コード編集とWeb閲覧履歴の時系列による統合とその活用

村松 啓寛[1,a)]　寺田 実[1,b)]

概要：プログラミングを行なっている際に疑問点等を解決するために Web ページを調べ, 参考にすることが多々ある. その際, 参考にした Web ページを別途記録しておくことはほとんどないがプログラミングしたコードを後から読む際に参考にした Web ページを再び閲覧したくなったり, プログラミング時に調べた疑問点を再び検索することがある. そこで, 本研究ではプログラミングのソースコードの編集履歴と Web 閲覧履歴を時系列で統合し, ユーザーに参考にした Web ページの情報や対象のソースコードを提示することで再度探す手間を省いたり, プログラミング時の意図の想起や推測, コードの理解の支援をするためのシステムを提案する.

1. はじめに

昨今では Web サイトでのプログラミングに関わる情報の充実や GitHub などの Web でのソースコードなどの管理, Stack Overflow などの Web でのプログラミングに関わる疑問点の質問ができるようになっている. そのため, プログラミングを行なっている際に疑問点などが出た場合に書籍で調べたり, 知識を有している人に聞くよりも手軽なため, 利用する頻度が高くなってきている.

Web を使って疑問点などを解決するためには Google などの検索エンジンを利用し, Web ページを調べ, 参考にしたり, さらに Web ページを探したりといった流れであることが多い. その際, 参考にした Web ページやその時の検索キーワードを別途記録・保存しておくことはほとんどない. しかし, 作成したソースコードを後から読む際に参考にした Web ページを再び閲覧したくなったり, プログラミングをしている時に調べた疑問点を再び検索することがある. だが, 参考にした Web ページのタイトルや検索時のキーワードを覚えていることは少ない. また, Web ブラウザの膨大な閲覧履歴から目的の Web ページや検索結果を探すことも困難である.

過去に参考にした Web ページはソースコードのある一箇所だけではなく, 複数の部分で参考にしていることは珍しいことではない. そのため, 参考にした Web ページを基にプログラミングしたソースコードの該当箇所を全て見たいことは多々ある. しかし, ある部分のソースコードで参考にした Web ページを見つけられたとしても, その Web ページを参考にした他の部分のソースコードを探すことはソースコードの量や変更量にもよるがさらに困難である.

本研究で提案するシステムはプログラミングを行う際に発生するソースコードの編集履歴と Web ブラウザでの閲覧履歴を記録・保存する. 同システムではそれらのデータを活用し, 時系列順やファイルごとに提示を行う. それらによって, ユーザーが別途記録・保存しなくても, 参考にした Web ページの情報や対象のソースコードを閲覧することでプログラミング時の意図の想起や推測, コード理解の支援を行うシステムを提案することが本研究の目的である.

2. 関連研究

ファイルと Web 閲覧履歴を紐付ける研究として俺デスク [1] があげられる. PC 上であるファイルを編集していた時に閲覧していた Web ページを紐付け, 紐付けた情報を後から提示することでそのファイルを編集していた時の行動や意図を想起する支援を行っている. 参照時刻やテキスト内検索などの複数のユーザーによる操作を基に関連度を算出し, 紐づけている. 本研究では対象としているファイルをプログラミングのコードとし, また, ファイル単位ではなくファイルの行単位の粒度で閲覧した Web ページを紐づけるという点で異なっている.

ソースコードへ情報を紐付ける研究として, ソースコードへのアノテーションによるプログラミング学習支援のシステム Source Code Annotator[2] があげられる. このシステムでは, アノテーションに関連する画像, 動画, 音声, スライド, 文書などのリソース, 参照すべき Web ページをソースコードに付加し, 共有している. ソースコードに対し行単位で参考にした Web ページなどの情報を紐づける

[1] 電気通信大学大学院情報理工学研究科
[a)] m1731155@edu.cc.uec.ac.jp
[b)] terada.minoru@uec.ac.jp

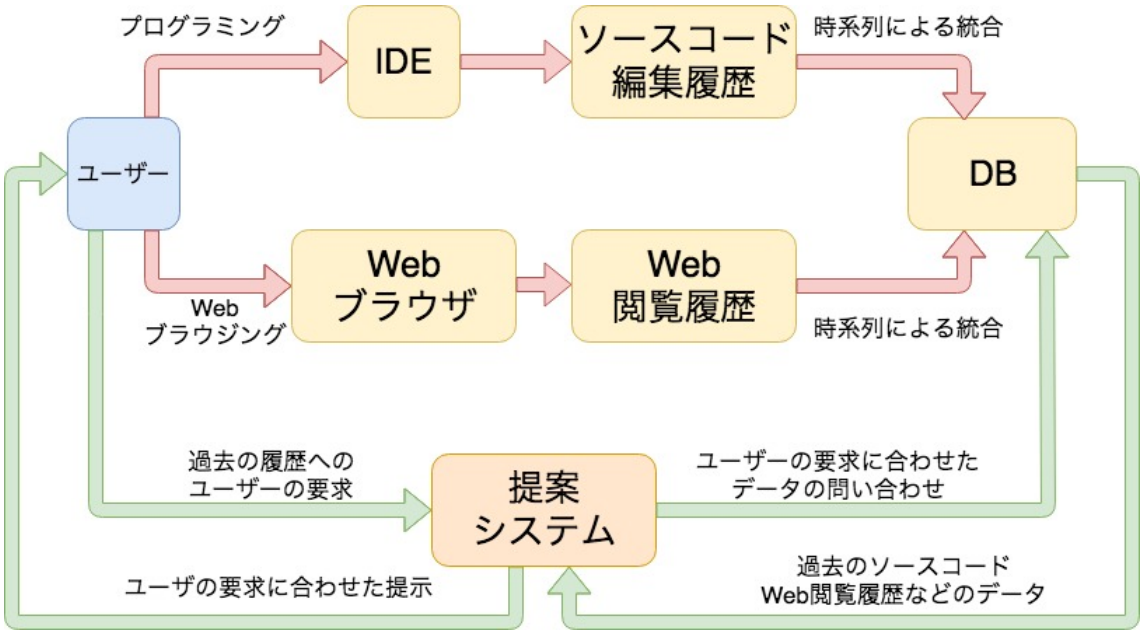

図 1 システムの概要

という点が本研究と共通している．本研究では編集履歴に基づいてソースコードに対し参考にした Web ページを自動的に紐付けている点で異なっている．

統合開発環境からソースコードの編集履歴を取得する研究として OperationRecorder[3] があげられる．このシステムは 統合開発環境である Eclipse のプラグインとして, Java 言語を対象に Eclipse 上で編集した Java ファイルの編集履歴をテキスト単位で記録を行う．記録した編集履歴は XML ファイルとして保存される．本研究での編集履歴取得には OperationRecorder を利用している．

3. 提案システム

3.1 概要

図 1 に提案システムの概要を示す．本システムは Eclipse プラグインとして実装している．統合開発環境 (IDE) は Eclipse, 編集履歴取得には Eclipse プラグインである OperationRecorder[3], Web ブラウザには Google Chrome または FireFox を利用している．また, 対象とするプログラミング言語は Java とする．

3.2 ユースケース

本システムのユースケースの一例を以下で述べる．ユーザーは本システムがインストールされた統合開発環境でプログラミングを行う．プログラミング中には様々な疑問点が生じ, Web ブラウザを利用して, Web 検索を行う．検索を行い, 参考になる Web ページを発見し, そのページを参考にしながらプログラミングを続ける．これらを繰り返し, プログラムを作成していく．

プログラムを作成して, ある程度の時間が経過した後に以下のようなユーザーの作業が発生することがある．

- ある部分のプログラムの説明
- プログラミング時にコメントを追加していなかった部分でのコメント追加
- バグが発見され, ある部分の修正
- ある部分のソースコードを参考に新たなプログラムの作成

このような作業が発生した場合にソースコードを見て, すぐにコードを理解できればいいが, 理解ができず再び Web 検索を行うことがある．本システムでは再び Web 検索を行わなくても, また, その部分をいつ編集したか覚えていなくても, 現在のソースコードの調べたい部分のコードを選択することで, その部分の過去の Web 閲覧履歴一覧を表示でき, そこから参考にした Web ページを得ることができる．

また, その部分の編集した時期を覚えていれば, 統合開発環境のプロジェクトビューから対象のファイルを選択することで編集履歴一覧を表示し, そこから対象の編集履歴を選択することでその部分の過去の Web 閲覧履歴一覧を表示でき, 参考にした Web ページを得ることができる．

ユーザーは Web ページと同時にその時のソースコードの差分も得ることができる．そのため, 参考にした Web ページとソースコードの編集の差分を同時に見ることで, プログラミング時の意図の想起や推測, コード理解ができ, 上記で述べた要求を解決できる．それぞれの機能の詳細な説明は次節以降で行う．

3.3 ソースコードの編集履歴

プログラミングは OperationRecorder および提案システムのプラグインがインストールされている統合開発環境の Eclipse で行う．ソースコードの編集履歴は Oper-

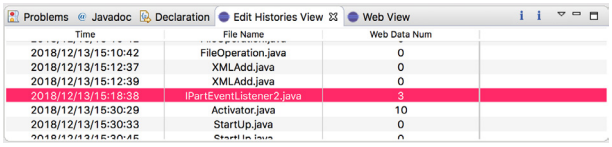

図 2　ソースコード編集履歴の時系列による一覧表示

図 3　Web 閲覧履歴の一覧表示

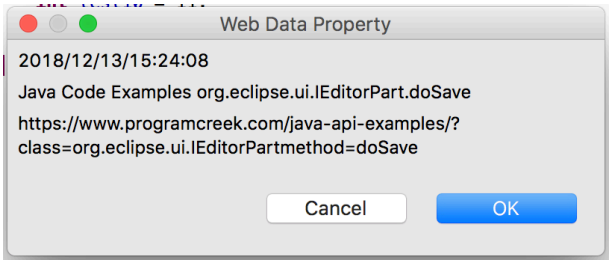

図 4　Web 閲覧履歴の詳細情報表示

ationRecorder が自動で取得し, ユーザーが開いているエディタのタブを閉じる際に XML ファイルとして保存される. そのため, ユーザーは適宜エディタのタブを閉じる操作が必要になる. しかし, 閉じる操作以外には特別な操作は必要ないため, ユーザーはプログラミングをいつも通り行うことができる.

3.4　Web 閲覧履歴

Web ブラウジングは Web ブラウザである Google Chrome または FireFox で行う. どちらのブラウザを利用するかはユーザーに選択してもらう. ブラウザによって取得できる Web 閲覧履歴に多少の差異はあるが本システムでの利用には特に影響はない. 本システムでは Web ブラウザが PC のローカルにある Web 閲覧履歴を記録しているデータベースを利用するため, ユーザーには特別な操作を要求はしない. そのため, ユーザーは普段通りのブラウジングを行うことができる.

3.5　データの時系列による統合

本システムはソースコードの編集履歴と Web 閲覧履歴は時系列による統合を行い, それらの統合された情報をデータベースに登録する. データの時系列による統合とデータベースへの登録はソースコードの編集履歴ファイルが保存されるタイミングで自動的に行う. そのため, 過去のデータに関してデータの時系列による統合およびデータベースへの登録はすでに完了しているため, ユーザーは本システムに問い合わせる操作のみで過去のソースコードや Web 閲覧履歴を見ることができる.

また, ユーザーが現在編集している部分に関する Web 閲覧履歴などの情報を閲覧したい場合には対象の Java ファイルを開いているエディタのタブを閉じることで情報を反映する必要がある.

3.6　ソースコード編集履歴の時系列表示

本システムは図 2 のようにユーザーにプロジェクト単位またはファイル単位でソースコードの編集履歴を時系列で一覧表示をする. Eclipse のビューで時系列表示を行う. 一覧表示では左から編集履歴の時刻, 対象ファイル名, 一つ前の編集履歴から今回の編集履歴までの間に閲覧した Web 閲覧履歴数を表示する. また, 一覧表示からある編集履歴をダブルクリックすることでその時に見ていた Web 閲覧履歴の一覧およびその時のソースコードを表示する. これによりユーザーはどのファイルのどの部分を編集していた時に多くの Web を見ていたかまたは探していたかを確認することができる.

3.6.1　Web 閲覧履歴の一覧表示

Web 閲覧履歴の一覧表示は図 3 に示す. 編集履歴の時系列表示と同様に Eclipse のビューで Web 閲覧履歴の一覧表示を行う. 表示されている内容はその時に閲覧した Web ページのタイトルである. Web の一覧表示の左側 (図 3A) は一つ前の編集履歴から今回の編集履歴までに見た Web の閲覧履歴である.

図 3B は今回の編集履歴より前 10 分に見た Web の閲覧履歴である. 図 3B は一つ前の編集履歴との時間間隔が短いなどの理由で図 3A に該当する Web 閲覧履歴がないがそれよりも前に今回の編集履歴に関係する Web 閲覧履歴が存在する場合があるため表示している. 現在の仕様では表示可能な時間は 10 分前までであり, 表示内容は図 3A と重複する可能性がある.

図 3C は今回の編集履歴より後 10 分に見た Web 閲覧履歴である. 図 3C は今回の編集履歴に該当するソースコードを編集していた際にさらに疑問点が発生して, Web 検索を行って, 別の Web ページを見ている可能性があるため表示をしている. 図 3B と同様に現在の仕様では表示可能な時間は 10 分後までである.

図 3A, B, C の Web ページタイトルの項目はダブルクリックを行うことでその Web 閲覧履歴の詳細情報を表示することができる. Eclipse のダイアログで Web 閲覧履歴の詳細表示を行う. 詳細情報の表示は図 4 に示す. 詳細情報表示ではその Web ページを見た時刻, ページタイトル, URL を表示している. また, OK ボタンを押すことで PC のデフォルトブラウザにて該当のページを開くことが出来る.

3.6.2　過去のソースコード表示

対象となっているファイルの選択された編集履歴におけ

図 5 選択された編集履歴におけるソースコード表示

るソースコード表示は図 5 に示す．Eclipse のエディタに読み取り専用として過去のソースコード表示を行う．ソースコードは選択された編集履歴後の内容を表示する．また，該当の編集履歴によって変更される部分の変更前のコードはコメントアウトの形式で該当部分に追加し，表示する．表示するソースコードと該当する Web 閲覧履歴一覧を見比べることで一覧の中でもこの Web ページを参考にしたといった確認を行うことが出来る．また，表示されたソースコードはある編集履歴後の状態を閲覧するために復元しているため，その状態がプログラムとして構文的に正しいとは限らない．

3.7 現在のソースコードから過去の情報表示

現在のソースコードから過去の Web 閲覧履歴やソースコードにアクセス出来るようになる機能を現在実装中である．この機能は図 5 のような現在のソースコードを編集しているエディタ上でソースコードのある部分を過去に編集していた時に閲覧していた Web ページやその時のソースコードの編集履歴を確認できる機能である．ユーザーは過去の情報を確認したい部分のソースコード上で右クリックでコンテキストメニュー (ホップアップメニュー) を表示する．その中から本システムのメニューである「過去の情報を参照する」を選択する．本システムは自動的にユーザーが選択したソースコード付近の過去の編集履歴に基づいて，過去の Web 閲覧履歴一覧を表示する．また，過去の Web 閲覧履歴一覧からユーザーがある Web 閲覧履歴を選択することで，本システムはその時の過去のソースコードを図 5 と同様に表示する．

4. 今後の課題

提案システムではユーザーは過去の Web 閲覧履歴やソースコードを取得した後にその情報を利用したユーザーへの機能が存在しない．具体的には Web 閲覧履歴やソースコードを取得した後，表示粒度が細かすぎるので粗くする，さらにその Web ページを見ていた時に編集していた他のソースコードを検索，その Web ページに関して現在のソースコードへコメントアウトの形で残したいと思うのではないかと推測する．そのため，情報を取得した後のユーザーへの機能の作成を検討している．

記録された Web 編集履歴は Web ブラウザで見た全てのページを取得しているため，ソースコードと関係ない履歴が含まれていることがある．しかし，現在はそれらの履歴も表示するようになっているため，ユーザーにはノイズを含んだ情報を提示している．そのため，ユーザーがこの Web 閲覧履歴はソースコードとは関係ない履歴だと判断した場合にはこの履歴を非表示にするといったことができる機能の作成を検討している．

本システムで取得している Web 閲覧履歴は Web ブラウザのローカルに保存している情報を取得している．その情報はその Web ページにアクセスした時刻が記録されている．その Web ページをタブに開いておいて，再び見る場合もある．しかし，その場合には Web ページのリロードは行われないことが多く，その時刻は記録されないことが多々ある．結果，本システムで提示される Web 閲覧履歴は必ずしも，そのソースコードを編集していた時に見ていた Web ページではないことが発生する．そのため，Web ブラウザでどのページをどのくらい見ていたか記録する機構の作成を検討している．また，現在のソースコードの編集履歴取得する機構は他者が作成したものであるため，取得できる情報はその機構が提供しているものに限られる．そのため，本システムにより合った情報を取得するためにソースコードの編集履歴を取得する機構を自前で作成することも検討している．

提案システムでは履歴に関するデータは PC のローカルで保存しているため，利用方法は個人開発を想定している．そのため，複数人での開発では利用できない．しかし，現在のソースコードから過去の情報へのアクセス機能および履歴情報の共有の機能を作成することで複数人での提案システムの利用が可能となる．複数人での履歴の共有が可能になることでソースコードとその Web 閲覧履歴から他者のプログラミング時の意図の想起や推測，他者が書いたコードの理解の支援に繋がるのではないかと予想推測する．そのため，複数人での履歴の共有に関する機能を検討していく．

5. おわりに

本稿では ソースコードの編集履歴と Web 閲覧履歴の時系列による統合を行い，ユーザーに提示し，ユーザーのプログラミング時の意図の想起や推測，コードの理解の支援のためのシステムの提案をした．本システムを利用すること

で自動的にソースコードの編集履歴と Web 閲覧履歴を記録・保存し, それらの情報をユーザーが活用しやすい形で提示することで再度探す手間を省いたり, プログラミング時の意図の想起や推測, コードの理解の支援が可能だと考える.

今後の課題はユーザーが本システムで記録・保存された情報を活用する手段が少ないため増やすことである. 具体的には情報の表示粒度の変更機能, 得られた情報からさらなる検索が出来る機能や参考にした Web ページに関する情報をコメントとしてソースコードに追加する機能, 記録・保存されたデータの編集機能である. 加えて, 取得する情報量を増やすために情報取得の自前のシステムの実装を検討していく. また, 複数人での利用が可能になると活用の範囲がより広くなると考えられるため, 履歴情報の共有の仕組みに関しても検討していく.

参考文献

[1] 大澤亮, 高汐一紀, 徳田英幸. “俺デスク：ユーザ操作履歴に基づく情報想起支援ツール”. 情報処理学会 第 47 回 プログラミング・シンポジウム, 2006.

[2] 伊藤瑠哉, 竹島亮, 長尾確. “バージョン管理システムと連携したソースコードへのアノテーションによるプログラミング学習システム”. 情報処理学会 第 78 回全国大会講演論文集, pp. 309-310, 2016.

[3] Omori, T. and Maruyama, K.: A Change-aware Development Environment by Recording Editing Operations of Source Code, in Proceedings of the 2008 International Working Conference on Mining Software Repositories, 2008, pp. 31–34.