して、ユーザに修正版を提供した。インターネットはなかったので、すべて郵送で対処する必要があった。問題を解決するのに、往々にして10日から2週間を要した。最初が間違っていたら、すべてのプロセスを初めからやり直さなければならなかった。代案として、モデムを通して直接接続する方法もあった。その方法を双方のユーザが熟知していれば、2,400ボーの通信速度で600キロバイトのデータを2、3時間かけて転送できた。ただし、その間に電話とコンピュータを占有でき、誰からも電話がかかってこないことが条件であった。今日、プログラムの問題は、数時間程度で解決できることが多い。プログラムの変更を制限する要因は、改訂の管理である。適切な処置によって、短期的に改訂を回避できる。しかし最終的には、プログラムを改訂する必要がある。

ユーザ・レベルでは、プログラムが改訂されると、変更履歴を確認し、改訂されたソフトウェアをインストールし、改訂プログラムが適切に機能することを試験・確認する必要がある。これはいずれも時間がかかり、ほとんどのユーザが望まない作業である。短期間に多くの変更を行いすぎると、ユーザは更新から取り残されることになる。そうなると、更新の連続性が途切れ、構成が損なわれるかもしれない。その結果、改訂版のバグ取りにより多くの時間がかかり、ユーザとプログラマーの不満がともに増すことになりかねない。当然ながら、プログラマーが責めを受けることになる。

プログラミングの問題

ほとんどの技術プログラムは、FORTRANで書かれている。FORTRANは、コンピュータ・プログラミングのための最古の言語ではないが、かなり古いほうである。FORTRANは、アセンブリ言語から、科学プログラミングのために必要な特定の算術演算への橋渡し役として開発された。コンピュータとの直接のやり取りがほとんどないことから、高級言語と考えられる。原子力用途で今日使われているプログラムの多くは、1960年代に書かれたものか、当時書かれたプログラムから派生したもので、当初のFORTRANプログラムからの変更点はわずかである。FORTRAN本体が提供するユーザ・インタフェースは限定的であり、分散コンピューティングに適するようには設計されていない。最新のツールを使ってシェル・プログラムを作成することにより、入力ファイルを構成し、FORTRANコンポーネントを呼び出し、出力ファイルを再解釈することができる。分散コンピューティングは、グラフィック・ユーザ・インタフェースの標準である。基本として、プログラムがRAMにロードされ、コマンドを待ち受ける。必要な作業に応じて、複数の地点か

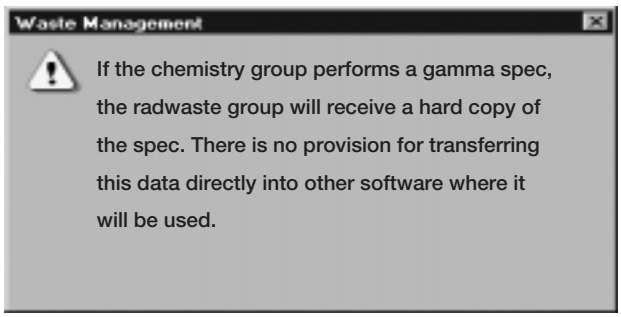
ら随時プログラムを入力できる。

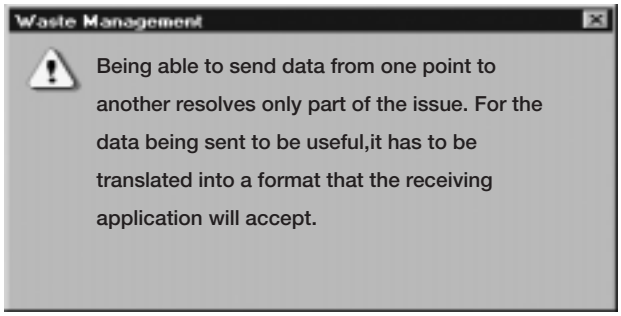
FORTRAN を擁護する主な理由の 1 つは、科学ソフトウェアにおいて、他の言語で作成されたプログラムより同じ関数を速く実行できることであった。しかし、プロセッサが 8 MHz から 2 GHz 以上に高速化したことで、このようなプログラムの多くは瞬時に実行されるようになり、速度はあまり問題ではなくなった。より大きな要因は、FORTRAN で書かれた過去からのソフトウェアが多量に蓄積されていることと、このソフトウェアを維持・促進させている国立研究所につきまとう聖職意識である。このソフトウェアを聖職の外で使用することは可能だが、それなりの覚悟が必要である。

Visual Basic、Visual C++、Delphi、Java などの、分散コンピューティングのための最近の言語には、Windows システム関数の呼び出しがカプセル化されているので、ユーザは呼び出しを知る必要がなく、それを直接扱うこともない。C++ のような C 言語系は、カプセル化された独自の関数をバイパスして、Windows インターフェース関数を直接呼び出す。Windows プラットフォームを完全に網羅するカプセル化は存在しないので、この仕組みは有用である。Win32プログラミング・インターフェースの説明は、5 巻、数千ページにわたるが、関数呼び出しとそのパラメータを列挙するだけで、2 巻を要している。

これらの言語の間には、トレードオフがある。Windows 関数が変更されれば、それをカプセル化したコンポーネントは機能しないかもしれない。マイクロソフト社が OS を更新するたびに、実際に一部の関数が変更される。従来のプログラムを新しい環境で実行すると、チェックボックスをクリックしたとたん、不意にクラッシュすることがある。プログラムは、新しい環境の標準である32ビットの整数ではなく、16ビットの短い整数を予想していたためである。また、使用している開発プラットフォームの最新版プログラミング・インターフェースが、新しい Windows の利点を活かすために変更されることがある。

Windows アプリケーションのための別のタイプの開発プラットフォームは、一部の Windows ソフトウェア（Excel、Access、および他のさまざまなデータベース・ソフトウェアなど）とともに提供される。これらは、プログラム関数への呼び出しをカプセル化したスクリプト言語で書かれ、プログラム関数はさらに OS 関数への呼び出しをカプセル化している。このようなプログラムは、OS が変更された場合に、おそらくもっとも問題が起きやすい。また、親プログラムが、望ましくなくなったり置き換えられたりすることによる影響も受けやすい。通常は、プラットフォームを変更するより、変化のペースに合わせるほうが容易である。すべてのメーカーは、ユーザを自社製品につなぎ止めたいと思っている。





放射性廃棄物業界のデスクトップ

当業界のほとんどの人は、データ入力が好きではなく、また必ずしも得意ではない。作業は遅く、間違いが生じやすい。インターネットのおかげで、データを共有する機会が増え続けている。ある地点から別の地点にデータを送れることは、問題解決の一部でしかない。送られたデータが有用であるためには、受け取り側のアプリケーションが読み取れる形式にデータを変換しなければならない。放射化学研究室ごと、処分サイトごと、処理業者ごと、および発電所ごとに、データ管理のためのシステムと形式の要件がそれぞれ異なる。データの用途に応じて、データ転送とは別に考慮すべき特定の情報要件がある。「ゲスト」プログラムがデータを有効に利用するには、入力データの形式と内容を正確に知るとともに、データ変換のための特定の機能を含める必要がある。「ホスト」ソフトウェアは、その形式が多少なりともわかりやすい構成になっているかもしれない。

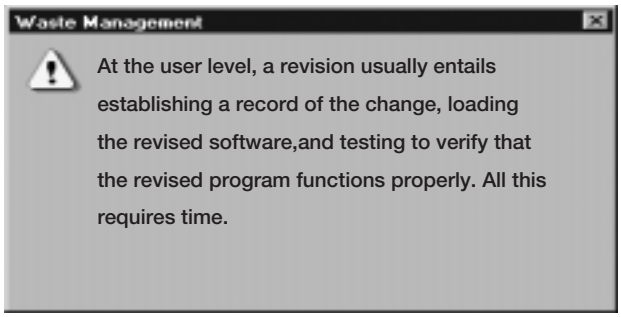
理想的には、放射性廃棄物業務のためのソフトウェアは、主要な作業機能をすべて単一のデータベースに統合するのが望ましい。その機能として以下が挙げられる。

- ●輸送
- ●廃棄物インベントリー
- ●放射線サーベイ・データ
- ●同位元素報告書

●資材と供給品
●職員
●報告（月報、米原子力規制委員会の報告書）
●連絡先
●契約
●作業スケジュール作成
●特殊な分析（遮へい、放射性崩壊、経済性）
●コミュニケーション

現在、これらの機能の多くは、ばらばらに取り扱われている。一部の機能は、手作業で行われている。例えば､化学部門がガンマ線分光分析を行った場合、放射性廃棄物部門はスペクトルのハードコピーを受け取る。このデータを、他のソフトウェアで使えるように直接変換する手段が用意されていない。輸送作業と支出の記録には、表計算ソフトが使われるかもしれない。処分サイトに関する通知は、ファックスで送られる。アドレス情報の多くは、ローロデックス（回転式卓上カードファイル）内に保管されているかもしれない。契約情報は、書類ケース内に保管され､その進展状況は表計算ソフトで記録される。

このような状況から前進するには、いくつかの段階が必要である。それには、情報・通信要件の詳細なモデル化、データベースの設計と形式の開発、および統合のためのユーザ・インタフェースの提供が含まれる。この作業は、アクセスが要求される多くのデータが保管されるローカル・データ管理システムの制約の下で行う必要がある。放射能カットオフ・レベルを標準化し、最小検出限界、四捨五入の問題、報告単位、および有効数字を取り扱うことは、小さな前進だが重要な段階になるかもしれない。



小さな市場

マイクロソフト社が自社のプログラムを更新する際、そのベータ版を、例えば１万人ほどの試用者に配布する。試用者は、数ヶ月にわたってプログラムを試用し、遭遇した問題とインターフェースの問題を、マイクロソフト社にフィードバックする。改訂版について満足な点と不満な点を記入するための質問表があると思われる。この情報はすべてデータベースに入力され、修正すべき項目が特定される。このプロセスが完了すると、マイクロソフト社は改訂版を発売し、1,000万パッケージくらい販売して、何十億ドルかを稼ぐ。

放射性廃棄物ソフトウェアの場合、発電所、処理業者、仲介業者を含め、市場全体で数百人のユーザしかいない。競争により、ほとんどのデベロッパにとって市場はさらに小さくなる。ほとんどの大手企業にとって、市場が小さすぎるので、前述のプログラムを統合するために何千人・時間もの労力を投入することができない。多数のベータ版試用者に恵まれているわけではなく、更新コストを多数のユーザに分散できるわけでもない。試験と検証は、デベロッパが初期ユーザとともに行わなければならない。このため、ソフトウェアは高価で、当初は不安定である。長期的には、ユーザとデベロッパの協力によって、より堅牢な製品が登場するであろう。

放射性廃棄物のためのソフトウェアは、非常に特殊化している。会計、在庫管理、資材追跡記録など、別種のアプリケーションと共通する部分もある。会計機能は、多層化している。各放射性核種は、その取り扱いについて独自の規則を持つ独自通貨のように扱われる。すべてのデータは、概算、重量、容積、放射能測定、および線量率に基づく。しかし、分析、報告、および規制の要件は、放射性廃棄物に独特のものがある。どのような応用でも当該分野の専門知識が最重要であるが、放射性廃棄物への応用は特にそうである。ソフトウェア・ソリューションを各分野のニーズと組み合わせる能力は、そのニーズが満たされるかどうかでまず判定される。よりよい方法を提供するだけでは不十分である。新製品は、新たな検証、新たな実装、プログラムへの新たなデータ入力を伴い、それらはいずれもユーザの支出増につながる。

究極の要件

放射性廃棄物ソフトウェアの要件の進化についてあるユーザに意見を求めたところ、「使える必要がある」というのが答えだった。この要件の重要性は、どんなに強調しても過分ではない。 ■

デイビッド・W・ジェームズ氏は、ミネソタ州ノースオークスにある D.W.ジェームズ&アソシエーツ社の経営者である。本記事は、2003年７月16～18日にルイジアナ州ニューオーリンズで開かれた2003年電力研究所（EPRI）低レベル廃棄物国際会議での発表論文に基づく。