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.

閲覧状態復元機能付き Web ブラウザの試作

今枝　誉明†　　早瀬　康裕†　　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-imaeda,y-hayase,matusita,inoue}@ist.osaka-u.ac.jp

あらまし　ソフトウェア開発を行う際には，開発状況やプロダクトを閲覧する為のシステムが広く使われている．こういったシステムではユーザーインターフェースとして Web ブラウザが用いられていることが多いが，それぞれのシステムは独立しており，複数のシステムにまたがる情報を閲覧する場合，情報間の繋がりは閲覧者自らが管理しなければならない．そこで本研究では，閲覧した内容を逐次閲覧履歴として保存することにより，ある過去の時点における閲覧状態を復元する機能をもつ Web ブラウザの試作を行った．さらに，他の利用者の閲覧した内容に基づいた情報推薦機能を実装した．試作した Web ブラウザを使用して，あるソフトウェアシステムの調査を行った結果，効率的に閲覧でき，その有用性を確認した．

キーワード　ソフトウェア開発ツール，Web ブラウザ，協調フィルタリング

A Prototype of Web Browser with Resuming Browsing State

Takaaki IMAEDA†, Yasuhiro HAYASE†, 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-imaeda,y-hayase,matusita,inoue}@ist.osaka-u.ac.jp

Abstract　Lots of software development supporting systems, such as process and product monitoring system, are used in development environments. These systems employ a web system. Since each system is independent, it took the developers many troubles with managing the browsing state of each system. In this paper, we implement a prototype of web browser providing functions that browsing states are saved as browsing history point by point, and specifying a certain point in the history lead users to the past state of the browser. Moreover, system could suggest the next preferrable page using histories. Through an experimentation of prototype browser, it is confirmed that the browser helps browsing more effectively.

Key words　Software Development Tool, Web Browser, Collaborative Filtering

1. はじめに

近年の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．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ともない，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．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ため，近年のソフトウェア開発では，プロダクトの開発状況を閲覧したりソフトウェア部品を検索したりするためのシステムを用いている．これらのシステムには，プロダクトの変更履歴やバグ情報などが蓄積されており，蓄積された過去の開発情報を理解することで，ソフトウェア開発や再利用の際の手助けとなることが知られている [1].

一般的に，これらのシステムはそのユーザーインターフェースとして Web ブラウザを用いており，開発者は自由に閲覧することが可能である．しかし，これらのユーザインターフェースはそれぞれ単独のシステムとして独立しているため，開発者は必要な情報を得るために複数のインターフェースを同時に使い分ける必要がある．このとき，各システム間で関連のある情報を同時に見ることが多いが，システム間の連係が無いため，開発者自身が全体の繋がりを管理しなければない．また，閲覧作業が長くなり，閲覧すべき問題が多岐に渡ってくると，こうしたシステム間の関連情報のみならず，本来解決すべき大元の問題までも見失いかねない．

そこで本研究では，Web ブラウザを用いるソフトウェア開発支援システムを対象として，複数のシステムを同時に利用する際に必要とされるインターフェースを持つ Web ブラウザの試作を行う．具体的には，まずブラウザでの閲覧状態を逐次，履

歴として保存し，またある過去の時点における閲覧履歴を指定することにより，いつでもその時点で閲覧していた内容を復元できる機能を実装した．また，閲覧ページの親子関係をツリー状のグラフで表示するインターフェースを付加した．さらに，複数の利用者間で閲覧履歴を共有することにより，他の利用者が閲覧した内容に基づいて，これから閲覧することが望ましい情報の推薦機能を実装した．

また，作成したシステムの有効性を確認する為に，実際のソフトウェア開発に関連する情報を，作成したシステムを用いて調査した．その結果，調査途中で過去に一度閲覧した項目に関してさらに調査を行う際，保存された過去の閲覧情報を読み込むことで，過去に行った手順を繰り返すこと無く，調査を再開できることを確認した．また，それまでに蓄積された閲覧履歴を分析することで，有用であるにも関わらず，開発者が取得できなかった情報が推薦されることを確認した．このことにより，ソフトウェア開発者の負担が軽減され，ソフトウェア理解の効率化が期待できる．

以降，2. 節では，ソフトウェアとその開発環境について説明し，その問題点について述べる．3. 節では，試作したブラウザの概要と設計について述べ，4. 節ではその実装について述べる．5. 節では本システムを用いた場合に解決する問題の例を示す．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. ソフトウェア開発環境とその問題点

2.1 ソフトウェア開発環境

近年のソフトウェア開発は，その規模の増大に伴い開発形態が多人数化しており，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多くなってきている．その一方で，開発中のソースコードやドキュメント等のプロダクトの進捗状況を把握するため，それらの管理を行う必要がある．そこで開発者はソフトウェア開発環境の中でプロダクトの管理を行う．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．

ここでは，これらのシステムの中から，ソフトウェア開発で広く用いられている版管理システムや電子メール，バグ追跡システムなどについて説明する．

版管理システム　版管理システムは，プロダクトに対して施された追加・修正作業を履歴として蓄積し，また蓄積した作業履歴を開発者に提供する，という役割を提供する．また，版管理システムには数多くの関連ツールが開発されており，例えば，CGI を利用した Web インターフェースならば，履歴内に存在するファイル一覧や，各時点でのデータ等を既存の Web ブラウザで閲覧することが可能である．

メーリングリスト　開発者が分散して作業を行う場合，互いの意思疎通の手段として電子メールによるメーリングリストシステムが広く利用されている．また，電子メールが膨大な量になると，アーカイブとして管理される．さらに，WWW(World Wide Web) を用いて，アーカイブの中から電子メールによる議論の内容を検索するシステムも存在する．

バグ追跡システム　バグ追跡システムはプロダクトのバグ報告や機能の追加要求を管理するためのシステムで，開発者が投稿したバグ情報はデータベースに蓄積される．これらのシステムも，Web インターフェースを通してバグ情報の投稿や抽出を行うことができ，さらに過去の解決された事例を閲覧することで，自身の問題解決に役立てることが可能である．

クロスリファレンサ　クロスリファレンサとは，ソースコードを解析して関数の定義場所を抽出するためのツールである．ソースコードを HTML として出力することが可能なものもあり，関数呼び出しが書かれている場所には，定義場所へのリンクが埋めこまれる．そのため，出力されたソースコードを Web ブラウザを用いて調査することが容易にできる．

2.2 問　題　点

このように，近年のソフトウェア開発ではネットワーク上に存在する開発環境にてプロダクトの管理が行われる．そして開発環境を構成する各システムは，一般的に WWW を通じて利用者に情報を提示する．そのため，開発者は現在開発しているプロダクトの情報を，Web ブラウザを用いて入手することが多い．

しかし，これらの WWW を利用した情報提示システム群には「システム間の連携不足による同時閲覧情報の欠損」,「複雑なリンク関係による閲覧ページ情報の欠損」という 2 つの問題が存在する．以下これら 2 つの問題点について述べる．

2.2.1 システム間の連携不足による同時閲覧情報の欠損

ソフトウェア開発では，開発管理を効率良く行うことを目的として，SourceForge や SourceCast，OSDL(Open Source Development Lab.) 等のサービスが提供されている．これらのサービスでは，電子メールや会議システム等の汎用的な CSCW(Computer Supported Cooperative Work) ツールや，その内容を記録したアーカイブ，WWW，版管理システム等，多くのシステムをまとめて開発者に提供する．

これらの各システムは独立して存在しているにも関わらず，それぞれが保持している内容間には関連が存在する．例えば，バグ追跡システムに登録されたバグ情報と，メーリングリストで交されたそのバグに関する議論とは，密接に関連している．そして開発者は 2 つのウィンドウを用いて，これら関連する情報を同時に閲覧することも頻繁に行う．

しかし，従来のブラウザでは，1 つのウィンドウ単位での履歴のみを管理しているため，複数のページを同時に見ていたという情報を保持することができない．そのため，これら複数のページを見ていた状態を復元したい場合には，利用者が自らそれらページの URL を記憶しておかねばならない．

2.2.2 複雑なリンク関係による閲覧ページ情報の欠損

一般に，ソフトウェア開発においては大量のプロダクトが存在する．そのため，プロダクト情報を提供するシステムでは，プロダクトおよびそれらを結びつけるリンク構造もまた大規模なものになる．そのため，リンクを辿って調査を進めている内に，以前に見た重要な情報を見失ってしまう危険性が存在する．

例えば，クロスリファレンサを用いて，main 関数から始まる一連の処理の調査をしている以下の状況を考える．

（1） main 関数を含んでいるページ A を表示している

（2） main 関数から最初に呼ばれる関数「init」をもつペー

ジ B に遷移する

（3） 何段ものリンク構造を辿り，データベースの初期化を行う関数をもつページ M に遷移する

（4） ブラウザの「戻る」を繰り返し，ページ B を経由し，ページ A に戻ってくる

（5） main 関数で，init 関数の次に呼ばれる関数を含むページ C に遷移する

（6） さらに何段かページを辿り，ページ N に遷移する

しかし，6 において，初期化したデータベースの内容を参照する必要が生じた．

このとき一般的なブラウザでは，そのヒストリとして現在見ているページに到達する際に直接経由してきたページのみしか保持されておらず，ページ B から始まる履歴も保持されていない．そのためページ M に再び辿りつくためには，ページ A からどのようなリンクを辿っていったかを思い出しながらリンクを探索しなければならず，リンクが膨大な場合には途中で辿れなくなってしまう危険がある．

このような状況に対しては，一般的なブラウザでは，ページ M に対してあらかじめブックマークをすれば良い，という方針で対応している．しかし現在調査している事柄に詳しくない閲覧者にとっては，どのページがブックマークすべき重要なページで，どこがそうでないかを判別することは非常に難しい．このように情報を閲覧している途中に以前に見た情報を参照したいという要求に，従来のブラウザでは十分応えているとはいえない．

3. 閲覧情報

これまで述べた問題点を解決するために，本研究では，ブラウザ自身に「どのページを」,「いつ」,「どのような形で」閲覧しているかという情報を保持することにより，利用者の要求に応じて任意の時点での状態を復元させる，という方法を用いる．本節では，そのための閲覧情報のモデルとその利用法について説明する．

3.1 閲覧情報モデル

3.1.1 同時閲覧状態とその履歴

本研究では，ある時点における Web ブラウザのウィンドウの状態を閲覧状態とする．そして，表示されているウィンドウに対して，移動や削除などの変化が加えられると，各ウィンドウの状態をその時点での閲覧状態として記録する．閲覧状態は，具体的には以下の情報から構成される．

- 時刻
- 直前の閲覧状態から変化した原因
  - ページ移動
  - ウィンドウの開閉・サイズ変更・移動
- 変化が起こったウィンドウ
- 変化後の各ウィンドウに関する以下の情報
  - 表示しているページの URL とタイトル
  - 位置・サイズ・スクロール位置
  - ツリーノード ID
- キーワード・リスト

ツリーノード ID に関しては，3.1.2 節で説明する．

キーワードとは，閲覧状態を特徴付ける単語であり，ウィンドウが表示しているページや，直前に表示していたページから抽出する．具体的には，ウィンドウが現在あるページを表示しているとき，そのページを記述している HTML 文書から以下の単語を抽出してキーワードとする．

- タイトル
- a 要素の name 属性の値
- meta 要素の content 属性の値 (name 属性が keywords か description である場合)
- リンク元のページのアンカーのテキスト (リンクを辿って表示されたページである場合)

各ウィンドウから抽出したキーワードをまとめて閲覧状態に対応付け，キーワード・リストとする．

そして，閲覧状態を時系列順に並べた列を閲覧状態の履歴とする．すなわち，その構造を図示すると，図 1 のようになる．

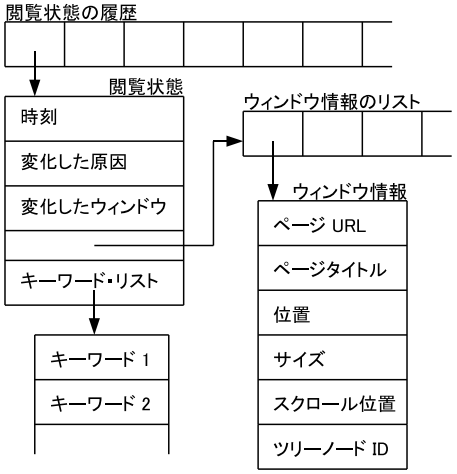

図 1 閲覧状態の履歴の構造

3.1.2 閲覧ページ構造情報

プロダクト情報の複雑なリンク関係により，もう一度見る必要が出てきたページがブラウザのヒストリに残されていない，という問題が生じる．

本研究では，例えば，2.2.2 節で挙げたようなページ遷移を行った場合，そのリンク構造を図 2 のようなツリー構造として捉え，それを表示するという方針をとる．すなわち，ページのリンクを開くときに，リンク元のページを親，リンク先のページを子とする構造を構築する．頂点がページで，ページの親子関係が頂点の親子関係に対応する．

そして，ツリーの各頂点には URL を記憶させ，任意のページを復元できるようにする．また，ツリーの各頂点には，その頂点を特定するための ID を持たせる．この ID を「ツリーノード ID」と呼ぶ．図 1 のように，ツリーノード ID をウィンドウ情報の 1 つとして保持させることで，あるウィンドウが現在表示しているページは，ツリー上のどの頂点に対応しているかを特定する．それにより，そのウィンドウ上でリンクを辿った場合，どの頂点の子とすればよいかが分かる．

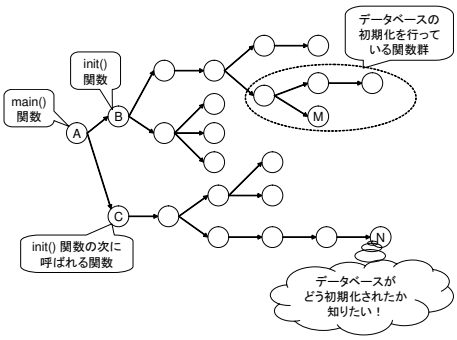

図 2　ページ遷移のツリー構造

3.2　閲覧情報を利用したページ推薦

さらに本研究では，蓄積された閲覧履歴情報を活用することで，有用と思われる情報の推薦を行うことを考える．そこで，協調フィルタリングの手法を用いて，他の類似した閲覧履歴を基に情報の推薦を行う．これにより求められた推薦ページ群は，その推薦値とともに利用者に提示される．

本節では，蓄積された利用者の閲覧履歴に対して協調フィルタリングを適用することで，ページの推薦を行う手法を説明する．本研究では，GroupLens [2] で提案された利用者間の相関を基にした手法をもとに，さらに相関係数の求め方に拡張を加えた推薦手法 [3] を用いる．手法の大まかな流れは以下の通りである．

1．利用者の評価の取得　協調フィルタリングでは，利用者の評価を利用してアイテムの推薦を行う．そのために利用者から各アイテムに対する評価を取得する必要があるが，本研究で用いる手法では，利用者がページを参照した時に暗黙的に投票したとみなし，そのページに対して評価値 1 で投票したとする．

2．相関係数の算出　協調フィルタリング手法においては，推薦を受ける利用者と似たような行動をとっている利用者を推定するために，利用者の類似度を相関係数として計算する必要がある．本手法では 2 人の利用者のいずれかが評価したページ，及びどちらも評価していないページを基に相関係数を求める．また未評価のページの評価値は 0 として計算を行う．

3．推薦値の算出　一般的な協調フィルタリングにおいては，推薦を受ける利用者と似たような行動をとっている利用者の評価を基に，各アイテムに対して，そのアイテムの推薦値を計算する．実際には，ある未評価のページに対する推薦値は，対象となる利用者がそのページに対して投票するであろう評価値の推定値として求めることができる．各ページの推定値は，それぞれの利用者との相関係数を重みとした，他の利用者のそのページへの評価値の加重平均により表される．

4．推薦の実行　求めた推薦値を基に，利用者に対してページの推薦を行う．ただし，利用者が既に参照したページに関しては，利用者はそのページについては既知であるとみなし，推薦対象から除外する．

4.　Web ブラウザの試作

前節で述べた閲覧状態の履歴を扱う Web ブラウザの試作を行った．本ブラウザは図 3 のように，表示部と閲覧履歴管理部からなる．

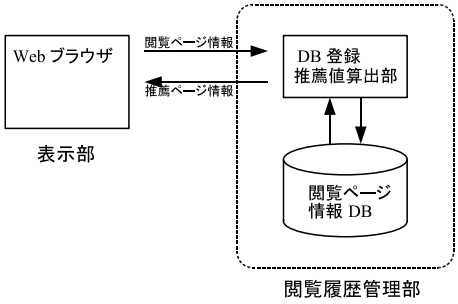

図 3　システムの構成

表示部は，MDI(Multiple Document Interface) とタブインターフェースを併用した Web ブラウザであり，ユーザインターフェースを提供する．

閲覧履歴管理部は表示部で閲覧した Web ページの情報を収集する．また，収集した情報を用いて，推薦する Web ページを決定し，表示部に送る．

以下，表示部と閲覧履歴管理部について説明する．

4.1　表　示　部

表示部の実装には，C#(.NET Framework v1.1) を用いて記述した．Web ページの表示および解析には，Microsoft 社の Web Browser コントロールを使用している．

表示部は，MDI とタブインターフェースを併用した Web ブラウザであり，ウェブページ表示部，ページの親子関係表示部，推薦ページ情報表示部からなる (図 4)．

図 4　表示部のインターフェース

4.1.1　ウェブページ表示部

試作したブラウザには，Web ブラウザとしての基本的な操作に加えて，閲覧状態の履歴を用いた以下の操作がある．

- 1 つ前・1 つ後の閲覧状態を復元する

  1 つ前の閲覧状態を復元する操作により，各ウィンドウで表示していたページや，ウィンドウのサイズ・位置などを，まとめて復元することができる．また，この操作は繰り返すことができ，いくつでも前の状態に戻ることができる．閲覧履歴の末尾にいないときには，1 つ後の状態を復元することもできる．

- 閲覧状態の一覧表示から復元する

  閲覧履歴が短いときは，1 つ前や後に戻る操作を繰り返すことで，容易に意図した閲覧状態を復元することができる．しかし閲覧履歴が長くなってくると，1 つずつの操作では，なかなか意図した状態に辿りつけない．そこで，閲覧状態の一覧を表示し，1 つ選択することで，過去の閲覧状態に一度に戻ることができる．
- 閲覧状態をキーワード検索し，復元する

  閲覧状態に対応付けられたキーワード群を対象として，閲覧状態の検索ができる．検索結果には，該当した閲覧状態が新しいものから順に表示される．ここから 1 つ選択し，その状態を復元できる．
- 閲覧状態に対してマークを付ける

  閲覧状態に対しては任意にマーク付けを行うことができ，マーク一覧を表示して 1 つ選択，復元することができる．マークを付ける際には利用者が独自に名前を付けられる．

4.1.2 ページ親子関係表示部

ページ親子関係表示部では，これまで辿ってきたページをツリー上にして表示する (図 7)．また，表示部の頂点を選択することで，対応するページを開くこともできる．

4.1.3 推薦ページ表示部

さらに表示部は，推薦するページを提案する機能を持つ．推薦ページの計算は閲覧履歴管理部で実現される．表示部は，利用者がリンクをクリックしたり URL を入力したりすることによって新しいページを表示したときに，そのページの情報を閲覧履歴管理部に送る．この情報は，閲覧履歴管理部が推薦ページを計算するのに必要なものである．

閲覧履歴管理部に情報を送ると，閲覧履歴管理部からは 0 個以上の推薦ページの情報が送られてくる．推薦ページ情報は，ページ URL，ページタイトル，そのページの推薦値からなる．この推薦ページの情報を，一覧として提示するのが，推薦ページ情報表示部である．推薦ページは，推薦値の高い順に表示される．この一覧から 1 つを選択すると，推薦されたページを表示することができる．

推薦ページは先人が解決した同様の問題の情報を用いているので，問題を短時間で解決するのに役立つ．また，利用者の操作環境とは別の部分に，非同期に表示されるので，利用者の操作を妨害せず，また利用者が自ら推薦ページを調べるような操作をしなくてもよい，という利点がある [4]．

4.2 閲覧履歴管理部

閲覧履歴管理部は Ruby 1.8.2 を用いて，Web サーバ Apache 2.0.53 上で動作する CGI プログラムとして記述し，データベースライブラリには GNU GDBM 1.8.3 を使用した．また，動作の確認は，FreeBSD 4.11-STABLE 上で行った．

4.2.1 DB 登録・推薦値算出部

閲覧履歴管理部は表示部に Web ページを推薦する役割を持つ．そのために，クライアント部で閲覧した一連の Web ページの情報を収集する．この表示部での一連の閲覧動作を，セッションと呼ぶことにする．セッションには一意な ID を付けて区別する．また，収集したページ情報はデータベースに登録される．

そしてデータベースを参照し，推薦ページを決定する．推薦ページに対する推薦値の計算には，3.2 節で説明した協調フィルタリングを利用する．そして，計算結果の推薦ページ群を表示部に送る．

4.2.2 データベース

閲覧履歴管理部は過去に閲覧されたページの情報を，セッション毎に記録したデータベースに持っている．表示部から，新しいページを表示する度に，そのページの URL とタイトルの情報が送られてくる．送られてきた閲覧ページ情報は，そのセッションに対応したデータベースに記録する．

5. 利 用 例

本節では実際の利用例として，開発経験の浅い人がソフトウェア理解を試みた事例をとりあげる．ここでは我々の研究グループで開発している SPARS [5] のソースコード解析部を題材とする．SPARS について予備知識を持たない開発者が，処理の内容を理解するために，main 関数を起点として呼びだされた関数を逐次クロスリファレンスツールの Web インタフェースを用いて閲覧する．このとき，試作した Web ブラウザを用いることで，どのような利点が存在するかを示す．

5.1 利用例 1: クロスリファレンサを用いた場合

5.1.1 調 査 開 始

まず利用者は，本ツールを用いてクロスリファレンサにアクセスし，main 関数のページを表示する．そして main 関数を起点として，そこから呼び出されている関数を順次読み進めていったところ，構文解析ライブラリ yacc を利用していることがわかった．そこで，yacc のマニュアルも同時に参照しながら調査を進めることとした．利用者は一通り yacc についての解析を行い，ひとまず構文解析部でどのような処理が行われているかについて，概要を把握した．

5.1.2 手 戻 り

そして一度 main 関数に戻り，さらに読み進めていくと，解析結果をデータベースに登録している部分を発見した (図 5)．利用者は，利用しているデータベースのマニュアルも同時に開きながら解析を進めていく．すると，利用者はパース部分からのデータを加工している箇所を発見したため，先程閲覧した yacc 解析部に戻って，どのようなデータが入るのかを確認をする必要に迫られた．

このとき，通常の閲覧ツールであれば，各ウィンドウごとに現状をブックマークで保存して，ウィンドウの戻るボタンで main 関数まで戻り，そこから再び yacc 解析部までのリンクを辿らなくてはならない．しかし，yacc 解析部へは main 関数から何段もの関数呼び出しを経由しないと辿りつけないため，先程のリンクを正確に辿るのは利用者にとって負担となる．

5.1.3 復 元

本ツールを用いている場合，利用者は閲覧状態リスト (図 6)

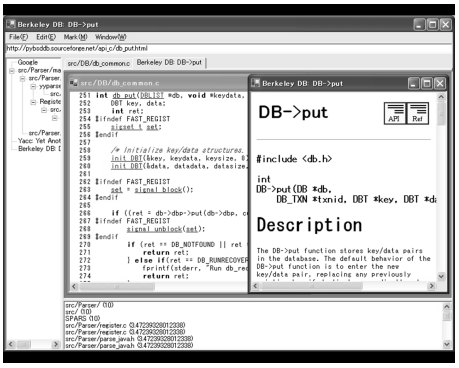

図 5　db について調査している状態

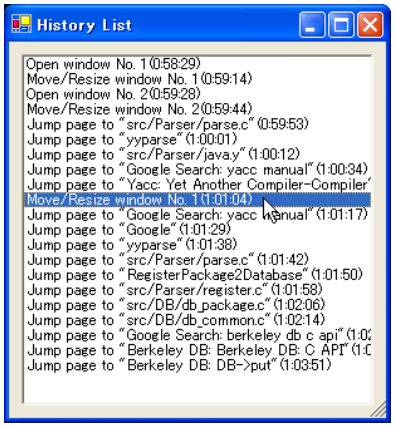

図 6　閲覧状態一覧リスト

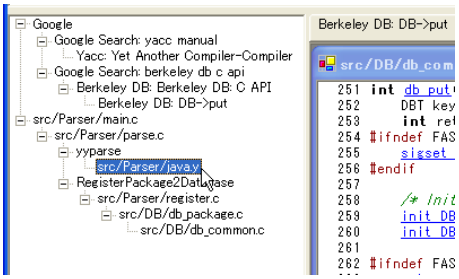

図 7　ページ親子関係表示部

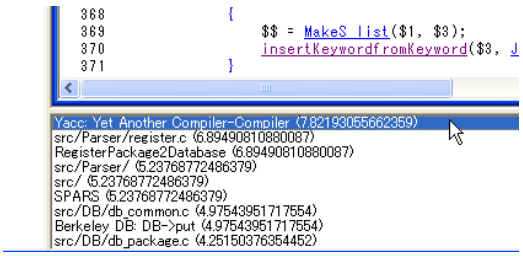

図 8　推薦ページ表示部

を呼び出して，得られた一覧表示から yacc 解析部を見ていたときの状態を選択することで，その状態を復元できる．また，そのとき同時に参照していたマニュアルも復元される．このように本ツールを用いることで，容易に任意の時点での閲覧状態を参照し，また復元することができる．

また，図 5 において，yacc のソースコードのみを再び閲覧するのならば，ページ親子関係表示部 (図 7) から該当する頂点を選択し，ページを復元すればよい．

5.1.4　調査の再開

yacc 解析部の内容が確認できたので，元のデータベース登録部の調査を再開する．そのためには，再び閲覧履歴リストを呼び出して，適切な状態を選択すれば，図 5 の状態に戻ることができる．

5.2　利用例 2: 利用例 1 の後に SPARS を調査する開発者がいた場合

本システムを用いることで，ページの推薦機能により，後に同様の調査をする人にとって有用なページが自動的に推薦される．上の行動が行われた後に，同様の調査が行われた場合，構文解析部のソースに辿りついた時点で，本システムは yacc のマニュアルページを自動的に推薦する (図 8)．このように先人の履歴から有用な情報を得ることができる．

5.3　考　　察

利用例 1 では，情報閲覧の際に閲覧状態の復元を行うことで，ソースコードとマニュアルという，同時に調べていた情報を失うことなく，閲覧作業を続けることができた．

また利用例 2 では，ページの推薦機能により，利用者の抱えている問題に関連しているページの情報を得ることができた．

6.　ま　と　め

本研究では，ソフトウェア開発環境の 2 つの問題点「システム間の連携不足による同時閲覧情報の欠損」,「複雑なリンク関係による閲覧ページ情報の欠損」を解消するための Web ブラウザの試作を行った．本システムは，閲覧状態を逐次保存し，任意の過去の時点における閲覧状態を復元する機能，閲覧ページの構造をツリー状に表示するインターフェース，他の利用者が閲覧した内容に基づいたページの推薦機能をもつ．そして，本システムを用いて実際の Web ベースのソフトウェアシステムの調査を行い，効率よく閲覧できることを確認した．

今後の課題として，定量的な評価，不要な閲覧状態記録の削除，閲覧状態の一覧を状態の縮小画像で表示，ページ内容や閲覧時間等を考慮したページ間の関連抽出，推薦機能の精度向上等が挙げられる．

文　　献

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

[2] P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom, and J. Riedl, “GroupLens: An Open Architecture for Collaborative Filtering of Netnews”, Proceedings of ACM 1994 Conference on Computer Supported Cooperative Work, pp.175-186, Chapel Hill, North Carolina, USA, 1994

[3] 市井 誠, 山本 哲男, 横森 励士, 井上 克郎, “ソフトウェア部品推薦のための協調フィルタリング手法の提案と実現”, 電子情報通信学会技術研究報告, SS2004-15, Vol.104, No.243, pp.7-12, 2004

[4] Yunwen Ye, Brent Reeves, “An Active and Intelligent Agent for Component Location”, Proceedings of Software Symposium 2000, pp.67-74, Kanazawa, Japan, 2000

[5] 横森 励士, 梅森 文彰, 西 秀雄, 山本 哲男, 松下 誠, 楠本 真二, 井上 克郎, “Java ソフトウェア部品検索システム SPARS-J”, 電子情報通信学会論文誌 D-I, Vol.J87-D-I, No.12, pp.1060-1068, 2004