4A-02

ソフトウェア群の差分アップデート手法

菅原 直樹† 三浦 昭浩† 古澤 康一†
三菱電機株式会社 情報技術総合研究所†

1. はじめに

近年、ソフトウェアの配布方法の主流がインターネットを介した方法へと変化していることや、ソフトウェアのサイズが増加していることにより、ソフトウェアのダウンロード、インストールに要する時間の増加が問題となっている。特に、機能拡張が頻繁に行われるソフトウェアでは、アップデートの度にダウンロード、インストールを行う必要があり、ユーザの負担が大きい。

ソフトウェアのダウンロード、インストールに要する時間を短縮する方法として、バージョン間の差分のみをインストーラに含め、インストールサイズを削減する差分アップデートがあるが、複数のソフトウェアで共有されているライブラリ等の共有ファイルを持つソフトウェアに対しては、従来手法を適用することが困難であった。

そこで本稿では、共有ファイルを持つソフトウェア群の差分アップデート手法を提案する。

2. 従来手法における課題

差分アップデートは、追加、更新されるファイルのみをアップデートするため、インストーラのサイズ削減、更新に要する時間の短縮が可能である。一般的に、差分インストーラは以下の手順でユーザ環境のソフトウェアをアップデートする。

① ユーザ環境におけるソフトウェアのバージョンを確認する。
② ソフトウェアのバージョンから、適用が必要な差分と適用順序を判断する。
③ 差分の適用を行う。

上記②の手順では、ソフトウェアのバージョンが同じユーザ環境であれば、そのソフトウェアを構成する各ファイルのバージョンも同じであることを前提として適用する差分を判断する。

しかし、共有ファイルが存在する場合、ソフトウェアのバージョンが同じユーザ環境でも、共有ファイルのバージョンが同じであるとは限らない。これは、共有ファイルは他のソフトウェアからもアップデートされるためである(図 1)。

共有ファイルのバージョンが複数存在する可能性がある場合、各バージョンのファイルをアップデートする差分を用意する必要がある。しかし、従来手法では単一のソフトウェアのバージョン間で差分抽出を行うため、他のソフトウェアにおけるアップデートに対応した差分を用意することが難しかった。また、どの差分をどのような順序で適用するかを判断することが困難であった。

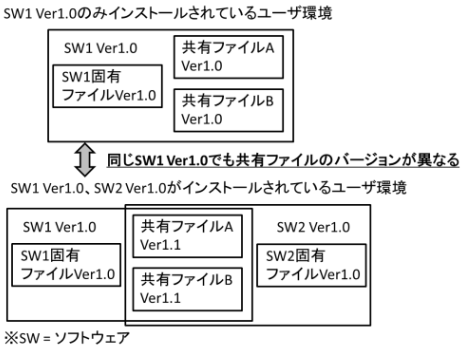

図 1 従来手法における課題

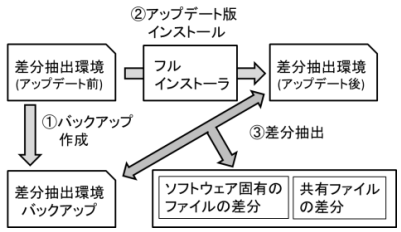

図 2 提案手法における差分の作成方法

3. 提案手法の概要

2 章に示した課題を解決するために、以下の差分アップデート手法を提案する。提案手法が従来手法と異なる点は、差分の抽出方法と、適用する差分の選択方法の 2 つである。以下、それぞれの詳細と提案手法における運用方法を示す。

3.1. 差分の抽出方法

提案手法では共有ファイルを持つソフトウェア群全体を 1 つの単位として差分を抽出する。具体的には、1 台のコンピュータを差分抽出環境とし、各ソフトウェアのアップデート版が開発される度にその時点の環境のバックアップを取り、インストーラを用いてアップデート版をフルインストールする。そしてバックアップとアップデート後の差分を抽出する(図 2)。

"Difference Updating Method for Software Having Shared Files"
† Information Technology R&D Center, Mitsubishi Electric Corporation

表 1　提案手法で抽出される差分

インストールしたソフトウェア		SW1	SW2	SW3	SW1	SW2
ソフトウェアのバージョン		1.0	1.0	1.0	2.0	2.0
抽出される差分	共有ファイルA	-	Ver1.0 ↓ Ver1.1	Ver1.1 ↓ Ver1.2	-	Ver1.2 ↓ Ver1.3
	共有ファイルB	-	Ver1.0 ↓ Ver1.1	-	Ver1.1 ↓ Ver1.2	-

※SW＝ソフトウェア

表 2　提案手法で用いる適用差分判断リスト

インストールされたソフトウェア		SW1	SW2	SW3	SW1	SW2
ソフトウェアのバージョン		1.0	1.0	1.0	2.0	2.0
ポインタ情報	共有ファイルA	1	1	2	1	0
	共有ファイルB	1	2	1	0	0

差分は、共有ファイルの差分と、各ソフトウェア固有のファイルの差分に分けて管理する。

表 1 は、共有ファイル A,B を持つソフトウェア 1～3(SW1～3)について、左に示すソフトウェアから順に差分抽出環境にインストールし、提案手法を用いた場合に抽出される共有ファイルの差分の例を示している。表から読み取れるように、共有ファイル A,B ともに、Ver1.0 から、段階的にアップデートを行う差分が抽出されているため、これらの差分を順に適用すれば、全てのバージョンの共有ファイルをアップデート可能である。

例えば、共有ファイル A Ver1.0 であれば、SW2 Ver1.0 インストール時の差分、SW3 Ver1.0 インストール時の差分、SW2 Ver2.0 インストール時の差分という順序で適用することで、Ver1.3 まで段階的にアップデートすることが可能である。

3.2. 適用する差分の選択方法

提案手法では、適用する差分を判断するため、表 2 に示す適用差分判断リストを用いる。

表 2 のポインタ情報とは、どの差分を用いれば、その共有ファイルをアップデートできるかを示す情報である。例えば、表 2 において、SW1 Ver1.0、共有ファイル A のポインタ情報の値は 1 である。そこで、このポインタ情報の値の分だけ右の列を確認すると、その列は SW2 Ver1.0 に対応した列である。表 1 で SW2 Ver1.0 の列を確認すると、共有ファイル A を Ver1.0 から Ver1.1 にアップデートする差分が抽出されている。この差分を適用することで、共有ファイル A を 1 段階アップデートすることが可能である。ポインタ情報が 0 の場合は、その列のソフトウェアに含まれる共有ファイルは最新バージョンであり、アップデートする差分は存在しないことを意味する。

ユーザ環境における共有ファイルを最後に更新したソフトウェアに対応した列を始点、アップデート後のソフトウェアと同じバージョンの共有ファイルを持つソフトウェアに対応した列を終点とし、上記の方法で始点から終点に至るまで列を順にたどるとき、始点を除く、経由した列に対応した差分が適用する差分である。

始点は、ユーザ環境にインストールされたソフトウェアに対応した列の中で、そのポインタ情報の指す列が最も右に位置する列となる。

終点となり得るのは、アップデート後のソフトウェアに対応した列(以下、主終点候補列)と、アップデート後のソフトウェアと同バージョンの共有ファイルを持つソフトウェアに対応した列(以下、副終点候補列)である。この中で、列を順にたどる過程で最初に現れた列が終点となる。

副終点候補列は以下の方法で求める。主終点候補列のポインタ情報が 0 の場合、ポインタ情報が 0 の列が副終点候補列である。主終点候補列のポインタ情報が 0 以外の場合、主終点候補列のポインタ情報が指す列を確認する。ポインタ情報の指す列が、その列と同じ列が副終点候補列である。

共有ファイルごとに適用する差分の選択を行うことで、適用する全ての差分を判断できる。これらの差分は古いものから順に適用する。

3.3. 提案手法における運用方法

3.1 節で示したように、提案手法では各ソフトウェアのアップデート版が開発されるたびにインストールを行い、アップデート前後の差分を抽出する。その際、適用差分判断リストの更新も同時に行い、リストを最新の状態にする。

そして、差分インストーラを作成する際には、差分の他に、適用差分判断リストを差分インストーラに含める。この適用差分判断リストは、差分インストーラが、3.2 節に示す方法を実現するために用いられる。

4. 提案手法の効果

提案手法を用いることで、ユーザ環境によって共有ファイルのバージョンが異なる場合でも、各ユーザ環境に対応可能な差分を用意し、適切な差分を適用可能である。また、提案手法では、複数の異なるソフトウェアについて、各共有ファイルのアップデートに必要な差分を選択するための情報を 1 つの適用差分判断リストに集約できるため、複数のリストを参照する必要がない。さらに、リストの項目としてバージョンではなくポインタ情報を記述することで、適用する必要がある差分や、適用順序を容易に判断できる。

5. 終わりに

本稿では、共有ファイルを持つソフトウェア群を差分アップデートする手法として、ソフトウェア群全体を対象に差分を抽出し、リストを用いて適用する差分を判断する方法を提案した。今後は、提案手法の実装、評価を進める。