

付箋によるコミュニケーション機能を備えた ソフトウェアブラウザ

沢田 洋平[†] 大久保弘崇[†] 粕谷 英人[†] 山本晋一郎[†]

[†] 愛知県立大学大学院 情報科学研究科 〒480-1198 愛知県愛知郡長久手町大字熊張字茨ヶ廻間 1522-3
E-mail: †sawada@yamamoto.ist.aichi-pu.ac.jp, ††{ohkubo,kasuya,yamamoto}@ist.aichi-pu.ac.jp

あらまし ソフトウェア開発において、情報の交換や共有などのコミュニケーションは重要な役割を果たす。本稿では、ソースプログラムに関係するソフトウェア開発者間のコミュニケーションの一種である、付箋によるコミュニケーションに注目し議論する。また、ソースプログラムに対して電子化された付箋を貼る機能を備え、ソースプログラム上のオンラインコミュニケーションを可能にしたソフトウェアブラウザ JSPIE2 を提案する。

キーワード 付箋, コミュニケーション, プログラム理解, ソフトウェア文書, ブラウザ, ソースプログラム

A Software Browser for supporting communication with sticky notes

Youhei SAWADA[†], Hirotaka OHKUBO[†], Hideto KASUYA[†], and Shinichirou YAMAMOTO[†]

[†] Graduate School of Information Science and Technology, Aichi Prefectural University 1522-3
Ibaragabasama, Kumabari, Nagakute-cho, Aichi-gun, Aichi, 480-1198, Japan
E-mail: †sawada@yamamoto.ist.aichi-pu.ac.jp, ††{ohkubo,kasuya,yamamoto}@ist.aichi-pu.ac.jp

Abstract In software development activity, communication such as exchanging and/or sharing information play a important role. In this paper, we discuss the communication style through sticky notes in software developer team. And we propose software browser JSPIE2 which enable us to stick note into source program in order to support communication among developers.

Key words sticky notes, communication, program understanding, software document, browser, source program

1. はじめに

ソフトウェアはソースプログラムと仕様書や設計文書などの各種のソフトウェア関連文書から構成される。それらの文書は、ソフトウェア開発グループの全員に関わる公式な情報を保持し、長期に渡って維持管理され、また文書量も多い。一方、相対的にプライベートで一時的かつ量が少ない情報も存在する。現在、この種の情報の管理は、アドホックに管理され、個人的に利用されるだけである。しかし、これらの情報が果たす役割も大きく、ソフトウェア開発・保守を支援する上で重要である。

本稿では、私的・一時的・少量の情報を管理するために付箋に着目した。付箋は、本来、対象文書の注目している場所に覚え書きを残したり目印を付けるために使われる。そして、不要になった場合に簡単に取り除くことができる。付箋に書かれるのは個人的な覚え書きや少人数で回覧される情報が多く、保存期間は比較的短期であるため、ソフトウェアの開発・保守工程における私的・一時的・少量の情報を管理するのにふさわしい。

ソースプログラムの閲覧やレビュー時に、作業者は気付いたことや疑問点などを付箋に保存したり、マーカーにより重要な箇所に印を付けることは一般的である。この種の情報は私的・一時的・少量の情報の典型例である。

本稿では、付箋を使ったコメントをソースプログラム上に貼ることによって行なわれるコミュニケーション(情報共有、情報交換)を支援する機能を備えたソフトウェアブラウザ JSPIE2 を提案する。JSPIE2 を用いることで、ソースプログラム閲覧中に発生する一時的な情報を電子的な付箋として保存し管理することができる。また、電子的な付箋に蓄積された情報をネットワークを介して利用でき、作業者間で共有できるため、付箋を使ったソースプログラム上のオンラインコミュニケーションが可能となる。

関連研究として Adobe Acrobat [5] のオンライン注釈が挙げられる。これはネットワークを介して共有された PDF 文書に注釈をつけ、複数の関係者の間で回覧やレビューを行うものである。Acrobat の注釈は PDF 文書上の位置やテキストなどを

対象に注釈を付けるが、JSP/ServletではJavaソースプログラムを3.5.1節で説明するJX-model [3] の特定のオブジェクトに分類し、これを対象としてHTML化されたJavaソースプログラム上に付箋を付ける点が異なる。

2. 付箋

2.1 付箋の特徴

付箋は本来、対象文書の注目している場所に覚え書きを残したり目印を付けるために使われる。ここではその付箋がどのような特徴を持ち、どのように利用されるべきかを考察する。まず付箋の特徴を挙げると以下のような点が挙げられる。

保存期間

必要なときに付け、必要が無くなれば取ってしまうものであり、永続的なものではない。長期に保存するような情報より、短期的な情報を持つ場合が多い。

位置情報

付箋は閲覧者が注目している場所につけられる。そのため、その場所を指しているという位置情報を持っている。場所は、一点を指している場合と、一定の範囲を指している場合が考えられる。

利用方法

仕様書やマニュアルはフォーマルな情報を扱うのに対して、付箋は個人の覚え書きや小規模のグループで共有するような、プライベートな情報を扱う場合が多い。

文書量

覚え書きなどに用いられるため文書量は少ない。また、ある文書の上に付けるため、大量の文書を載せることはできない。

付箋というのはこのような特徴を持った文書であることが分かる。そのため、付箋はソフトウェアの開発や保守時に存在する一時的な情報を文書化し管理するのにふさわしいと考えられる。紙の付箋でこれを行うのは難しいため、付箋は電子化する必要がある。

また、ソフトウェア関連文書には、ソースプログラム、仕様書、マニュアル、設計段階で用いられる文書や図などが考えられる。ソフトウェアの開発や保守時に存在する一時的な情報を電子化し管理すれば、付箋もこれらと同様にソフトウェアに必要ななくてはならない文書となる。

2.2 付箋によるコミュニケーション

付箋を付けることは、その場所に情報を保存するということであるが、その付箋を他の人が見れば情報を伝え共有するというコミュニケーションになる。個人的な覚え書きなどの場合も、将来の自分とのコミュニケーションと考えることができる。このように付箋によるコミュニケーションは、付箋が2.1節で挙げた特徴を持っているため、ある文書の特定の場所に関しての短期的でプライベートな情報をやりとりするのに向いていると考えられる。

付箋によるコミュニケーションが利用できる場面の一つとして、プログラムの理解支援が考えられる。プログラムを理解するときには、ソースプログラムを閲覧するという方法が一般的である。このソースプログラムを閲覧しているときに気付い

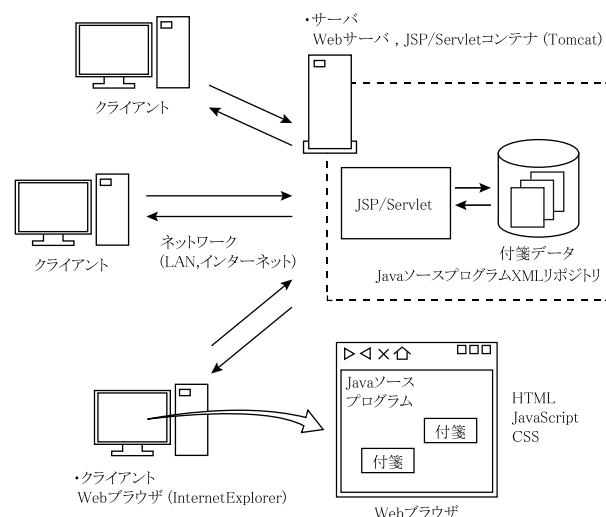


図1 JSPIE2の構成

Fig.1 Structure of JSPIE2.

たことや疑問点などを付箋を使ってソースプログラム上に保存する。これを個人の覚え書きとして利用したり、他の閲覧者やそのプログラムの作成者の間で共有し情報交換などのコミュニケーションに利用できる。プログラムを理解する場合、自分ひとりで考えるのではなく、他の人とコミュニケーションをとることは重要である。他の人の考えを聞くことで、理解が深まったり、そのコミュニケーションの中で新たな発見が見つかる場合もある。このように付箋によるコミュニケーションはプログラムの理解支援に利用できると考えられる。

3. JSPIE2

3.1 概要

本稿では付箋によるコミュニケーション機能を備えたソフトウェアブラウザ JSPIE2 を作成する。対象のプログラミング言語はJavaとし、Webブラウザを用いてJavaソースプログラムを閲覧する。特定のオブジェクトに対して付箋を付けることができ、ネットワークを介して多人数で閲覧できる。JX-modelについては3.5.1節で説明する。

このシステムの構成を図1に示す。JSPIE2はWebアプリケーションである。サーバにインストールし、クライアントマシンからWebブラウザを使ってアクセスして使用する。そのため、ネットワークを（LAN、インターネット）経由して利用できる。また情報はサーバを介してクライアント間で共有される。

サーバにTomcat [6] を使用し、JSP/Servletで実装されている。出力にはDHTML(HTML, JavaScript, CSS)を使い付箋機能を実現している。

3.2 画面構成

JSPIE2はフレームによって4つに分割されている。図2は動作中の画面である。ソースビューフレーム（画面右上）ではソースプログラムと付箋を表示し、ツリービューフレーム（画面左上）ではツリービューを表示する。表示する単位はファイルである。また、メニューフレーム（画面左下）とサブフレーム（画面右下）ではJSPIE2の様々な操作を行う。

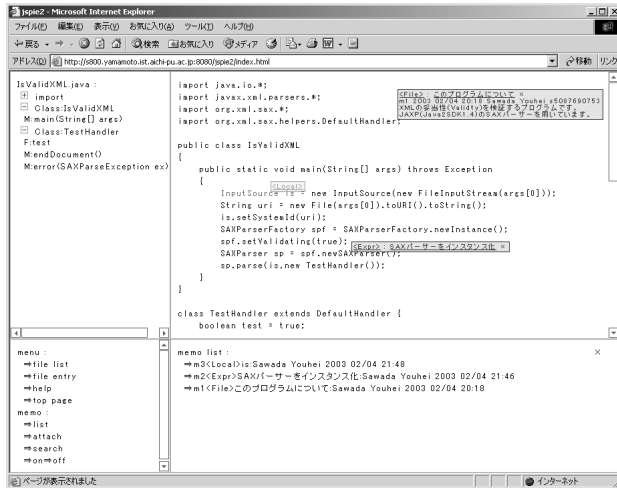


図 2 動作画面

Fig. 2 A screen of operation.

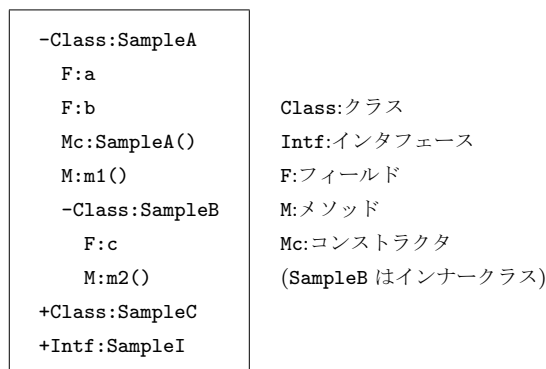


図 3 ツリービューの例

Fig. 3 An example of tree view.

3.3 機能

3.3.1 基本機能

JSPIE2 の基本機能は以下の 5 つである。

ツリービュー

Java ソースプログラムの構成をツリー型に表示する。クラスまたはインタフェースがノードとなり、フィールド、メソッド、コンストラクタがリーフとなる。ツリーは折りたたみが可能で、Windows のエクスプローラー風の動作をする。ツリービューの例を図 3 に示す。図 3 のツリービューではこのファイルにはクラス SampleA、クラス SampleC、インタフェース SampleI があることを示している。そして SampleA クラスを展開し、その中にフィールド a、フィールド b、コンストラクタ SampleA()、メソッド m1()、インナークラス SampleB があることを示している。

ソースビュー

ツリービューのオブジェクトのリンクから、ソースプログラム上のそのオブジェクトを参照する。付箋はこのソースビュー上に表示される。

付箋貼付

ソースプログラム上をクリックして対象オブジェクト、及びその位置を決め、テキスト形式の付箋を貼り付けることができる。

表 1 付箋データ

Table 1 Tag data

名前	説明
title	タイトル
name	作者
mail	作者のメールアドレス
contents	本文
obj	対象オブジェクト
id	対象オブジェクト id
posx	オブジェクト上の付箋位置 (x 座標)
posy	オブジェクト上の付箋位置 (y 座標)
id (memo)	付箋の id
date	日時
ip	IP アドレス

付箋の持つ情報を表 1 に示す。obj, id, posx, posy はユーザがマウスクリックで入力する。また title, name, mail, contents はユーザがキーボードで入力する。id(memo), date, ip はサーバで自動的に入力される。

付箋表示

貼付された付箋はソースプログラム上の対象オブジェクト上の指定された位置に表示される。付箋の状態は 4 つあり、状況に応じて変えられる。付箋の状態については 3.3.2 節で詳しく述べる。

付箋検索

付けられている付箋の検索を行う。HTML フォームから検索するキーワード、検索する付箋データの種類、検索する範囲（ブラウザ中のファイルについている付箋か全ファイルについている付箋）を入力し条件に一致した付箋のリストが表示される。この付箋のリストには、付箋へのリンクが張られている。

3.3.2 付箋の状態

付箋の状態（表示モード）は 4 つあり、クリックなどの操作によって状態を変化させることができる。この例を図 4 に示す。また、ドラッグで一時的に場所を移動させることができる。これはリロードすると元の場所に戻る。

通常状態

付箋の付けられているオブジェクトとタイトルを表示する。どの状態になっていてもリロードするとこの状態に戻る。

内容表示状態

付箋のタイトルをクリックし内容を表示させた時、貼り付け直後、リストから付箋を選択した時、検索結果から付箋を選択した時にこの状態になる。表示される内容は、付箋 id、日時、作成者、対象オブジェクト id、本文である。

縮小状態

付箋が付けられているオブジェクトを確認する場合に使用する。オブジェクト名のみを表示し、不透明度を 80% から 40% にする。オブジェクト名をクリックするとこの状態になり、このときソースプログラム上の対象オブジェクトが一定時間オレンジ色に変わり、そのオブジェクトが分かるようになっている。

非表示状態

付箋の“×”をクリックするとこの状態になる。全ての付箋を表

通常状態（不透明度 80%）

<File>: このプログラムについて ×

内容表示状態（不透明度 80%）

<File>: このプログラムについて ×

no.1 2003 01/21 16:20 Sawada Youhei s5087690753
XML の妥当性 (validity) を検証するプログラムです。
JAXP(Java2SDK1.4) の SAX パーサーを用いています。

縮小状態（不透明度 40%）

<File>

図 4 付箋の状態の例

Fig. 4 Examples of the state of a tag.

示にする memo on, 付箋のリストから選択, リロード, で非表示から戻る。

3.4 利用の流れ

基本的な利用の流れを説明する。

(1) JSPIE2 を開くと, サブフレームに file list (閲覧できるファイルのリスト) が表示される。そこから閲覧するファイルを選択する。

(2) ソースビューフレームには付箋の付けられたプログラムソースが表示される。このとき, サブフレームには付箋のリストが表示され, そこから付箋を参照できる。また, ツリービューフレームにはプログラムがツリー型に表示され, そこからオブジェクトを参照できる。

(3) 付箋を付けるには, メニューフレームの attach を選択する。サブフレームに付箋貼付用のフォームが表示されるので, ソースビューフレームのソースプログラム上で付箋を付けるオブジェクトをクリックして対象を決め, 内容を書き込み送信する。

(4) 貼り付けが完了すると, ソースビューフレームが更新されて貼り付けた付箋が内容表示状態で表示される。

3.5 実装

3.5.1 JX-model

JSPIE2 の実装において, Java ソースプログラムを特定のオブジェクト分類し, そのオブジェクトに対して付箋を付けるようにするため JX-model [3] を使用する。JX-model とは Java ソースプログラムの XML リポジトリのモデルである。JX-model は Java(Java2 ver. 1.4) のソースプログラムを 20 種類の非終端要素と 7 種類の終端要素によってモデル化している。ここで, 終端要素は子ノードとして JX-model 要素を持たない要素のことをいう。要素の一覧を表 2 に示す。表 2 において, File 要素から Expr 要素までが非終端要素であり, ident 要素以下が終端要素である。

JX-model は様々な CASE ツールの要求に応えるために, Java のソースプログラムを細粒度で抽象化して表現されている。ソースプログラムに直接タグ付けを行い, インデントや改行などのスタイルを保持している。また, XML リポジトリからは XSLT や DOM/SAX などの XML 関連技術を用いて情報を取り出すことができる。

```
public class Hello {  
    public static void main(String[] args) {  
        System.out.println("Hello");  
    }  
}
```

図 5 解析対象 Java ソースプログラム

Fig. 5 The Java source program for analysis

```
<?xml version="1.0"?>  
<!DOCTYPE File SYSTEM "JX-model3.dtd">  
<File id="s5087690753"><Class id="s5091885057" ...  
</nl><sp> </sp><Method id="s5104467969" typefi ...  
</nl><sp> </sp><Stmt id="s5121245186" sort=" ...  
</nl><sp> </sp><op></op></Stmt></Method><nl>  
</nl><op></op></Members></Class><nl>  
</nl></File>
```

図 6 XML リポジトリ

Fig. 6 XML repository

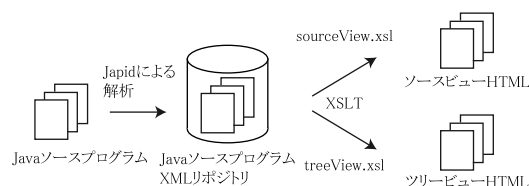


図 7 システム構成 1

Fig. 7 System configuration 1.

この JX-model に基づいた XML リポジトリは Japid [2] を使って生成する。Japid とは, CASE ツールプラットフォーム Sapid [1] の一部であり, Java ソースプログラムの解析を行う。JX-model の例として, 図 5 の Java ソースプログラムに対する XML リポジトリ図 6 のように得られる。

JSPIE2 では, この JX-model のオブジェクトのうち, Members, Type, ident, kw, op, sp, nl の 7 種類の要素を除いた 20 種類の要素を単位として付箋を付けを行うようにした。

3.5.2 システム構成

まず Japid を用いて Java ソースプログラムから XML リポジトリを生成する。XSLT を用いて, ツリービュー HTML とソースビュー HTML を生成する。この処理を図 7 に示す。このとき図 7 にはないが, 付箋のデータを保存するための付箋データ XML も作成する。この処理はファイル登録時に一度だけ行われる。

ソースビュー HTML は, ソースプログラムの表示と付箋付けのベースとして使用する。付箋の対象とするオブジェクトを span タグで囲み, id 属性に JX-model の id を, title 属性にオブジェクトの種類を設定したものである。ソースビュー HTML の例を図 8 に示す。ツリービュー HTML はツリービューのためのもので, クラス, インタフェース, フィールド, メソッド, コンストラクタの情報を取り出し, ツリー型になるように配置したものである。

次に付箋の貼り付け時と表示時に行われる処理を示す。Web

表 2 JX-model の XML 要素
Table 2 The XML element of JX-model

要素名	説明
File	ソースファイル全体を表す
Package	パッケージ宣言を表す
Import	パッケージインポート宣言を表す
Class	クラス定義を表す
Intf	インタフェース定義を表す
Ctor	コンストラクタ宣言を表す
Method	メソッド宣言を表す
SInit	static 初期化子を表す
Field	フィールド宣言を表す
Param	手続き宣言内のパラメータ宣言を表す
Local	手続き内のローカル変数宣言を表す
ExtdOpt	型宣言における派生元宣言を表す
ImplOpt	型宣言における実装インタフェース宣言を表す
ThrwOpt	手続き宣言における例外宣言を表す
Members*	クラス・インタフェースの本体を表す
QName	パッケージ名と未解決の名前を表す
Type*	型を構成するトークンを表す
Stmt	文を表す
Label	ラベル定義を表す
Expr	式を表す
ident*	識別子を表す
literal	リテラルを表す
comment	コメントを表す
kw*	キーワードを表す
op*	演算子を表す
sp*	空白文字とタブを表す
nl*	改行文字を表す

* 付箋の対象としない要素

```

<span id="s5104467970" title="Method">
  public static void main(
    <span id="s5100273665" title="Param">
      String[] args</span>)
    <span id="s5121245187" title="Stmt">{
      <span id="s5121245188" title="Stmt">
        <span id="s5125439493" title="Expr">
          hello()
        </span>;
      </span>
    </span>
  </span>
</span>

```

図 8 ソースビュー HTML の例

Fig. 8 An example of source view HTML.

ブラウザ上のフォームから送信された付箋データは、付箋データ XML に保存される。ここから必要な情報を読み出し、ソースビュー HTML と合わせて出力する。この処理を図 9 に示す。

ソースビューでのマウスクリックは新規付箋の貼り付け操作になる。ソースビュー HTML に埋め込んだ `span` タグの `id` 属性より、どのオブジェクトを指しているかを特定し、付箋貼付フォームにデータを送る。このときフォームの `object id`, `object` (対象オブジェクトの種類), `position` (付箋を付ける位

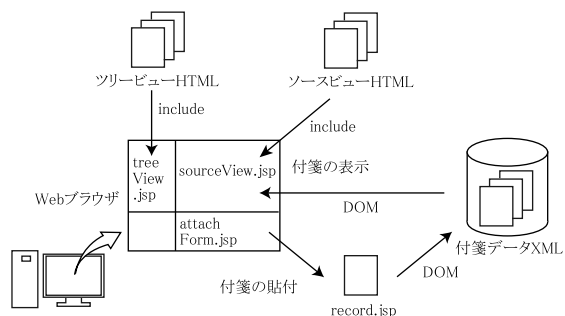


図 9 システム構成 2

Fig. 9 System configuration 2.

置) の欄が入力される。付箋を付ける位置はオブジェクト上での位置である。残りの `memo`, `title`, `name`, `mail` の欄がユーザにより入力され、フォームデータが送信されると、そのデータを付箋データ XML に記録する。

SourceView.jsp は付箋データ XML からデータを読み出して付箋部分の HTML を生成し、これとソースビュー HTML を合わせてソースフレームに出力する。ソースビューフレームに出力される HTML には JavaScript が入っており、これが CSS の `top` 属性と `left` 属性を操作することで付箋を正しい位置に表示する。

3.6 利 用 例

JSPIE2 の利用例を挙げ考察する。ユーザ A が作成したプログラムを、ネットワーク越しにユーザ B とユーザ C がレビューするというケースを考える。この作業の流れを図 10 に示す。

(1) ユーザ A が作成したプログラムを JSPIE2 に登録する。このとき、覚書やユーザ B やユーザ C に伝えたいことなど、プライベートな内容の情報や短期的な情報などは付箋に書き付けておく。(フォーマルな情報や長期的な情報はコメントや別の文書に書く。)

(2) ユーザ B, ユーザ C がレビューし、気になることを付箋で書いて付ける。ユーザ B が先にレビューしたとすると、ユーザ C はソースプログラムやユーザ A の付箋だけでなく、ユーザ B が付けた付箋も見ることになる。

(3) ユーザ A はユーザ B とユーザ C がつけた付箋を見ながらプログラムを修正し、もう一度そのプログラムを JSPIE2 に登録する。修正前付いていた付箋は、その修正前のプログラムに付いたままで、修正後の方は何も付いていない状態になる。このとき、ユーザ B とユーザ C の指摘に、どのように対応したかや新たに気付いたことなどを付箋で付けておく。

(4) ユーザ B, ユーザ C は修正内容を確認する。十分な内容でなければ、この手順をさらに繰り返す。

目の前に作成者がいる場合は、間違いを指摘したり、気になることを伝えるのことは難しくないが、ネットワーク経由の場合などで、そうできないことも多い。このケースでは JSPIE2 を利用することで作業を効率よく進められ、有効な利用例だと考えられる。

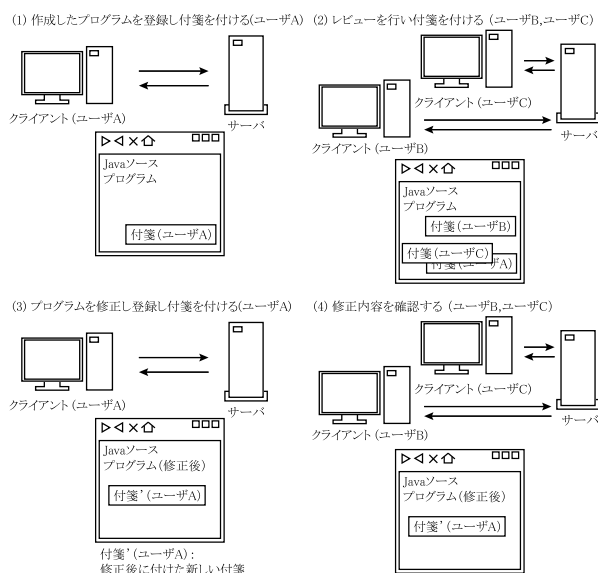


図 10 利用例

Fig. 10 An example of use.

4. おわりに

付箋によるコミュニケーションについて議論し、付箋によるコミュニケーション機能を備えたソフトウェアブラウザ JSPIE2 を作成した。XML に関する技術 (XML, XSLT, DOM) や Web に関する技術 (JSP, JavaScript, CSS) を多く活用したことで、Web ブラウザを通してソースプログラムに付箋を付けるというシステムを実現した。これによりソースプログラム上で付箋を利用したコミュニケーションが可能となり、プログラムを再利用する場合や保守する場合など、プログラム理解を支援に有効に利用できる。利用法としては次のような場合が考えられる。

共同開発

3.6 節で例を挙げたネットワーク経由のレビューなど、共同開発においてプロジェクトのメンバーの間の情報共有や情報交換に JSPIE2 を使用する。JSPIE 上で行われるコミュニケーションによって効果的に開発を進めることができる。

ソースプログラムの公開

ソースプログラムを公開する場合、単にソースプログラムを公開するだけでなく、JSPIE2 を使って公開する。この場合、プロジェクト外の閲覧者も、Web ブラウザでプログラムソースが閲覧し、コミュニケーションに参加できるので、より多くの情報を集めることができる。

チュートリアル

プログラムのチュートリアルを行う場合、サンプルプログラムを用意しその解説を行うという方法がよく行われるが、このとき JSPIE2 を使いサンプルプログラム表示し、解説を付箋でつけておく。付箋で簡単に解説が付けられるので、チュートリアルが作成しやすい。学ぶ側は付箋が付けられたサンプルプログラムを見ることで、より理解が進む。また学ぶ側も疑問点を付箋で付け、教える側とのコミュニケーションをとることも考えられる。このように JSPIE2 を用いれば、チュートリアルを効

果的に行うことができる。

今後の課題としては以下の 2 点が挙げられる。

ソースプログラムの変更に対応

JSPIE2 ではソースプログラムを登録後に変更することはできない。ソースプログラムを変更する場合、変更後にもう一度登録することになるが、このとき変更前のソースプログラムに付けられていた付箋データを移して使用することはできない。よって変更されたソースプログラムにはゼロから付箋を付けることになる。ソースプログラムの変更によって必要なくなる付箋もあるが、そのまま残して引き続き使いたい付箋もあるため、これをどう引き続き使えるようにするかがこの課題である。

他の文書への対応

現在対象としているのは Java ソースプログラムのみであるが、他の文書でも、付箋を付けることで情報の共有や交換を行うシステムは有効に利用できる。例えば、他のプログラミング言語のソースプログラム、ソフトウェアに関連する文書、一般的な文書などが考えられる。他の文書に対応するためには、Java ソースプログラムの場合の XML リポジトリのような、その文書の要素を分類しタグ付けした XML ファイルが必要になる。ある文書に対して、このような XML ファイルを生成するシステムができれば、JSPIE2 の大部分をそのまま利用して対応することが可能である。

JSPIE2 は以下のアドレスで公開している。

<http://www.sapid.org/sample/JSPIE2/>

謝 辞

情報処理振興事業協会による「次世代ソフトウェア開発事業」において、東京大学大学院情報理工学系研究科 米澤明憲教授と東京工業大学大学院情報理工学研究科 柴山悦哉教授に貴重なアドバイスを頂きました。また、立命館大学理工学部 丸山勝久助教授、株式会社ピクト、財団法人ソフトウェア工学研究財団、名古屋大学大学院情報科学研究科 阿草清滋教授をはじめとする Sapid プロジェクトの皆さんに感謝します。

文 献

- [1] Sapid <http://www.sapid.org/>
- [2] Hachisu Yoshinari, Yamamoto Shinichirou and Agusa Kiyoshi: "A CASE Tool Platform for an Object Oriented Language", IEICE Trans. on Information and Systems, Vol.E82-D, No.5, pp.997-984 (1999/5)
- [3] 吉田 一, 山本 晋一郎, 阿草 清滋: "XML を用いた汎用的な細粒度ソフトウェアリポジトリの実装", 情報処理学会 OO2002 シンポジウム, pp.83-90 (2002/8)
- [4] T.Bray, J.Paoli, C.M.Sperberg-McQueen: Extensible Markup Language (XML) 1.0 (Second Edition), W3C Recommendation (6 October 2000), <http://www.w3.org/TR/REC-xml>
- [5] Adobe <http://www.adobe.com/>
- [6] Jakarta Project <http://jakarta.apache.org/>