科学研究費助成事業　　研究成果報告書

平成 ２６ 年　 ６ 月 １３ 日現在

機関番号：３２６４４
研究種目：若手研究(B)
研究期間：2012～2013
課題番号：２４７００９１２

研究課題名（和文）受講者の理解の程度をリアルタイムで把握するシステムの開発とその評価

研究課題名（英文）A Programming Process Visualization System With Global Hooking

研究代表者
森田　直樹（MORITA, Naoki）

東海大学・情報通信学部・准教授

研究者番号：５０４１３５７１

交付決定額（研究期間全体）：（直接経費）　　3,100,000円、（間接経費）　　930,000円

研究成果の概要（和文）：プログラミング演習の授業において，受講者がPCに対して行ったすべての操作を取得するシステムを開発した．プログラミング演習において，受講者や教師を支援するシステムは，数多く存在する．従来の方法は，ソースコードを編集する過程を取得するために専用のエディタが必要であった．
本研究で開発したシステムは，専用のエディタを用いることなくソースコードを編集する過程を取得することができる．具体的には，PCを操作することで発生するOSのシステムメッセージを解析することにより取得する．これにより，従来では不可能であった既存のソフトウエアを用いてもソースコードを編集する過程を取得することができる．

研究成果の概要（英文）：I have developed a system capable of acquiring and later reproducing all the operations performed by a student that uses a personal computer (PC) to complete a programming exercise as a class assignment, with the completed program to be submitted to the instructor. Submission of a completed assignment is sometimes required, to assess the students' understanding of the class content. The assessment is generally made on the basis of the final form of the completed assignment, but in some assignments the process of assignment performance is also important. In contrast to the systems proposed in other studies for such assessment, the system described in the present study does not require a special editor to capture the process of source code editing performed by a student. Rather, it captures the process by analyzing the OS system messages generated in the student's operation of the PC.

研究分野：総合領域

科研費の分科・細目：科学教育・教育工学

キーワード：プログラミング教育　演習履歴　データベース　理解度

１．研究開始当初の背景

（１）従来の演習は、課題に対する提出物をもとにフィードバックを行っていた。つまり、結果のみを重視していた。そのため、提出物からだけでは、どのように考えたのかを把握しづらい場合がある。特に、処理工程が重要となるプログラミングなどの講義では、その傾向が顕著となる。そこで本研究は、演習に取り組む試行錯誤の過程に着目する。

（２）プログラミング演習において、受講者や教師を支援するシステムは、数多く存在する。特にプログラミング演習では、ソースコードを編集する過程が重要である。ソースコードを作成する過程には、受講者がどのように演習に取り組んだのかを講師が判断する情報が豊富に含まれる。従来の方法は、ソースコードを編集する過程を取得するために専用のエディタが必要であった。

２．研究の目的

（１）本研究の１つ目の目的は、専用のエディタを用いることなくソースコードを編集する過程を取得し、かつ、コンパイルや実行ファイルの実行などのプログラミングに関するすべての動作を記録することである。

① 本研究で想定するプログラミング演習の環境は、OS は Windows とし、実行ファイルを作成する方法として、Visual Studio コマンドプロンプトからソースコードをコンパイルする方法をあげる。この方法は、プログラミング以外の最低限の操作のみで演習を行うことができ、プログラミング初心者を対象とした授業で用いられることが多い為である。

② 本研究で想定する２つ目の環境は、実行ファイルを作成する方法して、Visual Studio IDE を用いてビルドする方法を対象とする。この方法は、ソースコードを作成する前にプロジェクトを準備したり、Visual Studio IDE の操作を習得したりする必要がある。そのため初心者には，プログラミング以外の要素も習得する必要はあるが、眼には見えないメモリ空間上の変数の値を可視化したり、自分が作成したソースコードをどのような順番で実行しているのかトレースしながら確認したりすることができる為である。

（２）本研究の２つ目の目的は、理解の修正が必要な受講者には、受講者と共に試行錯誤した過程を振り返りながら適切なコメントを返すことができるように、その振り返りポイントを素早く探せるようにすることである。具体的には、目的の（１）で取得した情報の中から有益な情報に素早くアクセスする為の手段の確立である。

３．研究の方法

（１）受講者の PC 画面とその取得した画面の特徴をあらわすための情報を取得するシステムを開発することで、専用のエディタを用いることなくソースコードを編集する過程を取得し、かつ、コンパイルや実行ファイルの実行などのプログラミングに関するすべての動作を記録するシステムを作成する。具体的には、マウス操作のタイミングで画面を取得しその画像をサーバに蓄積することと、キー操作のタイミングで編集内容をテキスト情報としてサーバに蓄積するシステムを開発する。

（２）C 言語や Java 言語などのプログラミングの講義では、ソースコードを作成する、コンパイラでソースコードを実行ファイルに変換する、実行ファイルの動作を確認する、一連の工程を行う。そのため、構文エラーは、コンパイルエラーで確認することができ、アルゴリズムエラーは、実行ファイルの動作を確認することによりある程度可能である。このことから、（１）で取得した情報に対して、コンパイルのタイミングの演習状態にいち早くアクセスできることと、そのタイミングから演習状況を遡ぼったり、進めたりすることができるシステムを開発する。

４．研究成果

（１）本システムで開発したシステムのシステム構成

本システムは、プログラミング過程取得モジュール、プログラミング過程蓄積モジュール、プログラミング過程再現モジュールから構成される。

プログラミング過程取得モジュールは、受講者の PC 上で動作し、受講者が PC を操作するイベントの取得とその時の受講者の PC 画面をキャプチャした画像をプログラミング過程蓄積モジュールに送信する。プログラミング過程蓄積モジュールとプログラミング過程再現モジュールは、Web サーバ上で動作し、蓄積モジュールは、プログラミング取得モジュールから送られたデータをサーバ上に蓄積し、再現モジュールは、Web ブラウザ上で受講者の PC 画面を再現するための HTML を形成する。以下に各システムの詳細と成果を示す。

（２）プログラミング過程取得モジュール

本モジュールは、受講者の演習過程を取得するアプリケーション OperationLog.exe とグローバルフックを行うための OperationLog.dll、Visual Studio IDE の機能を拡張するアドイン、Visual Studio コマンドプロンプト上での操作を取得するバッチファイル cl.bat から構成される。

①演習過程として取得する情報とタイミング

Windows PC は、マウスやキーボードを用い

て操作を行う。それらの操作の結果、アプリケーションが動作し、その結果が画面に反映される。そのため、受講者の PC 画面をキャプチャし蓄積することで、受講者が行った演習過程を後から振り返ることができる。本研究で開発した OperationLog.dll は、OS に対して SetWindowsHookEx 関数を用いてグローバルフックを行いウインドウメッセージを監視する。これにより、マウス操作やキー操作を操作対象となるウインドウに反映させながら、その操作イベントを取得することができる。本 OperationLog.dll が監視する項目は、WM_LBUTTONDOWN（マウス左ボタン押し下げ）、WM_LBUTTONDBLCLK（マウス左ボタンダブルクリック）、WM_RBUTTONDOWN（マウス右ボタン押し下げ）、WM_CHAR（キーボードからの文字の入力）である。

本研究で開発した OperationLog.exe は、DLL が監視中に該当操作を取得したタイミングで受講者の PC 画面のキャプチャを行う。さらに、その該当の操作が、画面全体のどの部分に対しての操作であるかを確認しやすくするために、キャプチャ画像の対応箇所に赤枠でハイライト処理を施す。これにより、キャプチャした画像を後から振り返った時に、ウインドウ全体に対してどの部分に行った作業であるのかを確認することができる。さらに、OperationLog.exe は、取得した画像を後から検索しやすくするために、GetWindowText 関数や GetModuleFileNameEx 関数を用いて、操作対象となるアプリケーション名やファイル名を取得する。これらの情報は、ユーザ識別情報やキャプチャ画像などと総合的に関連付けられ UDP でプログラミング過程蓄積モジュールへ送られる。

②ソースコードの編集過程を取得する

プログラムの基となるのは、ソースコードである。そのため、ソースコードを作成する過程には、受講者の理解の程度を講師が推測するための情報が豊富に含まれている。OperationLog.exe を用いることで、ソースコード作成時の受講者の画面も取得することができる。しかし、蓄積してある大量の画像データの中から受講者にコメントを返すための適切な画像を探しだすことは困難である。そのため、ソースコードを編集している場合には、ソースコードも取得し画像データと関連付けて蓄積する手法をとる。これにより、本研究では未実装であるが、先行研究のようにシステムが受講者の演習状況を自動で判断するリソースとして利用することができる。

（あ）テキストエディタ上のソースコードの取得法

メモ帳や Terapad などのテキストエディタは、ウインドウ内に Edit クラスのエディットコントロールが配置されており、ユーザは、その領域内で文字列の編集を行う。エディットコントロール内にある文字列は、WM_GETTEXT メッセージにより取得できる。この特徴を利用して、OperationLog.exe は、テキストエディタになりすまして OS に WM_GETTEXT メッセージを発行し、受講者が作成しているソースコードを取得する。取得したソースコードは、同時に取得したキャプチャ画像と関連付けを行い、プログラミング過程蓄積モジュールへ送信する。

（い）Visual Studio IDE 上のソースコードの取得法

Visual Studio IDE 上のコードエディタは、VsTextEditPane クラスから構成される。そのため、コードエディタ上で編集されるデータは、メモリ空間に存在するデータを基に描画データとして、つまり、画像としてユーザに提供される。そのため、役割ごとに色分けを施したり、あるまとまりを折りたたんだ状態で表示したりすることができる。その一方で、テキストエディタのように WM_GETTEXT メッセージではコードエディタ上のソースコードを取得することができない。本研究では、コードエディタ上のソースコードを自動保存するアドイン開発することにより、ソースコードの作成過程をテキストデータとして取得できるようにした。

③コンパイルの結果を取得する

ソースコードの作成に区切りがついた段階で、構文エラーや記述ミスがないかを確かめるために、また、実行ファイルを作成するためにコンパイルを行う。コンパイラの出力結果は、受講者の取り組みに対して客観的なフィードバックとなる。そのため、エラーがない場合は、または、エラーがあっても自分の力で間違いを修正できる受講者は、自分の理解を修正したり、理解を確実なものにしたりできる。一方で、プログラミングに苦手意識を持つ受講者は、やみくもに変更し始める要因となりえる。そのため、演習過程を振り返った際にコンパイラが出力したメッセージを確認できることは重要である。

（あ）Visual Studio コマンドプロンプト

受講者がコンパイルしたタイミングでコンパイルメッセージなどの情報をプログラミング過程蓄積モジュールへ送るために、コンパイルコマンドと同じ名前のバッチファイル cl.bat を記述した。バッチファイルに記述した内容は、通常のコンパイル処理に加え、コンパイルメッセージをファイル化し、ソースコード、メッセージファイル、ユーザ識別情報をプログラミング過程蓄積モジュールへ送信するための処理である。

（い）Visual Studio IDE の場合

Visual Studio IDE を用いてコンパイルを行う場合には、コンパイルメッセージは、該当のフォルダに HTML ファイルとして自動保

存される。本研究で開発したアドインは、コンパイルメッセージが記述された HTML ファイルをユーザ識別情報と共にプログラミング過程蓄積モジュールへ送信する。

（3）プログラミング過程蓄積モジュール

プログラミング過程蓄積モジュールは、Web サーバ上で動作する。本モジュールは、受講者の PC 上で動作するプログラミング過程取得モジュールから送られたデータを、ユーザ情報ごとに、画像データとその画像に関するウインドウ情報やソースコードなどのテキスト情報を時系列で管理する。

（4）プログラミング過程再現モジュール

Web サーバ上で動作するプログラミング過程再現モジュールへのアクセスは、Web ブラウザを用いる。本モジュールが提供する情報は、HTML で構成されており３階層構造となる。

１階層目である TOP 画面は、受講者全員の現在の状況が確認できるリストが提供される。リストは、テーブルタグを用いて構成されており、その要素は、学生証番号、最後に取得した画像に関連付けられたテキストデータである。テキストデータがソースコードの場合には、カーソル行の１行が表示される。これらの情報は、Ajax を用いて常にプログラミング過程再現モジュールと通信を行う。これにより、再読み込みの操作を行うことなく情報を更新し提供することができる。操作の過程を確認するには、学生証番号をクリックすることにより次の画面に移動する。

2 階層目で提供される情報は、TOP 画面で選択した受講者のテキスト情報を時系列から逆順でリスト化したものである。リストの構成要素は、プログラミング過程蓄積モジュールが取得したタイムスタンプ、画像に関連付けられたテキスト情報であり、その情報がソースコードの場合には、編集行を、コンパイル項目の場合には、コンパイラのメッセージが表示される。コンパイラのメッセージに対しては、エラー時は赤色、成功時は青色で色分けが施される。確認したい時間帯のタイムスタンプをクリックすることにより受講者の PC 画面のキャプチャ画像が表示される 3 階層目へ移動する。

3 階層目で提供される画面は、受講者の PC 画面のキャプチャ画像を、サムネイル形式で複数枚表示したり、1 枚ずつ表示したりすることができる。画像は、ボタンにより進めたり戻したりすることができる。

（5）本システムの効果

プログラミングを行う場合、目的の動作となるプログラムの記述方法は、いく通りも考えられる。そのため、講師が予測できる注意事項や間違いは、あらかじめ学生に説明することができる。しかし、初心者が感じる疑問や素朴な間違いは、講師の予測をはるかに超える。このような間違いに関しては、取得した操作過程をスクリーンで再現しながら受講者全員に説明することができるようになった。

従来の演習スタイルでは、コンパイルエラーの内容とプログラムの修正の関係を明示的にすることができなかったが、本研究にて開発したシステムは、演習の過程をすべて蓄積し再現することができるようになった。これらにより、演習中に間違いのきっかけとなった内容を学生と共にふりかえったり、一見でたらめに作成したとしか思えないプログラムであってもどのような理由により学生が間違えるきっかけとなった原因を確認したりすることができるようになり、適切なコメントを返すことができるようになった。

５．主な発表論文等

〔学会発表〕（計２件）

① 森田 直樹、グローバルフックを用いたプログラミング過程可視化システム、FIT2013(第 12 回情報科学技術フォーラム)講演論文集、査読有、第３分冊、2013、61-64

② Naoki MORITA、A Programming Process Visualization System With Global Hooking、SITE 2014 proceeding、査読有、2014

６．研究組織

(1)研究代表者

森田　直樹（MORITA,Naoki）

東海大学・情報通信学部・通信ネットワーク工学科・准教授

研究者番号：50413571