変更履歴を利用したパッケージ管理のためのバージョン管理システム

1431090 藤井亮太 岩崎研究室

1　背景

パッケージとは，プログラムとそのドキュメントのように，互いに関連する複数のファイルを一つにまとめたものであり，ソフトウェアを管理する際の単位として広く用いられている．例えば，多くの Linux ディストリビューションにおいては，様々なアプリケーションがパッケージの形で管理されている．

パッケージのインストールや更新，削除といった管理作業には，パッケージマネージャと呼ばれる専用のツールを用いる．ユーザは，パッケージマネージャを用いてインストールを行うことで，パッケージを利用できるようになる．パッケージには，内容物であるファイルに加えて，パッケージの名称やバージョン，含まれているファイルの一覧などのメタデータが含まれている．パッケージマネージャは，このメタデータを利用してパッケージを管理する．

2　パッケージに対する変更

パッケージに含まれるファイルの中には，インストール後にユーザが変更を加えて利用することが想定されるものがある．例えば，プログラムとその設定ファイルを含むようなパッケージをインストールしたユーザは，図 1 (a) のように自分の利用目的に合わせて設定ファイルを編集する．また，パッケージの提供者も，ソフトウェアのバージョンアップなどに伴い，図 1 (b) のようにパッケージに含まれる設定ファイルの内容を変更することがある．

ユーザがパッケージを更新する際には，パッケージ提供者による変更を取り込む必要がある．そのため，ユーザによる変更とパッケージ提供者による変更を適切にマージする必要がある．

ユーザによる変更とパッケージ提供者による変更は，互いに相反する内容を含む可能性がある．また，そうでない場合であっても，パッケージ提供者の変更がユーザの意図に反している可能性もある．そのため，マージ作業を行う際にはユーザの判断が必要である．

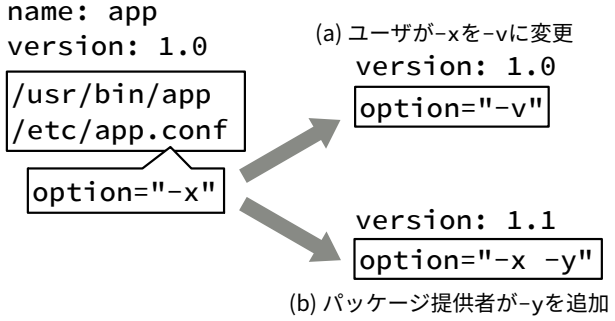

図 1　パッケージに対する変更

3　問題点

ユーザによる変更とパッケージ提供者による変更をマージする際，ユーザが正しい判断をするためには，ユーザがパッケージに対してどのような変更を行ってきたか，という変更履歴が必要である．しかし，パッケージマネージャはこのような情報を保持しておらず，変更履歴の管理はユーザ任せとなっている．

また，ユーザが何らかの形で変更履歴を管理していたとしても，その情報はパッケージマネージャから利用することができない．そのため，変更履歴を利用する際はユーザ自身で全ての作業を行う必要がある．

このように，パッケージに関する変更履歴の管理と利用がユーザに委ねられている状況は，継続的にパッケージの更新と管理を行う上でユーザの負担となる．

4　目的と方針

本研究では，継続的なパッケージ管理におけるユーザの負担を軽減することを目的とする．

そのための方針として，バージョン管理システム (VCS) を用いてパッケージに対する変更の履歴を管理する．VCS とは，ファイルの変更履歴を記録し，後から利用できるようにするシステムであるが，本研究における VCS は，単にファイルの変更履歴を管理するだけでなく，変更履歴とパッケージのメタデータを対応付けて管理することで，変更履歴をパッケージ管理に利用可能とする．

さらに，プラグインによってパッケージマネージャを拡張することで，VCS とパッケージマネージャを連携させ，ユーザがパッケージマネージャの支援を受けながら変更履歴の管理とパッケージ管理への利用を行えるようにする．そのため，本研究のシステムが想定するパッケージマネージャは，プラグインによる機能拡張が可能なものであるものとする．

5　実装

変更履歴の管理には，既存の VCS の一つである Git [1] を利用する．また，変更履歴とパッケージのメタデータを対応付けるために，パッケージのメタデータを通常のファイルと同様に Git を利用して管理する．

VCS とパッケージマネージャの連携は，パッケージマネージャのプラグインによってパッケージマネージャの動作をフックし，VCS を利用する処理を割りこませる形で実現する．対象とするパッケージマネージャとしては，Apt [2] と pkg [3] を予定している．

6　システムの動作

本研究のシステムの動作の概要を，図 2 を用いて述べる．ユーザがパッケージ app のバージョン 1.0 (以降，パッケージ名とバージョンを合わせて app-1.0 のように表す) をインストールすると，パッケージマネージャのプラグインが起動し，app-1.0 のメタデータ (a) と app-1.0 に含まれる設定ファイルの内容 (b) を VCS に記録する．このとき，本システムが (a) と (b) の対応関係を記録しておくことで，ファイルの内容がパッケージのどのバージョンを元にしたものであるか判別できるようにする．

その後，ユーザが設定ファイルを編集し，さらにパッケージ提供者が app-1.1 を公開した場合を考える．ユーザは設定ファイルを編集した後，本システムが提供する VCS のユーザインターフェースを用いて，変更後のファイルの内容 (c) を VCS に記録する．このとき，本システムによって (c) は変更元のパッケージである app-1.0 のメタデータ (a) と対応付けられる．

パッケージ提供者が app-1.1 を公開した後，ユーザが app-1.0 から app-1.1 への更新を行うと，インストール時と同様にパッケージマネージャのプラグインが起動し，app-1.1 のメタデータ (d) と app-1.1 に含まれる設定ファイルの内容 (e)，および (d) と (e) の対応関係を VCS に記録する．

ここで，app-1.0 のメタデータ (a) に関連付けられた (b) と (c) を VCS から取り出して比較することで，ユーザがパッケージをインストールした後に加えた変更の差分がわかる．また，app-1.0 のメタデータ (a) に関連付けられた (b) と，app-1.1 のメタデータ (d) に関連付けられた (e) を比較することで，パッケージ提供者がバージョンアップに伴って加えた変更の差分がわかる．

このようにして得られる双方の差分をユーザに提示することで，ユーザは自分が行った変更がどのようなもので，パッケージ提供者が行った変更がどのようなものであるかを明確に認識した上で，どのようにマージを行えばよいかを判断することができる．また，変更履歴の管理や差分の比較などの作業は，システムによって可能な限り自動的に行われるため，ユーザは判断に集中できる．

7　関連研究

etckeeper [4] は設定ファイルのバージョン管理を支援するツールであり，パッケージマネージャの動作に連動して VCS を起動することで，設定ファイルの変更履歴を記録する．etckeeper では，パッケージに関する情報は管理されていないため，変更履歴をパッケージマネージャから利用することができない．本研究のシステムでは，変更履歴をパッケージマネージャから利用できるようにすることで，ユーザがパッケージマネージャの支援を受けながら変更履歴を利用することができる．

Conary [5] は，VCS が統合されたパッケージマネージャであり，インストールや更新といった通常のパッケージ操作が VCS 上の操作によって実現されている．そのため，本研究で提案するシステムと同様に，変更履歴の管理とパッケージ管理への利用が可能である．一方で，Conary はパッケージマネージャであるため，他のパッケージマネージャを使用している環境では使用できない．本研究のシステムは，プラグインによって動作をフックすることが可能という条件を満たすものであれば，複数のパッケージマネージャに対応することが可能である．

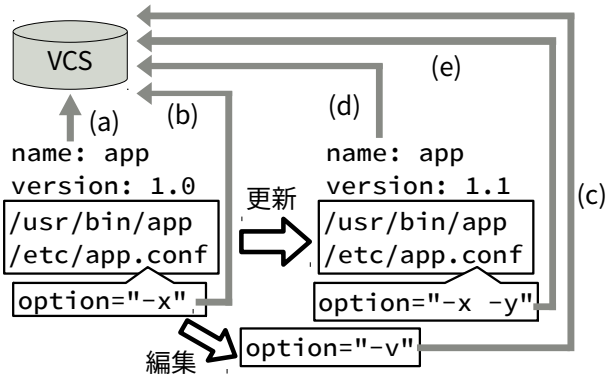

図 2　システムの動作

8　現状と今後

これまでに，パッケージ管理に関連する既存の研究及び技術の調査を行い，本研究で提案するシステムの設計を行った．

今後は，システムを実装し，現実の OS で利用されているパッケージを用いて，システムの有用性を評価する予定である．また，9 月に行われる日本ソフトウェア科学会大会での発表を予定している．

参考文献

[1] Git. http://git-scm.com/.
[2] Apt. https://wiki.debian.org/Apt.
[3] pkg. https://wiki.freebsd.org/pkgng.
[4] etckeeper. http://etckeeper.branchable.com/.
[5] Michael K. Johnson, Erik W. Troan, and Matthew S. Wilson. Repository-based System Management Using Conary. In *Proceedings of the Ottawa Linux Symposium*, Vol. 2, pp. 557–571, 2004.