LifeLineにおけるファイルアクセス制御

5H−7

入交晃一　岸知二　坪谷英昭
日本電気(株) ソフトウェア生産技術開発本部

1　はじめに

我々は、ソフト開発環境を構築する際の共通基盤として、LifeLineの開発を行なっている[3]。

LifeLineは、成果物の論理的な構成を管理する機能、成果物に対する版管理、アクセス制御、インテグレーション制御等の機能、成果物中に定義された設計情報を管理する機能などをCのライブラリとして提供し、広い範囲のCASEツールに対する共通のプラットホームとなることを目的としている。

本稿ではLifeLineの機能のうち、複数人の共同開発時におけるファイルのアクセス制御の機能についてその考え方を述べる。

2　背景

ソフト開発環境においては複数人でのソフト開発に伴う様々な問題がある。情報のアクセス制御の問題もその中の1つである。ベースラインと作業領域間でのファイルのやりとりによってこれを解決する、リザーブ/デポジットの手法[1]が一般的に行なわれている。

すなわち、作業者はベースライン中にあるファイルにロックをかけ自分の作業領域にコピーし、そのコピーに対して修正作業を行なう。その間他の作業者はそのファイルに対して修正作業を行なうことはできないが参照することは可能である。修正作業が終ると作業者はそのファイルをベースラインにコピーしロックを解除する。

このような、1つのファイルに対し同時には1人しか修正作業が行なえない、という方法では複数の作業者が自由に並行に作業を進めることが困難である。

一方NSEでは、作業領域へのコピーは自由に行なわせて、ベースラインに戻す時に他の作業者の修正とマージするという楽観的なポリシーを取ることによって並行に作業を進めやすくしている[2]。しかしこの場合は、必ずしも修正のマージがうまくいくとは限らないという問題がある。

LifeLineでは、修正のための作業領域へのコピーと、ファイルに対するロックとを分離して扱うという考え方を採用しこの問題に対応している。

3　基本概念

3.1　プロジェクト

プロジェクトとはUNIXディレクトリに対応付けられた成果物の管理空間であり、全ての成果物はプロジェクト中のオブジェクトとして管理される。

File Access Controle in LifeLine
Koichi IRIMAJIRI, Tomoji KISHI, Hideaki TSUBOTANI
NEC Corporation

プロジェクトには、ベースラインに対応したルートプロジェクトと、作業者の作業領域に対応したサブプロジェクトがある。サブプロジェクト生成時には親プロジェクトを指定する。これによって1つのルートプロジェクトを頂点としたプロジェクトの階層構造ができる。これをプロジェクトツリーと呼ぶ(図1)。

プロジェクト中では、フォルダによって階層的な管理構造が定義される。フォルダの下には、他のフォルダ、ファイルといったオブジェクトが管理される。

1つのプロジェクトツリーを構成している各々のプロジェクト中からは原則として同一の論理構造(フォルダの階層構造)が見える。物理的にはオブジェクトは各プロジェクトに分散して管理され得るが、プロジェクトを通してアクセスしている限り、作業者はそれを意識する必要はない。

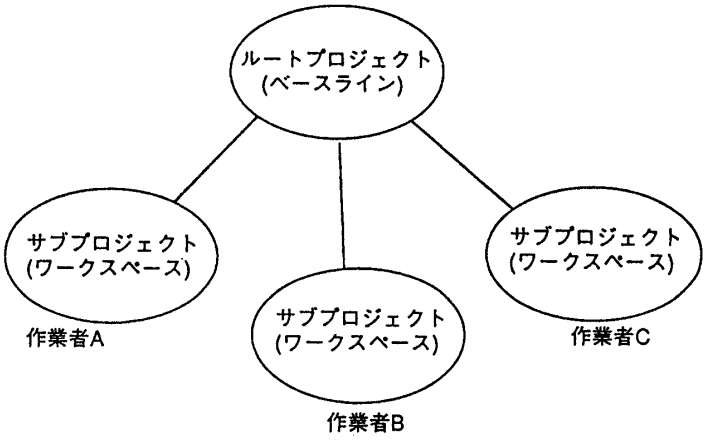

図1: プロジェクトツリー

3.2　基本操作

プロジェクト中のファイルに対するアクセスのために次の4つの基本操作を考える。

- 自プロジェクトへのコピー
  親プロジェクト中にあるファイルのコピーを行ない、自分のプロジェクト中にプライベートなファイルを作る操作である。この操作はファイルにロックがかかっていても可能である。
- 親プロジェクトへのコピー
  プライベートなファイルを親プロジェクトへコピーして返す操作である。自分がロックをかけているオブジェクトに対してのみ可能である。親プロジェク

トにあるファイルをこれ以外の方法で修正することはできない。

- 修正権の取得
  親プロジェクト中のファイルにロックをかける操作であり、これにより親プロジェクトへのコピーが可能になる。修正権の取得はその時点でロックがかかっていないファイルに対してのみ可能である。

- 修正権の放棄
  親プロジェクト中のファイルのロックを外す操作であり、この操作によって親プロジェクトへのコピーが不可能になる。この操作は、自分がロックをかけていたファイルに対してのみ可能である。

LifeLineにおいては、これらの機能を単体で提供するのではなく、以下で述べるような形に制限することによってファイルに対するアクセスを矛盾なくコントロールしている。

4 リザーブ / デポジット / キャンセル

前節の4つの基本操作の組み合わせにより、リザーブとデポジットとキャンセルの機能を実現することができる。

リザーブは、対象ファイルの修正権の取得を行なった上で自プロジェクトへのコピーを行なう、という操作である。既に他の作業者によってリザーブされているファイルに対してリザーブを行なった場合は失敗する。従って1つのファイルをリザーブしているのは高々1人ということになる。

デポジットは、親プロジェクトへのコピーを行なった後で対象ファイルの修正権の放棄を行なう、という操作である。デポジットは、自分が既にリザーブしていたファイルに対してのみ可能である。

キャンセルは、修正権の放棄のみを行なう、という操作である。キャンセルは自分が既にリザーブしていたファイルに対してのみ可能である。これは、リザーブしていたファイルを親プロジェクトにコピーしたくなくなった場合に用いる。

これらリザーブ / デポジット / キャンセルの機能を使う限り同時編集の危険性は完全に排除される。しかし、実際のソフト開発の場においては、これだけでは不十分な面がある。

たとえば1人の作業者が比較的長い期間ファイルをリザーブし続けたとき、他の作業者がそのファイルを部分的に修正してテストを行なう、ということは不可能になってしまう。

LifeLineでは、そのためにファイルの二重化の機能が用意されている。

5 二重化

二重化は、自プロジェクトへのコピーのみを行なう、という操作であり、そのファイルに対する他の作業者のリザーブ及び二重化の有無にかかわらず、自由に行なうことができる。

二重化は親プロジェクトのファイルに対して、一時的な変更を行なってテストするといった場合に用いられ、一般的には二重化して修正したファイルを親プロジェクトにコピーすることはない。しかし、二重化して修正したファイルを親プロジェクトにコピーするという状況もありうるため、二重化しているファイルに対して、後から修正権の獲得を行なうことができるようになっている。

修正権の獲得を行う時点で他の作業者がそのファイルをリザーブしていると、修正権の獲得に失敗する。その場合は、ファイルがデポジットされるまで待たなければならない。

修正権の獲得に成功した場合、ファイルをデポジットすることが可能になる。しかし、ファイルを初めからリザーブしていた場合と異なり、二重化した後他の作業者によって親プロジェクトのファイルに修正が加えられている可能性がある。従って単にデポジットするとその修正が失われてしまうことになる。

LifeLineでは、このような状況でファイルをデポジットする場合、警告が与えられる。これにより作業者は、他の作業者によって加えられた修正と自分自身による修正を比較することによって、それらを矛盾しないようにマージした後でデポジットすることができる。

6 おわりに

LifeLineにおける情報のアクセス制御の機能について述べた。

現在、LifeLineライブラリは一部機能を除いて完成しており、既に複数のCASEツールに適用されている。

しかしながら、現ライブラリは汎用的でプリミティブな機能の集合であるため、今後CASEツールの共通基盤として利用されるためには現在提供している基本的な機能に対して特定の意味づけを与えたより高レベルなインタフェースを提供していく必要があると考えている。

参考文献

[1] Charles W.Krueger. The SMILE Reference Manual. *The GANDALF System Reference Manuals.* Carnegie Mellon University,1986.

[2] Evan W.Adams, Masahiro Honda, Terrence C.Miller. *Object Management in a CASE Environment.* Proc. of the 11th ICSE, May, 1989.

[3] 岸知二、入交晃一、坪谷英昭. CASE環境構築のためのファイル管理機能. 情処学会CASE環境シンポジウム, 1989年3月.