リンク構造ファイルシステムのためのファイル編集アプリケーションの試作

08T274　山本道明　(最所研究室)

リンク構造ファイルシステムのファイルを編集するファイル編集アプリケーションの設計と実装を行う.

1.　はじめに

伝統的なファイルシステムでは，ファイルの汎用性を高めるために，ファイルを構造のもたないバイト列の集合として扱ってきた．しかし，本研究室で研究しているリンク構造ファイルシステム[1]では，ファイルを可変長バイト列を単位とするブロック(以降，ブロック)の集合としてとらえ，従来のファイルシステムではできなかった挿入操作，削除操作を実現できる．先行研究では，ファイル操作の履歴を保存し過去のバージョンを参照できる履歴機能[2]の追加が行われている．しかし，現在このファイルシステムを直接利用できるアプリケーションがないため，本ファイルシステムにアプリケーション側から求められる要求を検討できていない．そこで本研究では，リンク構造ファイルシステムを利用できるファイル編集アプリを試作することにより，アプリケーションの開発の指針を確立すると共に，アプリケーション側から要求されるファイルシステムの機能を洗い出す.

2.　リンク構造ファイルシステムの概要

リンク構造ファイルシステムでは，ブロックの挿入，削除の 2 つの操作でファイルを変更する．図 1 に示すように，挿入，削除が行われると，それぞれ履歴を保持する構造が作成される.

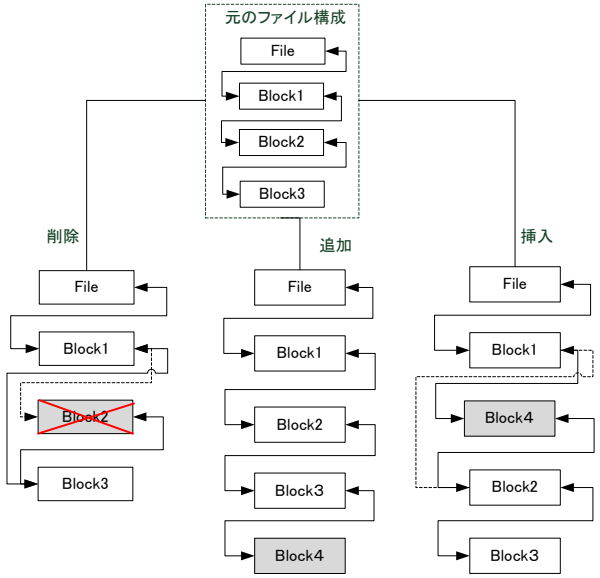

図 1：リンク構造ファイルシステムの概要図

挿入操作は，指定した場所に新規ブロックを入れて，リンク情報を変更する．図では，示してないがファイルの先頭と末尾に空のダミーブロックを持たせることで，追加と挿入操作を統合している．削除操作は，指定したブロックをリスト内から消し，リンク情報を変更する.

3.　ファイル編集アプリケーション

リンク構造ファイルシステムのブロック構造を意識し，リンク構造ファイルシステムの最新バージョンのブロックを自由に編集することができるアプリケーションである．このアプリケーションは，ファイルシステムの挿入操作，削除操作を直接利用できるため，アプリケーションで行われるファイル操作の履歴をファイルシステム側に残すことができる.

4.　設計

アプリケーションの開発指針として，ファイルシステムからブロック単位で，メモリに読み込みブロック情報を維持しながら編集する．このアプリは，リンク構造ファイルシステムのブロック構造を維持するためにリストで管理する.

ユーザは，ブロックを自由に編集し，ファイルシステム側に編集した内容を反映できる．ファイル編集の際に考えられる機能として，以下の 13 個を実装することにした.

読み出し:リンク構造ファイルシステムの最新バージョンをアプリケーションに読み出す.
ブロック編集：指定したブロックの内容を編集する.
新規ブロック作成:指定した位置に新規のブロックを作成する.
移動:選択した複数のブロックをブロックの順序を維持しながら指定した位置に移動する.
削除：選択した複数のブロックを削除する.
統合：複数のブロックを一つに統合する.
分割：一つのブロックを二つに分ける.
検索：全てのブロックから keyword を探し出し，色をつけ表示する.
置換：全てのブロックからテキスト情報の置換を行う.
ブロック情報：シーケンシャル番号，作成時間，ブロックサイズ，新規フラグ等のフラグ情報などを保持する.
差分表示:編集前のファイル内容と編集中のファイル内容の差分情報を表示する機能である.
ファイル読み出し:ブロック構造でない一般のファイルシステム上のファイルを読み出し．ブロック構造に直しながらアプリケーションに読み込む.
保存:編集した内容の保存．変更された部分をリンク

構造ファイルシステムの挿入，削除操作を用いて実現する．

行われた操作は，ファイルシステムの挿入操作，削除操作に対応させる．以下に例を示す．

移動：移動するブロックの削除，移動先への挿入．
統合：統合前のブロックの削除，統合後のブロックの同一位置への挿入．
分割：分割前のブロックの削除，分割した 2 つのブロックの同一位置への挿入．
保存：各編集操作に対応したファイルシステムへの操作．

これらの操作の特徴として，ユーザがブロックの意味を変更したい時に効果的である．使用例として，独立したブロックを引用して挿入したい時などが想定される．

5. 実装と考察

設計したアプリケーションのデータ構造と機能を，それぞれ GTK2 ツールと Ruby 言語を用いて，実装した．実装例として，移動操作を説明する．移動操作前では，図 2 で示すように Block1 と Block2 を選択し，移動用のインターフェイスを使用してブロック位置を指定する．位置の指定方法は，ブロックの先頭，末尾，ブロックの位置情報を格納したリストから指定できる．新規に挿入したブロックは，読み出しブロックと区別するため，色が異なる．移動操作後は，図 3 に示すように，移動するブロックが指定位置に挿入された状態となる．

ファイル編集アプリケーションを実装することでリンク構造ファイルシステムが求める機能が検討できた．1 つ目は，履歴を残さない操作の実現である．ファイル編集アプリケーションで保存する時にファイルシステムの挿入，削除操作を行い反映するため，履歴が残ってしまう．編集されているブロックに関しても，リンク構造ファイルシステムの挿入，削除操作を用いて反映している．しかし，これでは別に参照している履歴が複数でき，履歴管理の負担が増える．履歴を残すために，ファイルの変更後の状態のみ必要な場合，その間にできる履歴は必要ない．従って，履歴の管理の負担軽減のために履歴を残さない操作が必要と考える．2 つ目は，リンク構造ファイルシステムの挿入操作，削除操作を代用する replace 機能の実現である．ファイル編集アプリケーションでは，編集されたブロックをリンク構造ファイルシステムに反映させる時，削除操作，挿入操作を用いて実現している．削除操作と挿入操作をまとめて実行する replace 機能があると，一度の操作で編集されたブロックをリンク構造ファイルシステムに反映できる．3 つ目は，リンク構造ファイルシステムの履歴を元に操作前のバージョンに戻す機能の実現である．今回試作したファイル編集アプリケーションでは、実装できていないが，編集中に前のバージョンに戻したいまたは前のバージョンを閲覧したいという要望が考えられる．そのため，リンク構造ファイルシステムで管理されている履歴情報を元に，戻る機能を実現する仕組みをファイルシステム側に必要だと考える．

図 2: 移動操作前のブロック状態

図 3: 移動操作後のブロック状態

6. まとめ

以上，リンク構造ファイルシステムを利用したファイル編集アプリの開発指針とアプリ側から要求されるファイルシステムの機能について述べてきた．機能面では，実用化する上で必要と思われる機能を実装した．将来的には，アプリ側で過去のバージョンを参照することも検討している．さらに編集しているバージョンの差分情報を読み込み，バージョン情報を更新する機能の追加も考えている．これは，読み込んだ差分情報で変更があるブロックに関して，挿入操作と削除操作を使用することで，実現が可能であると考えている．最新のバージョンブロックと過去のバージョンブロックを比較し，差分を表示できるようにすることで，過去の操作履歴を見ることができるようにすることも考えている．これにより，間違って変更された内容があれば，修正しやすくなる．

参考文献

[1] 津紀孝，最所圭三，“行指向ファイルシステムについて”,平成 18 年度 電気関係学会四国支部連合大会論文集, 15-37, p.242, 2006

[2] 森敏文．“リンク構造ファイルシステムへの履歴機能の追加．” 香川大学工学部．卒業論文．2008.