社団法人　電子情報通信学会
THE INSTITUTE OF ELECTRONICS,
INFORMATION AND COMMUNICATION ENGINEERS

信学技報
TECHNICAL REPORT OF IEICE.

関数の変更履歴と呼出し関係に基づいた開発履歴理解支援システム

中山　崇† 　松下　誠† 　井上　克郎†

† 大阪大学大学院情報科学研究科　〒 560–8531 大阪府豊中市待兼山町 1 番 3 号
E-mail: †{t-nakaym,matusita,inoue}@ist.osaka-u.ac.jp

あらまし　オープンソースソフトウェアの開発を行う際には, 開発されたプロダクトを管理する為にリビジョン管理システムが用いられる. リビジョン管理システムは過去の開発履歴を保持しており, それを利用して効果的な開発を行うことが出来ると考えられる. しかし開発が進むにつれてこれらの情報は膨大になる為, 必要な情報を的確に取得することは容易ではない. そこで本研究では, 蓄積されたソフトウェアの開発履歴を解析し, 開発者が必要とする開発履歴情報の検索・閲覧を行うシステム CREBASS を試作した. さらに実際のソフトウェア開発で用いられた版管理システムの開発履歴情報を用いて本システムの評価を行った. その結果, 本システムを用いることで, 開発者は開発履歴から有益な情報を入手することが可能となり, 現状の問題点が解決されることを確認した.

キーワード　リビジョン管理, クロスリファレンス, リポジトリ, 検索

Software Development Recognition Supporting System Using Revision History and Function Calling Relationship

Takashi NAKAYAMA†, Makoto MATSUSHITA†, and Katsuro INOUE†

† Graduate School of Information Science and Technology, Osaka University
1–3, Machikaneyama–cho, Toyonaka–shi, Osaka, 560–8351 Japan
E-mail: †{t-nakaym,matusita,inoue}@ist.osaka-u.ac.jp

Abstract　Revision control system (RCS) is used for managing software products efficiently. Usually, RCS saves all software products from the beginning of target software development which have lots of valuable information for future software development and it could be used for recognizing software development itself. However, these information tends to be larger as time have past, it would be hard to grasp what was happened. In this study, I implement a system which extract software development information, search the information, and show them via web browser. Also I applied this system to actual software development information, and evaluate system adequacy. As a result, I confirmed this system enables developers to retrieve more useful information.

Key words　Revision Control, Cross Reference, Repository, Search

1. ま え が き

オープンソースソフトウェアの開発規模の増大に伴い, その開発形態は多人数化, 分散化している. 大規模なオープンソースソフトウェア開発では, 複数の開発者が互いにソースコードを共有しながら同時に一つの開発作業に携わることが一般的になりつつある. またインターネットに代表されるネットワーク環境の発展に伴い, 分散した多くの開発者が異なる場所で開発作業を行うことも多い.

このように複雑化しているソフトウェアシステムを効率よく管理する為, 近年のオープンソースソフトウェア開発では, リビジョン管理システムを用いることが多くなっている. リビジョン管理システムは, プロダクトの開発履歴をリポジトリと呼ばれるデータベースに格納して管理する. リポジトリに含まれる履歴の中には, 将来の開発に活用することの出来る情報が多く蓄積されている.

ソフトウェアを再利用する際, これらの履歴情報を閲覧することによって, 以前の開発についてより深い理解が得られ, 開発の手助けになることが知られている [9]. しかし, 蓄積された膨大な情報の中から, 開発者が必要とする情報を的確に取得することは容易ではない. 例えば, ある機能の欠陥を修正する為に, 一見無関係に見える複数のファイルが同時に修正される, などという状況が考えられる. このとき, 後からその履歴を閲覧する開発者は, 修正されたファイルがどれであるかを知ることすら

困難である.

本研究では, この問題を解決する為のソフトウェア開発支援環境の構築を行う. 具体的には, リポジトリに蓄積されたソフトウェアの開発履歴を解析し, 開発者が必要とする開発履歴情報の検索・閲覧を行うシステムの試作を行う. 本システムは, 開発履歴情報の特定の為のリビジョン検索, 及びヒストリイベント検索, そしてソフトウェア開発の具体的な手法を閲覧する為のリビジョン関係を考慮した関数クロスリファレンサなどを持つシステムであり, これらの機能を提供することでオープンソースソフトウェア開発の支援を行う. 例えば, リビジョン検索やヒストリイベント検索を用いての修正箇所の特定や, 関数クロスリファレンサを用いての実装の詳細の理解における支援を行う.

また, 実際のオープンソースソフトウェア開発で用いられたリポジトリを用いて本システムの適用実験を行なった. 具体的には, 実際のソフトウェア開発で蓄積された開発履歴情報を用いて, 過去の開発情報の検索・閲覧及び理解が出来るか実験した. その結果, 本システムを用いることで, 開発者は有益な情報を入手することが可能となり, 現状の問題点が解決されることが確認できた. このことにより, ソフトウェア開発者の負担が軽減され, ソフトウェア生産性の向上が期待できる.

以降, 2 節では版管理システム CVS と既存の開発履歴閲覧システム, そしてその問題点について述べる. 3 節ではシステムの概要と設計について述べ,4 節ではそのシステムの実装について述べる.5 節では本システムに対して検証を行う. 最後に 6 節で本研究のまとめと今後の課題について述べる.

2. 版管理システムを用いたソフトウェア開発

本節では, オープンソースソフトウェア開発などで一般的に用いられている版管理システム CVS と, その中に蓄積された開発履歴を閲覧する為のシステムについて述べる. また既存の開発履歴閲覧システムが抱える問題点について指摘する.

2.1 オープンソースソフトウェア開発環境

オープンソースソフトウェア開発では, 各開発者がそれぞれ分散して並列的に開発作業を行うことが可能である. その一方で, 開発中のソースコードやドキュメントなどのプロダクトを広く公開するため, それらの管理を行う必要がある. そこで, オープンソースソフトウェア開発に参加する開発者は, オープンソースソフトウェア開発環境と呼ばれる環境の中でプロダクトの管理を行う. オープンソースソフトウェア開発環境の構成例を図 1 に示す.

オープンソースソフトウェア開発環境は一般に複数の既存システムから構成される. 図 1 の構成例の場合, ソースコードやドキュメントなどのプロダクトは, 版管理システム [6] の一つである CVS(Concurrent Versions System) [3] [7] [12] を用いて管理される. また CVS に蓄積された開発履歴情報は CVSWeb などの開発履歴閲覧システムを用いて閲覧することが出来る. このようにして開発されたプロダクトは, rsync や ftp を利用して, 各開発者に複製, 配布される. また開発者間の意思疎通の手段として, 電子メールやメーリングリストが用いられる. その内容はアーカイブとして保存され,WWW(World Wide Web) を用いた検索エンジンによって自由に検索や閲覧が可能である.

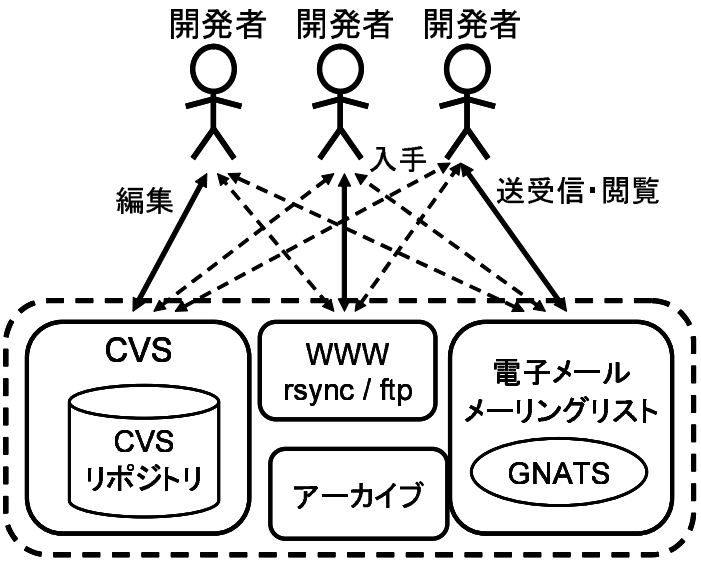

図 1　オープンソースソフトウェア開発環境の構成例

以下では, これらのシステムの中からオープンソースソフトウェア開発で広く用いられる版管理システム CVS と開発履歴閲覧システムについて説明する.

2.2 オープンソースソフトウェア開発で用いられるシステム

- 版管理システム CVS

CVS は UNIX 上で動作するシステムとして構築された版管理システムであり，近年最も良く使われるシステムの１つである．CVS は複数のファイルを処理でき, またリポジトリを複数の開発者で利用することも考慮し，開発者間の競合にも対処可能となっている．さらに，ネットワーク環境を用いた分散開発を行うことも出来る為, Apache [1] や FreeBSD [10] 等, オープンソースソフトウェア開発やグループウェアなどで用いられている.

次に CVS を使う上で良く用いられる用語について説明する. CVS では, プロジェクトの開発履歴をリポジトリというデータベースにリビジョンの列として格納する. このリビジョンとはプロジェクトのある時点でのスナップショットを表すものである. 開発者はチェックイン, チェックアウトという作業を行って, リポジトリから特定のリビジョンを取得したり, 変更内容をリポジトリに反映したりといったことを行う. チェックイン時にはログメッセージというコメントを添えることで, そのチェックインを行う意図をはっきりさせる．また, あるリビジョンからバグ修正などの為に新たにリビジョン列を派生させることも出来る. この派生したリビジョン列をブランチという. このようにして得られるリビジョン列は一般に木構造をなしている為, リビジョンツリーと呼ばれる.

- CVSWeb

CVSWeb [5] は CVS リポジトリ内に保存されている開発履歴情報を Web ブラウザ経由で視覚的に閲覧できるシステムである.

このシステムは, プロジェクト内のファイル階層の閲覧や, 各

ファイルのチェックインの履歴の確認, 各リビジョンの内容の表示, そしてあるファイルの任意のリビジョン間の差分を色付きで表示する機能なども持っている.

- ViewCVS

ViewCVS [11] は CVSWeb と同様に CVS リポジトリ内の開発履歴情報を Web ブラウザ経由で閲覧できるシステムである.

このシステムは CVSWeb の機能に加え, 正規表現やワイルドカードを用いたリビジョン検索や正規表現を用いたファイルの検索, そして CVSGraph [4] を用いたリビジョンツリーの視覚的表示などが行える.

2.3 問 題 点

版管理システムと履歴閲覧システムを用いてソフトウェアの開発作業を行う場合, 以下のような問題点がある.

- リビジョン検索機能を持たない

版管理システムを用いた開発を行う時, 過去のソースコードや変更内容を, 新たな開発に再利用するといったことが行われる. このとき, 開発者は自分の開発内容に関連するリビジョンをリポジトリから探し出す必要があるが, プロダクトの過去の開発状況を熟知していない限り, 莫大な履歴情報から開発者が欲するリビジョンの情報を即座に取得するのは非常に困難である. リビジョン検索機能がないと過去の開発履歴情報を利用することが困難になると考えられる.

- 開発された時期を考慮した関数の参照元と定義を得るのが困難

版管理システムを用いた開発を行う時は, 必ずしも最新版のソースコードを編集することになるとは限らない. 例えば, 過去の安定版のソースで見つかったバグを修正する時は, 最新版のソースを編集するのではなく, 安定版のリビジョンからバグ修正用のブランチを派生させて作業を行うことが多い. このような状況下では過去のリビジョンのソースコードを理解する必要があるが, そのリビジョンから参照している関数の, その時点での関数定義を取得することは既存のシステムでは不可能である. このことは過去のソースコードの理解の大きな障壁になっていると考えられる.

- 開発履歴の追跡を単一ファイルでしか行えない

一般的に, システムの 1 つの機能を複数のソースファイルを用いて実装することは珍しくない. このような機能の開発作業の流れを理解するには, その機能を実装している全てのファイルの開発履歴を同時に追跡する必要がある. しかし既存のシステムでは単一ファイルの開発履歴を追うことしか出来ず, 開発作業におけるファイル間の関係を把握することが困難である.

これらの問題点は開発者への負担を増加させ, 開発効率の低下を招いていると考えられる. そこで以上の問題点を解決することを目的とした, 開発履歴閲覧システム CREBASS を設計, 試作した.

3. CVS リポジトリ閲覧・検索システム CREBASS

本節では, 試作した開発履歴理解支援システム CREBASS (Cvs REpository Browse And Search System) について説明する.

図 2 に CREBASS の概要を示す.

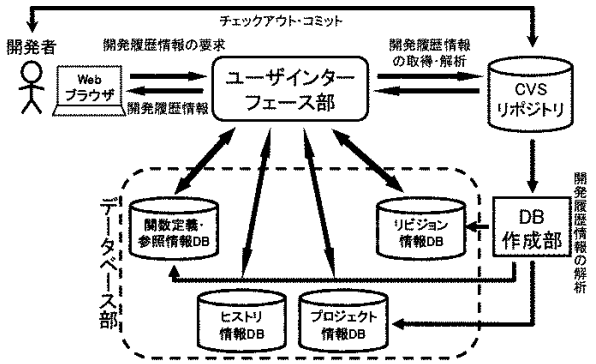

図 2 CREBASS の概要

3.1 システムの概要

本システムは, 前節で述べた問題点を解決する為に, 以下の機能を持つ.

- リビジョン検索機能

  各リビジョンの更新作業に関する情報を指定した条件の元で検索する機能である.

  検索条件としては, リビジョンの更新日時, 更新作業者, 更新作業を象徴するキーワード, リビジョンが所属するブランチ名, そしてファイルが存在するディレクトリ名を指定することが出来る.

  検索を実行すると指定した条件に合致するリビジョンの情報が列挙される. 各リビジョン情報にはファイル名, リビジョン番号, 更新日時, 更新作業者, 所属ブランチ名, ログメッセージ, そしてキーワードが記載されている.

  この機能を用いることで, 自分の興味がある開発履歴をリポジトリから容易に取り出すことが可能になる. 例えばあるディレクトリ以下のファイルに対して行った, ある開発者の更新作業の検索といったことが行える.

- リビジョン関係を考慮した関数クロスリファレンス機能

  ソースコードを閲覧中に関数を参照している箇所をクリックすることで, その関数の, 閲覧中のリビジョンの時点での定義元へとジャンプする機能である.

  過去のリビジョンのソースコードを閲覧中に, そこから参照している関数の実装がどのようになっているか調べたいことがある. しかしその関数の仕様が現在とその時点で等しいとは限らないので出来る限り時間的に対応する関数定義を得たい. この機能を用いることで既存システムでは多大な労力がかかるこの操作をリンクのクリックだけで行うことが出来る.

- プロジェクトデータの変遷の折れ線グラフによる表示機能

ファイルの総行数, 編集量, 累計 commiter 数の変遷を折れ線グラフにして表示する機能である.
入力として対象のファイル名とブランチ名を与えることで, そのブランチ上における対象ファイルの変遷を折れ線グラフとして閲覧することが出来る.
入力にはファイル名だけではなくディレクトリ名を与えることも出来る. この場合は指定されたディレクトリ以下の全ファイルの値を足しあわせたものがグラフ化の対象となる. これによって複数のファイルの変遷をまとめてみることが可能となる.
また 1 つのグラフ上に複数の対象を同時にグラフ化することも出来る. これによって複数対象間の開発履歴の違いを確認することが可能となる.

- CVS history 情報の検索機能

CVS には history というコマンドがあり, それを用いることによって過去, リポジトリに対して起きたイベントのヒストリを得ることができる. ここで言うイベントとはファイルのチェックイン, チェックアウト, タグ付けなどのリポジトリに対するアクションのことを指す. 本システムではこのイベント情報をデータベース化し, 検索する機能を持つ.
検索条件としては, 検索するイベントの種類, イベントを起こしたユーザ, 検索する期間, そして検索対象とするディレクトリを指定することが出来る.
検索を実行すると指定した条件に合致するヒストリイベントの情報が列挙される. 各ヒストリイベント情報にはイベントの種類, イベントが起きた日時, イベントを起こしたユーザ, そして各イベントに特有の情報が記載されている. また複数のイベントが同時に発生したと考えられるときはそれらのイベントをまとめて一まとまりのヒストリイベントとして表示するといったことも行う.

これらの機能の他に既存のシステムにもあるような, ディレクトリ構造の表示機能, 任意のリビジョン間の差分表示機能, ファイルログ表示機能, そして各リビジョンの内容の表示機能を持つ.

3.2 データベース

本システムでは以下の 4 つのデータベースを用いることによりユーザに有用な情報の提示を行う.

- 関数定義・参照情報データベース

あるリビジョンの何行目に何の関数の定義があるのか, また何行目で何の関数を参照しているかをデータベース化したもの.C 言語のソースコードを表示する際のクロスリファレンスの構築に用いる.

- リビジョン情報データベース

各リビジョンのチェックインの情報をデータベース化したもの. リビジョン検索の際に用いる.

- プロジェクトデータベース

各ファイル, リビジョンの総行数, 編集量, 累計 commiter 数をデータベース化したもの. これらの情報についての折れ線グラフの描画の際に用いる.

- cvs history 情報データベース

リポジトリに対してこれまでに起きたヒストリイベントをデータベース化したもの. ヒストリイベントの検索の際に用いる.

3.3 ユーザインターフェース

この部分はユーザと各データベースを結ぶインターフェースの役割を果している. 開発者から要求を受けると CVS リポジトリやデータベースと連結し, 要求されたデータを表示する.

- ディレクトリ構造表示部

あるディレクトリの中にどのようなファイルやサブディレクトリが存在するかを確認できる.

- ファイルログ表示部

あるファイルのコミットログの履歴を確認することが出来る.

- 差分表示部

あるファイルの 2 つのリビジョン間の内容の差分を表示する. 前のリビジョンからどの行が削除され, どの行が加えられ, そしてどの行がどのように変更されたかを判別しやすいようにそれぞれの場合について別の色で背景を塗り, わかりやすいように表示する.

- コード表示部

あるファイルのあるリビジョンの内容を表示する.
そのファイルが C 言語のソースファイルであった場合は, ここにリビジョン関係を考慮した関数クロスリファレンス機能が加わる. 関数を参照している箇所にはリンクが張ってあり, そこをクリックするとそのリビジョンのチェックイン時点での関数定義元へとジャンプする.

- リビジョン検索部

指定した条件に合致するリビジョンを検索する. 指定する条件の種類には更新日時, 更新作業者, キーワードがある. また検索する範囲をディレクトリ名やブランチ名を指定して絞ることも出来る.

- プロジェクトデータ閲覧部

ファイル, もしくはディレクトリ下のファイルの総行数, 編集量, 累計 commiter 数の折れ線グラフを生成して表示する. 複数のファイル, ディレクトリのグラフを同時に表示してそれぞれを比較することも出来る. またブランチ名を指定することでグラフ化の対象をそのブランチのみに絞ることも出来る.

- cvs history 表示部

過去にリポジトリに起きたイベントを検索, 表示する. 検索時にはイベントの種類, イベントを起こしたユーザ名, 期間, そしてディレクトリを指定することが出来, その条件に合致するイベント情報群が時系列で表示さ

れる. このとき同時に起きたと考えられる複数のイベント情報群は 1 つにまとめて表示される.

4. CREBASS の実装

本節では, これまで述べた開発履歴理解支援システム CREBASS の実装を行なった. 開発には Java を用い, コードサイズは全体で約 9100 行となった. 開発環境は以下のとおりである.

- CPU:Penitum4 1.5GHz
- RAM:512MB
- OS:FreeBSD 4.9-STABLE
- データベース:GNU GDBM 1.8.3
- WEB サーバ:Jakarta Tomcat 5.0.16
- JDK 1.4.2

4.1 データベース

必要な情報を容易に取得できるように, あらかじめ CVS リポジトリを解析してデータベースを作成しておく.

CVS のリポジトリは, プロジェクト中の各ファイル毎に作成された RCS 形式のファイルの中に過去の変更履歴が保存されている. 本システムではこの RCS ファイルを解析することで必要な開発履歴情報の抽出を行う.

関数定義・参照情報データベースでは関数の定義位置情報などの, リビジョン関係を考慮した関数クロスリファレンサを構築する上で必要な情報をデータベース化し, リビジョン情報データベースでは各リビジョンの情報とリビジョン検索で用いる情報をデータベース化する. またプロジェクトデータベースでは各種情報の折れ線グラフの作成に必要な情報をデータベース化し, ヒストリ情報データベースではヒストリ検索に必要な情報をデータベース化する.

4.2 ユーザインターフェース

ユーザインターフェース部はユーザからの要求を受けて, リポジトリやデータベースから情報を取り出し, 表示する.

例えばリビジョン検索部 (図 4) では, リビジョン情報データベースと連結してリビジョン情報の検索, 表示を行う. 同様にヒストリ情報検索部 (図 6) やプロジェクトデータ閲覧部 (図 5) でもヒストリ情報データベースやプロジェクトデータベースと連結して各機能を実現している.

また, ディレクトリ構造表示部 (図 3), ファイルログ表示部, 差分表示部は CVS リポジトリから必要な情報を取り出して開発者に対して提示する. コード表示部も, ファイルの中身や各リビジョンの情報はリポジトリから取得するが, コード中に関数を参照している箇所が存在する時は関数定義・参照情報データベースと連結して, リビジョン関係を考慮した関数クロスリファレンス機能を実現する.

5. 適 用 実 験

本節では実際にシステムを動作し, 得られた結果について考察する.

5.1 実 験 対 象

CREBASS 内に保持するデータとして,OpenSSL [8] の CVS リポジトリとそこから得られるデータを用いた.

図 3 ディレクトリ構造表示画面

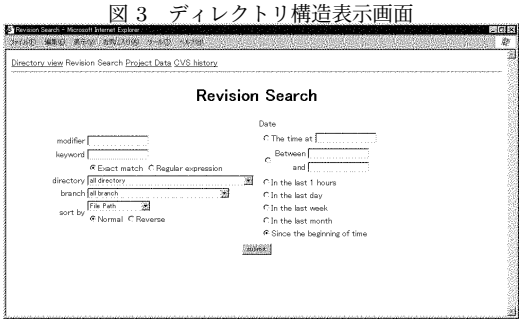

図 4 リビジョン検索画面

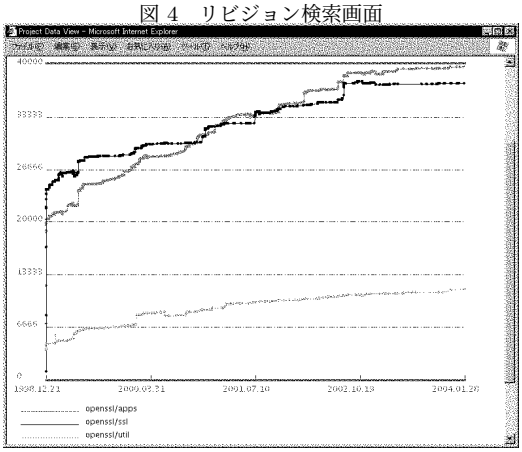

図 5 総行数の折れ線グラフ

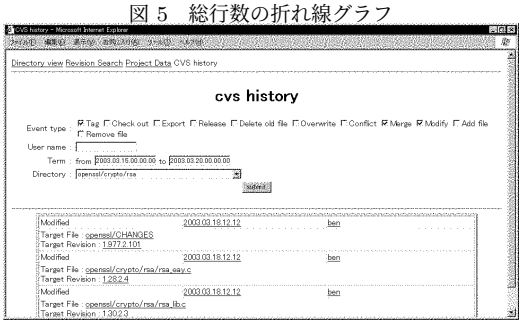

図 6 cvs history 表示画面

5.2 実験の内容

2003 年 3 月 17 日に OpenSSL v0.9.7a と v0.9.6i に対するパッチのリリースが行なわれた. このパッチは, ホストの秘密鍵が流出する可能性があるという,RSA blinding に関する脆弱性を修正する為のものである. 今, 利用者はこのパッチによるソフトウェアへの変更点の詳細を確認したいものとする.

まず利用者はこのパッチに関する変更箇所を確認する為に, リビジョン検索機能を用いて,2003 年 3 月 15 日から同 20 日までにチェックインされ, かつキーワード RSA を持つリビジョンを検索した.

検索を行なった結果,7 つのリビジョンが条件に一致するものとして表示された. 表示された結果を見ると, そのうち 3 件が ben というユーザによって同時にチェックインされたリビジョン, 残り 4 件が bodo というユーザによって同時にチェックインされたリビジョンであることがわかった.

チェックインの内容を確認する為にログメッセージを確認すると, ben のチェックインには”Turn on RSA blinding by default.” と書かれていた. また ben がチェックインを行なった openssl/CHANGES ファイルを読むと, タイミング攻撃を回避する為に,RSA blinding をデフォルトでオンにするという記述が確認できた. 以上のことから ben のチェックインは RSA blinding に関する脆弱性を修正する為のものであることが確認できた.

次に RSA blinding をデフォルトでオンにする為の具体的な修正内容を確認する為に, チェックインされた他の 2 つのファイルの差分を確認することにした. すると新たに関数とマクロが定義されており, それらを用いて RSA blinding のデフォルトでの機能使用を行なっていることが確認できた.

しかし新たに定義された関数内から参照している関数の実装についてはこのファイルからはわからず, また同時にコミットされたファイルからもその記述は発見できなかった. そこでチェックインされたソースコードをコード表示部で表示したところ, 両関数の定義へのリンクが張ってあったので, それをクリックすることでその関数の定義を確認することができた.

5.3 考　　察

今回の適用実験から, 利用者は日時やキーワードを用いてリビジョン検索を行うことで自分の知りたい開発履歴情報を検索できることがわかった. またリビジョン関係を考慮した関数クロスリファレンスシステムを用いることで, あるリビジョンのソースコード内で参照している関数の定義を調べることが出来, ソフトウェアの実装の理解を深めることが出来ることがわかる.

ここで既存のシステムを用いて今回と同じ実験を行うことを考える. CVSWeb ではリビジョンの検索が出来ない為, プロジェクトの詳細を理解している人でない限り, 該当する変更箇所がどこであるかを瞬時に捜し出すことは困難である. 一方,ViewCVS や Bonsai [2] などはリビジョン検索機能を持っているが, ログメッセージに関する検索を行うことが出来ないので, プロジェクトに精通していないユーザは知りたいトピックに関するリビジョンを絞り込むのに困難を要すると思われる. また, 該当するリビジョンを検索できたとしても, そのリビジョンがその時点で参照している関数の定義を取得する方法が無い為, 対応する関数定義を得る為には多大な手順が必要であったと考えられる.

しかし, 現在の CREBASS では不十分な機能も存在する. 例えばリビジョン検索のときに用いるキーワードの抽出は必ずしも完全とはいえず, キーワードだけで目的のリビジョンを正しく検索できない. また, 本システムでは関数の定義, 参照関係の導出を, 関数名の一致のみから判断している. この結果, 複数のファイルで同名の関数が定義されていたときに, 関数の参照先を一意に特定することが出来なくってしまう. 現在は, 複数の定義が見つかった場合にはユーザがどの定義を閲覧するか選択できるようになっているが, この選択で必ずしも正しい定義元を選択できるとは限らず, 間違った関数定義情報を取得してしまう恐れがある.

6. ま と め

本研究では CVS リポジトリに蓄積されたリビジョン情報を解析して, 開発履歴情報の閲覧・検索を行ない, ユーザの開発履歴理解を支援するシステムを作成した. 本システムは各ファイルの開発履歴や任意のリビジョン間の差分表示の他に, リビジョン関係を考慮した関数クロスリファレンス, 更新日時・更新作業者・キーワードによるリビジョン検索, 任意のファイル, もしくはディレクトリ下のファイル群の開発情報の遷移を折れ線グラフで視覚的に確認する機能, 過去にリポジトリに対して起こったヒストリイベントを検索・表示する機能などがある. そして実際のソフトウェア開発で蓄積された開発履歴情報を用いて適用実験を行ない, ユーザが必要とする開発履歴情報の閲覧・検索が行なえることを確認した.

今後の課題としては, リビジョン検索時に用いるキーワードの抽出手法の改善, 関数クロスリファレンサによる関数の参照先特定手法の改善, より大規模なリポジトリによる適用実験, そして実行速度, 及びスケーラビリティの向上が挙げられる.

文　　献

[1] The Apache Software Foundation, Apache Projects, http://www.apache.org/.

[2] Bonsai, http://www.mozilla.org/bonsai.html.

[3] Brian Berliner, “CVS II:Parallelizing Software Development”, In USENIX Association, editor, Proceedings of the Winter 1990 USENIX Conference, pages 341–352, Berkeley, CA, USA, 1990.

[4] CvsGraph, http://www.akhphd.au.dk/~bertho/cvsgraph/.

[5] CVSWeb, http://stud.fh-heilbronn.de/~zeller/cgi/cvsweb.cgi/. http://www.freebsd.org/projects/cvsweb.html/.

[6] Jacky Estublier, “Software Configuration Management: A Roadmap”. The Future of Software Engineering in 22nd ICSE, pp.281–289, 2000.

[7] Karl Fogel, “Open Source Development with CVS”, The Coriolis Group, 2000.

[8] OpenSSL, http://www.openssl.org/.

[9] Peter H. Feiler, “Configuration Management Models in Commercial Environments”, CMU/SEI-91-TR-7 ESD-9-TR-7, March, 1991.

[10] The FreeBSD Project, The FreeBSD Project, http://www.freebsd.org/.

[11] ViewCVS, http://viewcvs.sourceforge.net/.

[12] 鯉江英隆, 西本卓也, 馬場肇, “バージョン管理システム (CVS) の導入と活用”, SOFT BANK, December, 2000