Consistent Web Site Updating Based on Bidirectional Transformation

Keisuke Nakano*
The Education and Research Center for Frontier Science
The University of Electro-Communications
1-5-1 Chofugaoka, Chofu-shi, Tokyo, Japan
ksk@cs.uec.ac.jp

Zhenjiang Hu
GRACE Center
National Institute of Informatics
2-1-2 Hitotsubashi, Chiyoda-ku, Tokyo, Japan
hu@nii.ac.jp

Masato Takeichi
Department of Mathematical Informatics
University of Tokyo
7-3-1 Hongo, Bunkyo-ku, Tokyo, Japan
takeichi@mist.i.u-tokyo.ac.jp

Abstract

A transformation-based Web site can keep the content of a Web site consistent by furnishing a single database and a set of transformation programs, each of which generates a Web page. However, when someone notices an error or stale content on a Web page in this style of Web site construction, the Web site maintainer must access a possibly huge database to update the corresponding content.

This paper proposes a new approach to Web site construction based on bidirectional transformation*, together with a practical updating system,* Vu-X*. Because of the bidirectionality of the transformations, users can directly modify a generated Web page rather than accessing the database and the modification is automatically reflected in the database. The* Vu-X *system is also implemented as a Web server so that users can edit it in WYSIWYG style on their Web browsers. Since the* Vu-X *system employs a bidirectional transformation language* Bi-X *to describe bidirectional transformations, we can obtain both transformations only by specifying a transformation in one direction.*

Keywords Web site maintenance, XML transformation, Bidirectional transformation, Web site generation, HTML editor

1 Introduction

Maintaining a Web site is costly and time-consuming [15], which requires consistent and frequent updating of the content to maintain its freshness and quality. The maintenance problem generally covers any type of update to a Web site, including recognizing its structure and modifying its content. A more specific problem has to do with replicated information: it is onerous for Web site administrators to avoid inconsistency between all Web pages provided on their own Web site because some of the pages may contain the same information, which should be synchronized.

A natural solution to this problem is to prepare a single database that lumps the information for all Web pages on the Web site, and to consider a Web site as a set of views that are generated from some existing raw data (database) [2, 8, 9] through transformation programs. We call this *transformation-based Web site construction*. This helps to easily keep all Web pages consistent since all pages are generated from the same database. Figure 1 shows a simple instance of such transformation-based Web site construction. The database is usually constructed in XML format,

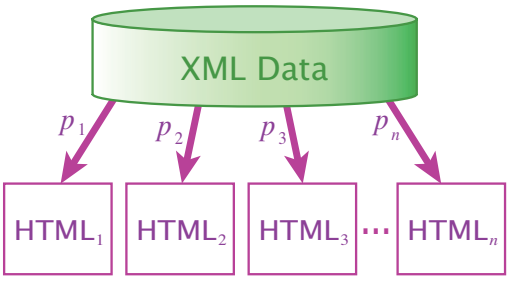

Figure 1. Transformation-based Web site construction

*Corresponding author.

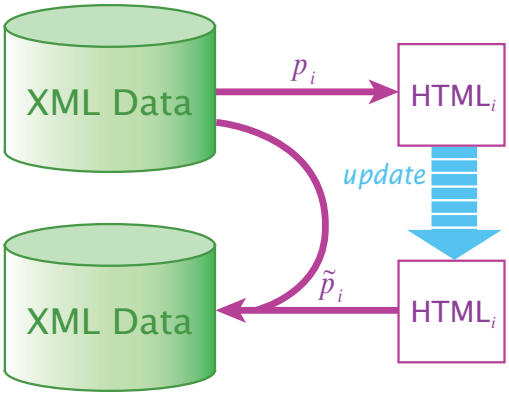

Figure 2. Bidirectional transformation between XML data and HTML sources

which is convenient for describing structured information. Transformation programs $p_1, \ldots, p_n$ are used to generate an HTML source corresponding to a Web page from the XML data. Typically, these programs are written in XSLT [19].

Transformation-based Web site construction, however, is not as good at frequently updating its content as simple Web site construction that consists of a set of independent HTML files. When someone points out an error or stale content on a page at the Web site, it is not easy to find which part of the database should be updated, particularly when the database is very huge or when the maintainer is not the person who designed the database. It would be nice if we could correct errors, update content, or change the layout just by editing the HTML view of the Web page, which results from the transformation, instead of accessing the database to update it. Furthermore, it would be ideal if we could edit Web pages on Web browsers in WYSIWYG style.

This paper proposes a new approach to solving the Web maintenance problem based on *bidirectional transformation*, together with a practical updating system, Vu-X. We divide the problem into two parts, i.e., how to edit Web pages on Web browsers in WYSIWYG style and how to reflect modifications of the HTML source to the original database. The former is attained by adding JavaScript to the HTML source. The latter, which is more difficult, is accomplished by a novel use of *bidirectional transformation* [10, 13, 17], which is a new technique for synchronizing data.

Bidirectional transformation involves a pair of transformations: forward transformation maps one data structure called a source onto another called a view, and backward transformation reflects changes in the view to the source. Bidirectional transformation has many practical applications including the synchronization of replicated data in different formats [10], presentation-oriented structured documents development [13], interactive user interface design [18], coupled software transformation [14], and the well-known mechanism for *view updating*, which has been intensively studied in the database community [3, 5, 11, 12, 16].

In our approach shown in Figure 2, forward transformation p_i transforms XML data into an HTML source, while backward transformation $\tilde{p}_i$ reflects a modification of the HTML source to the database. Since the HTML source generated by forward transformation generally does not have sufficient information to construct the corresponding XML data, backward transformation takes not only an updated HTML source but also the original XML data to generate the updated XML data. These two transformations, p_i and $\tilde{p}_i$, should be consistent with each other. We chose a bidirectional XML transformation language Bi-X [17] to specify the bidirectional transformation. One advantage of using Bi-X is that a program written in the Bi-X language only describes forward transformation and the corresponding backward transformation is automatically derived.

We have implemented a Web site updating system, Vu-X [1], which facilitates the maintenance of a Web site based on bidirectional transformation. The Vu-X system has five main features:

- We can easily maintain consistency between multiple Web pages on the Web site by means of transformation-based Web site construction.
- We can edit a generated HTML source itself based on bidirectional transformation. The modifications are automatically reflected to the XML data.
- We do not need to specify two transformations to attain a bidirectional transformation using the Bi-X language. It suffices to only specify forward transformation. Backward transformation consistent with the forward one is automatically derived.
- We can edit both HTML sources and Bi-X programs in WYSIWYG style. The Vu-X system provides two editing modes, content updating and code editing, the former allowing us to edit HTML sources and the latter Bi-X programs.
- We do not need to install any special software to use the Vu-X system. The editor for HTML sources and Bi-X programs is provided as a Web application, i.e., it runs on Web browsers. WYSIWYG editing on Web browsers is accomplished with the help of JavaScript.

[1] The system can be played with Web browsers such as Internet Explorer and Firefox and its URL is `http://www.psdlab.org/vux/start.html`. You may login with your email address for ID and no password is required to try our system while possible operations are limited for security.

The remainder of this paper is organized as follows. We start by giving an overview of the Vu-X system with some examples of practical situations in Section 2. After summarizing the basic ideas of bidirectional transformation and the bidirectional transformation language Bi-X used in our system in Section 3, we explain the architecture and the implementation of our Vu-X Web site updating system in Section 4. We discuss related work in Section 5, and conclude the paper in Section 6.

2 Overview of Vu-X System

This section explains how users maintain a Web site with the Vu-X system, giving some examples of practical situations. The Vu-X system is implemented as a Web application that runs on Internet Explorer or Firefox. Users are not required to install special software to employ the Vu-X system.

Users start the Vu-X system for updating their own Web sites by logging in with a correct password. The XML data and Bi-X programs for the Web site are registered through the Vu-X system. After logging in, they open a small window to select the name of the XML data and Bi-X program that corresponds to the Web page they want to edit. Here, users have to choose either of two modes, content updating or code editing. In the former, they can update information in the XML data through editing the corresponding HTML source. In the latter mode, they can edit the Bi-X program or create a new Bi-X program from scratch. Both modes provides a WYSIWYG editor that runs on Web browsers. We generally employ the content-updating mode to edit the content of the Web site and the code-editing mode to change or define the layout.

When users select the name of the XML data and Bi-X program, the corresponding Web page is loaded on their Web browser as seen in Figure 3 no matter which mode is chosen. The Web page to be updated is displayed at the lower right of the Web browser. The HTML source for the Web page is generated from the designated XML data and Bi-X program. It contains a JavaScript code so that users can edit it on the Web browser and reflect the change to the XML data or Bi-X program. Let us look at some situations involving editing or updating Web pages.

Changing email address Assume that users want to change an email address that occurs many times on the Web page. The email address may occur at the footer of all Web pages on the Web site. If all occurrences are generated from the same part of the XML data, it suffices for users to click and edit just one of the occurrences on the Web browser. Then, the other occurrences of the email address are automatically updated because the corresponding part of the XML data is updated through backward transformation

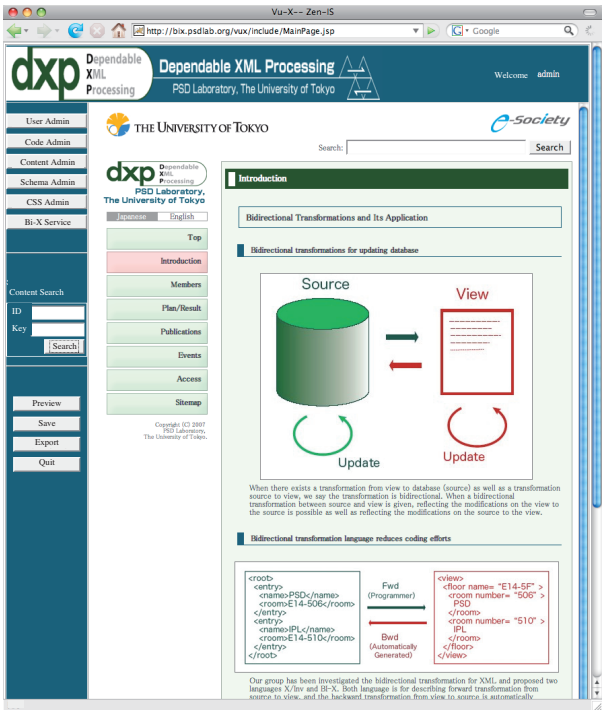

Figure 3. Screenshot of Vu-X system

and forward transformation modifies all occurrences of the email address. This editing is done in the content-updating mode.

Updating publication list Assume that users want to update a list of publications sorted by year in descending order. The XML data contain the information on all publications including the title, author(s), and year of publication for all entries. The Bi-X program describes a (forward) transformation from the XML data to an HTML source for the Web page for the publication list. The list is represented as a `ul` component in the HTML source. When users click it, a small window pops up for them to edit the items in the list (Figure 4). The window contains a list of items in the `ul` component and several buttons for editing functions and reflecting the updates. The Vu-X system supplies three editing functions for a `ul` component, i.e., modification to an item, deletion of an item, and insertion of a new item. Users can also copy and paste an existing item. When they want to modify the year of a publication, they click the publication item to open another editing window. Since the item consists of multiple HTML components as

```
<li>
  <span>K. Nakano</span>...<span>2006</span>
</li>,
```

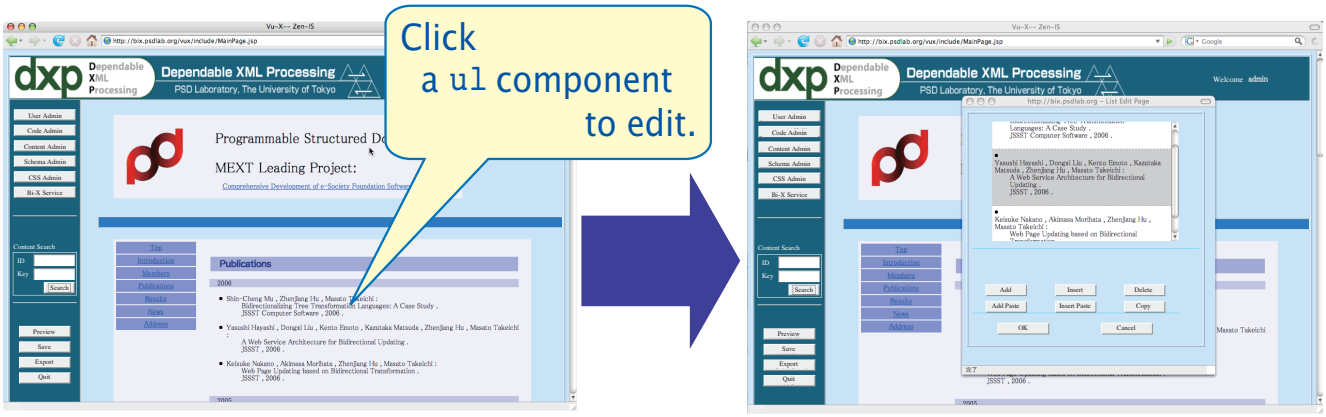

Figure 4. Editing a `ul` component on the Web browser

the small window that has opened graphically shows these components, each of which opens another editing window for the HTML component by clicking it. Users click the component `<span>2006</span>`, which represents the year of publication, to modify the text into `2005`. After the modification, they click the "OK" button to close each window and reflect the modification in the previous small window. When all small windows are closed, the Web page on the Web browser is reloaded. Interestingly, the occurrence of the publication in the list will be automatically moved because the year of publication is changed. This is, as far as we are aware, one of the unique functions that cannot be found in other Web site maintenance systems. This editing is done in the content-updating mode.

Changing layout of Web page Assume that users want to change the layout of a Web page from a `table`-based style into a `div`-based one. In a transformation-based Web site construction, this kind of information about layout styles is specified by the transformation program but not the XML data. Therefore, what users should do in this situation is to modify the transformation program. The Vu-X system provides a function to edit Bi-X programs with a graphical user interface. Users can place arbitrary HTML components on the Web page to create a Bi-X program that generates the HTML components by forward transformation. An HTML component that depends on the XML data is added or modified through the Bi-X code builder, which will be explained in Section 4. Since the Vu-X system shows how the Web page is generated from the actual XML data when editing a Bi-X program, users can easily update the layout of the Web page. This editing is done in the code-editing mode.

3 Bidirectional Transformation and Bi-X Language

This section summarizes the general concept of bidirectional transformation and gives a brief introduction to a bidirectional XML transformation language, Bi-X, which serves as a basis for our Vu-X Web site updating system.

3.1 Bidirectional Transformation

Bidirectional transformation [10], originating from the view-updating technique in the database community [3, 5, 11, 12, 16], involves a pair of transformations between a *source* and its *view*: the first transformation, called *forward transformation*, maps sources to views and is used to reflect changes in a source in its view. The second transformation, called *backward transformation*, maps views to sources and is used to reflect changes in a view in its source. Since a view generally has less information than its source, backward transformation takes not only an changed view but also the original source to generate a source corresponding to the new view.

It is difficult to describe the two transformation programs for well-behaved bidirectional transformation because they must be consistent. When a view is generated from a source by forward transformation, backward transformation must generate the same source for the view and the source. Therefore, we need to simultaneously maintain forward and backward transformation. Many bidirectional transformation languages, such as Lens [10], X/Inv [13], and Bi-X [17], have been proposed to support easy bidirectional programming. Even if our framework does not depend on any specific bidirectional language, we chose Bi-X (as will be discussed later in detail), for which we developed over 20,000 lines of code in our experiment.

```
<xseq>
  <xchild/>
  <xmap>
    <xif><xwithtag>author</xwithtag>
      <xid/>
      <xconst/>
    </xif>
  </xmap>
</xseq>
```

Figure 5. Bi-X **program**

3.2 Bi-X: Bidirectional Transformation Language

The Bi-X bidirectional transformation language has been presented by Liu et al. [17]. A program written in Bi-X specifies a bidirectional transformation between two XML documents (or XML fragments). The main feature of Bi-X is that all programs have two semantics corresponding to forward and backward transformation. It suffices to write a Bi-X program only as forward transformation. Programmers do not need to consider the consistency between forward and backward transformation to specify bidirectional transformation. See [17] for details on the Bi-X language and its theoretical background.

A Bi-X program is written in XML format. Figure 5 shows a small example taken from a Bi-X program. It can informally be read as follows: sequentially transform the data by first obtaining all their children and then for each child retaining these if they have the tag of `author` and otherwise omitting them. On one hand, forward transformation using this program takes an XML document in which the root element has some child elements whose tag name is `author` and returns a list that consists of all `author` elements. For example, when an XML document S is

```
<paper>
  <title>Vu-X system</title>
  <author>Keisuke Nakano</author>
  <author>Zhenjiang Hu</author>
  <section>...</section>
</paper>
```

which is offered as an input of the forward transformation of the program, output V is

```
<author>Keisuke Nakano</author>
<author>Zhenjiang Hu</author>.
```

On the other hand, if a new author

```
<author>Masato Takeichi</author>
```

is inserted into V, backward transformation will return an XML document that is obtained by adding the `author` element to S.

```
<xlet><var>doc</var>
  <xseq>
    <xconst><root/></xconst>
    <xpar>
      <xseq>
        <xvar>doc</xvar><xchild/>
      </xseq>
      <xvar>doc</xvar>
    </xpar>
  </xseq>
</xlet>
```

Figure 6. Bi-X **program with duplication**

Consider another example of a Bi-X program in Figure 6. This illustrates that a Bi-X program can deal with duplication. It is generally difficult to attain bidirectional transformation for a program including duplication because a duplicated view should be synchronized for every update. Duplication in a Bi-X program is described by binding a variable (`doc` in the example program) and using it in many places. When the program takes XML source S of

```
<body>
  <section>A</section>
  <section>B</section>
</body>,
```

output V of forward transformation is

```
<root>
  <section>A</section>
  <section>B</section>
  <body>
    <section>A</section>
    <section>B</section>
  </body>
</root>.
```

If either text `B` in V is changed into `C`, the backward transformation of the program will return S in which `B` is replaced with `C`. Applying forward transformation to XML again, we will have XML view V in which both texts `B` in V are synchronized to be text `C`.

Even though Bi-X language helps us to describe a consistent bidirectional transformation, it is still difficult to write a large Bi-X program for practical use. To facilitate Bi-X programming, the Vu-X system provides a graphical user interface to generate a Bi-X program. Users may also employ a interpretation tool from an XQuery program to a Bi-X program [17].

4 Vu-X: Web Site Updating System

The Vu-X system provides an integrated environment to maintain a Web site based on bidirectional transformation.

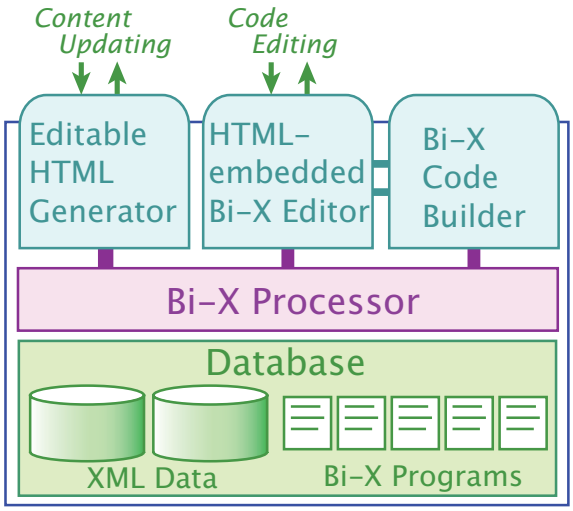

Figure 7. Architecture of Vu-X system

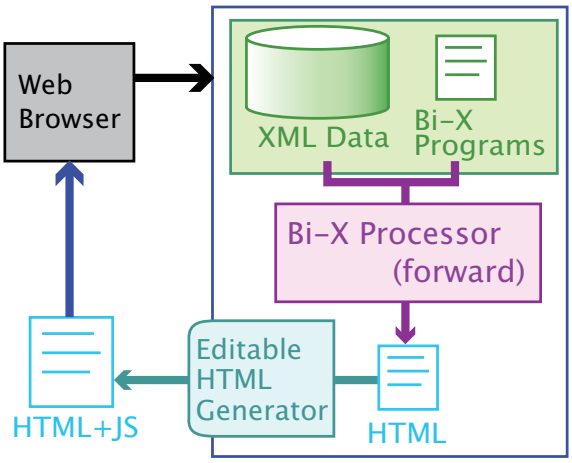

Figure 8. Initial action in content-updating mode

It is a kind of transform-based Web site as shown in Figure 1 where transformations $p_1, \ldots, p_n$ are bidirectional. This section presents the architecture of the Vu-X system and explains how the two updating modes introduced in Section 2 are implemented in the Vu-X system.

4.1 Architecture of Vu-X system

The Vu-X system is implemented as a Web server based on Apache Tomcat so that users can update their own Web site through Web browsers and special software is required to install it.

Figure 7 outlines the architecture of the Vu-X system. The system consists of five parts: the XML and Bi-X database, Bi-X processor, editable HTML generator, HTML-embedded Bi-X editor, and Bi-X code builder. We will explain these one by one.

XML and Bi-X database It stores the XML data and Bi-X programs registered by users. Each XML data or Bi-X program is accessed by authorized users with its identifier and read/write instruction. Note that all Web pages on the Web site are generated by the forward transformation of the Bi-X programs from the XML data.

Bi-X processor It executes the forward/backward transformation for a given Bi-X program. The forward transformation takes the identifier of the XML data and returns an HTML source. The backward transformation takes the identifiers of the XML data and an updated HTML source and returns updated XML data.

Editable HTML generator It adds a JavaScript code to a given HTML source and creates an *editable HTML*, which allows us to edit it on Web browsers. The obtained editable HTML and the original HTML source have the same view on Web browsers, except when users click a component of the view of the editable HTML on the Web browser and a small window pops up for them to edit the component. This part is employed in the content-updating mode.

HTML-embedded Bi-X editor It provides a WYSIWYG HTML editor that runs on Web browsers. Users can also insert multiple Bi-X programs together with HTML components. This editor can generally create an HTML source that contains Bi-X programs, called an *HTML-embedded* Bi-X *program*, which will be discussed in detail in the following subsection. This part is employed in the code-editing mode.

Bi-X code builder It provides a graphical user interface to enable Bi-X programs to be easily constructed. Users can write a Bi-X program using actual XML data that is an input of forward transformation. The Bi-X code builder shows a partial result of the transformation for each fragment of the Bi-X program. This part is employed in the code-editing mode.

The Vu-X system provides two modes for updating a Web site, content updating and code editing, as shown in Section 2. The following subsections explain how these updating modes are implemented with these five parts of the Vu-X system.

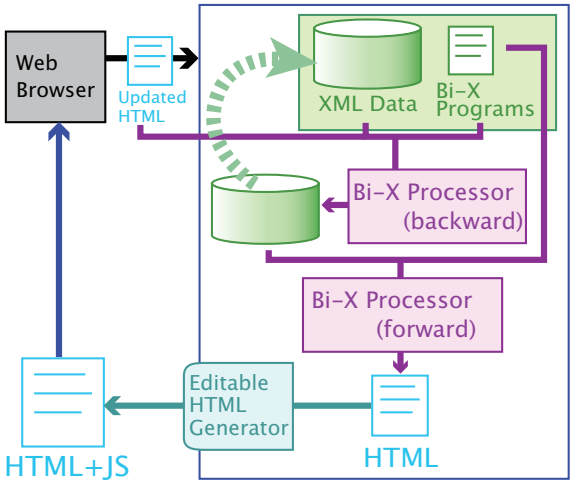

Figure 9. Update reflection in content-updating mode

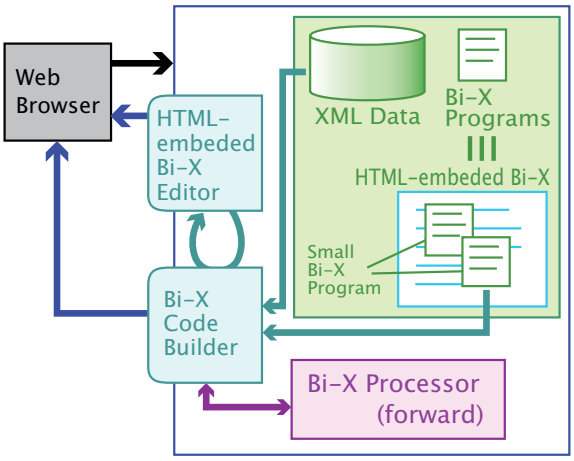

Figure 10. Code-editing mode

4.2 Content Updating

Figure 8 outlines an initial action in the content-updating mode. Content updating starts with a user's request to the Vu-X system with the identifiers of the XML data and a Bi-X program. The Vu-X system executes forward transformation of the Bi-X program for the designated XML data through the Bi-X processor to obtain a transformation result that is an HTML source. Next, the editable HTML generator of the Vu-X system adds a JavaScript code to the HTML source so that users can edit it on their Web browsers. The JavaScript code does not change the layout on the Web browsers.

The JavaScript code plays two roles. The first is to allow us to edit in WYSIWYG style for several kinds of HTML components. Possible editing depends on the kind of clicked component. For example, a text component allows us to edit its text data and a `ul` component allows us to modify an item on the list, delete an item, or insert a new item as explained in Section 2. The second role of the JavaScript code is to transmit an updated HTML source to the Vu-X system by clicking the "OK" button to finish editing. The transmitted HTML source does not include the added JavaScript code.

Figure 9 shows how the Vu-X system treats the updated HTML transmitted from the Web browser. The updated HTML is passed to the Bi-X processor to execute backward transformation of the Bi-X program with the original XML data to update the XML data in the database. Then, the Vu-X system again calls the Bi-X processor for forward transformation of the Bi-X program with the updated XML data. It generates a new HTML source corresponding to the updated XML data. Note that the HTML source is not always the same as the HTML source updated by users because the edited part may influence the other part in the HTML source where the Bi-X program contains duplication. After forward transformation, the Vu-X system generates the editable HTML source by again adding a JavaScript code to the obtained HTML source. Then, users can edit the other component on the Web page to update the content of the XML data.

4.3 Code Editing Mode

Like content updating, code editing starts with a user's request to the Vu-X system with the identifiers of the XML data and a Bi-X program[2]. In the code-editing mode, users edit an *HTML-embedded* Bi-X *program* through a graphical user interface using the HTML-embedded Bi-X editor of the Vu-X system. An HTML-embedded Bi-X program has a form analogous to PHP or JSP, e.g,

```
<html>
  <head><title>Vu-X</title></head>
  <body>
    <h1>Vu-X system</h1>
    <bixpro>... Bi-X program1 ...</bixpro>
    <p>Here is a paragraph.</p>
    <hr/>
    <bixpro>... Bi-X program2 ...</bixpro>
  </body>
</html>
```

where `Bi-X program1` and `Bi-X program2` are Bi-X programs. An HTML-embedded Bi-X program represents

[2]The code-editing mode allows us also to create a new Bi-X program from scratch.

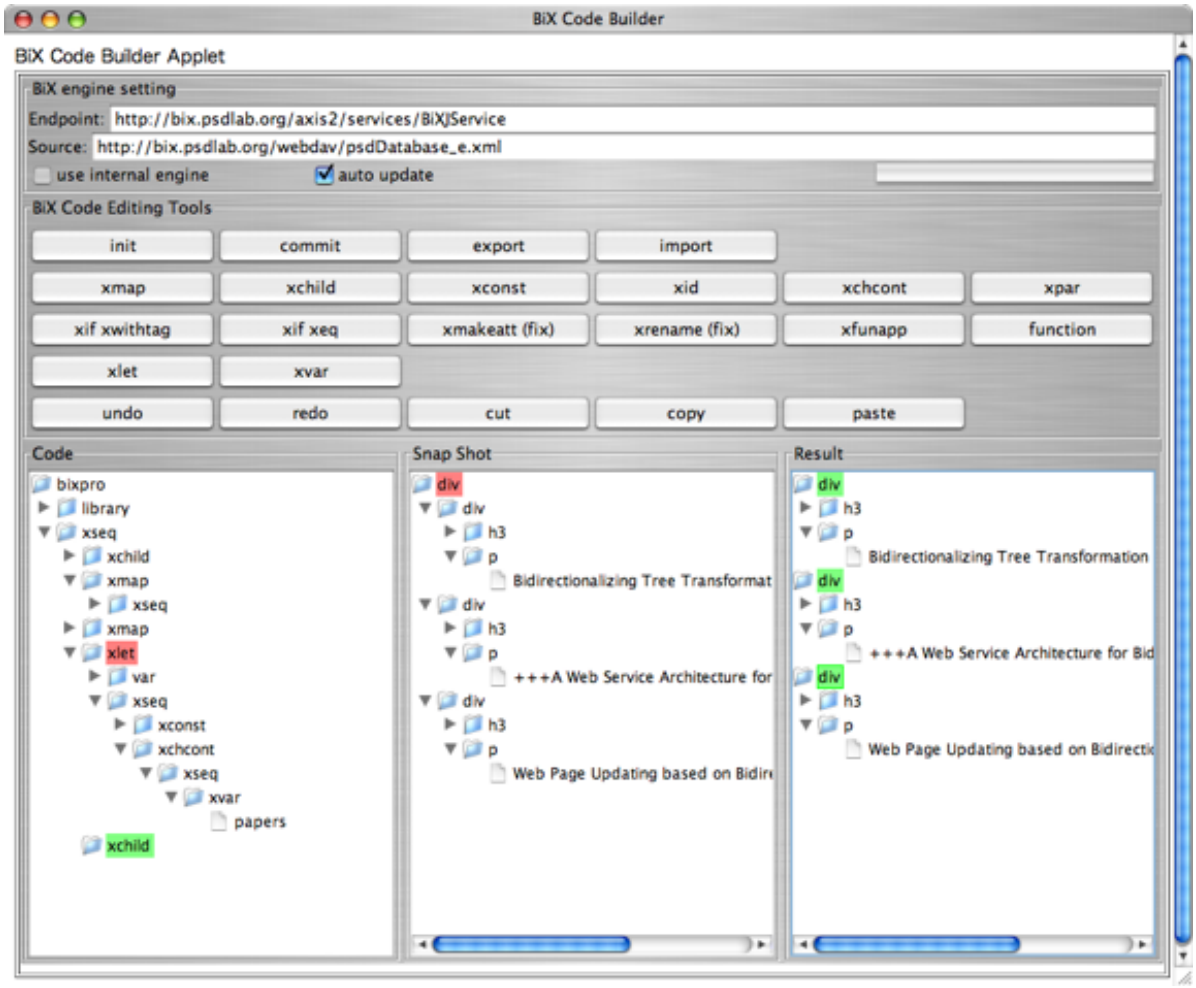

Figure 11. Bi-X **code builder**

a bidirectional transformation whose forward transformation generates an HTML source obtained by replacing each `bixpro` component in the program with the result of the forward transformation of the Bi-X program in the component. The backward transformation can be given in a similar way. The Vu-X system provides a transformation tool from an HTML-embedded Bi-X program to a single Bi-X program. Both the Bi-X program and the HTML-embedded Bi-X program are stored as a pair in the database of the Vu-X system.

The code-editing mode supplies WYSIWYG HTML editors such as Adobe Dreamweaver [6] and Microsoft Expression Web [7]. These have two differences from the existing WYSIWYG HTML editors. The first is that users can add `bixpro` components to create an HTML-embedded Bi-X program. The Bi-X program in the `bixpro` component is edited by the Bi-X *code builder*, which will be explained later. The second difference is that the editor is implemented as a Web application using JavaScript. Users do not have to install any special applications to edit it.

When a new `bixpro` component is created or the existing `bixpro` component is clicked, the Bi-X code builder shown in Figure 11 is invoked to edit a Bi-X program to be inserted. The Bi-X code builder is implemented as a Java applet that communicated with a JavaScript code in the HTML-embedded Bi-X editor.

The Bi-X code builder helps users to write a Bi-X program using an actual input XML. The Bi-X code builder has several buttons and three tree views. Each button corresponds to a construct in the Bi-X language. When the construct requires arguments, the Bi-X code builder opens a prompt window to provide these. The tree view at the left shows an abstract syntax tree of the Bi-X program that users want to edit. There are two cursors colored green and red in the view. The green cursor stands for the current editing position. They can easily place a new Bi-X construct imme-

diately behind the green cursor by clicking the corresponding button. The red cursor represents the previous editing position. The tree view in the middle shows an XML tree, which is a transformation result at the position designated by the red cursor in the Bi-X program. The tree view at the right shows an XML tree, which is a transformation result at the position designated by the green cursor in the Bi-X program. After editing a Bi-X program on the Bi-X code builder, the editing stage returns back to that of the HTML-embedded Bi-X editor. The `bixpro` component occurs as an HTML component, which is the transformation result of the inserted Bi-X program with the XML data, by sending a request for forward transformation to the Bi-X processor in the Vu-X system. Users can thus edit the HTML-embedded Bi-X program in WYSIWYG style.

When Users click the "Save" button on the HTML-embedded Bi-X editor, the HTML-embedded Bi-X program is converted into a single Bi-X program so that it can be used in the content-updating mode of the Vu-X system.

5 Discussion

This section discusses what the features are of the Vu-X system. As explained in Section 1, there are several features of the Vu-X system that maintain a Web site.

First, users can easily maintain consistency between multiple Web pages on the Web site. The Vu-X system employs a kind of transformation-based Web site construction furnished with XML data and Bi-X programs, each of which generates a Web page from the XML data. Since every piece of information shared by multiple Web pages occurs once in the XML data, all occurrences of that information in Web pages are definitely synchronized.

Second, users can edit the generated HTML source itself based on bidirectional transformation. Modifications to the HTML source are consequently reflected in the XML data by the backward transformation. An HTML source is modified with a WYSIWYG editor provided by the Vu-X system as the content-updating mode.

Third, users do not have to specify both forward and backward transformations to attain a bidirectional transformation thanks to the bidirectional transformation language Bi-X. It suffices to only specify forward transformation. The corresponding backward transformation is automatically derived. All transformation programs in the Vu-X system that generate Web pages are written in Bi-X.

Fourth, users can easily generate and edit a Bi-X program with a WYSIWYG editor provided by the Vu-X system as the code-editing mode. They construct a Bi-X program step by step by discerning the transformation result.

Finally, users do not have to install any special software to work with the Vu-X system because it is implemented as a Web application. The WYSIWYG editor in both content updating and code-editing modes runs on Web browsers such as Firefox and Internet Explorer.

The framework of the Vu-X system can be considered as a combination of a WYSIWYG HTML editor and bidirectional transformation. Let us discuss the possibility of replacing them using the existing technology. First, let us consider replacing the WYSIWYG HTML editor. We have independently developed the WYSIWYG HTML editor for the Vu-X system so that updates on the editor can be reflected back to the database. If the existing WYSIWYG HTML editor can be extended with a function for communicating with the database, the editor will be able to be used as a front-end of the Vu-X system. Next, let us consider replacing bidirectional transformation language. We employ the Bi-X language for the Vu-X system. There exist few languages for bidirectional transformation. A possible choice is Boomerang [4]. It suffices to write a single transformation in this language to specify a bidirectional transformation like that in the Bi-X language. However, if one wants to employ it as a back-end of the Vu-X system, it would be nice to have an editor that allows us to write a program in WYSIWYG style as is done by the HTML-embedded Bi-X editor and Bi-X code builder.

6 Related work

There are dozens of WYSIWYG HTML editors, such as Adobe Dreamweaver [6], Microsoft Expression Web [7], and Aptana Studio [1]. Most of these, however, require special software to be installed to employ them. Furthermore, they provide a tool for editing HTML sources one by one. Since they do not take into account dependencies between multiple HTML sources, the Web site maintainer has to consider their consistency, which is in sharp contrast to transformation-based Web site construction. Although the WYSIWYG HTML editor provided by the Vu-X system is less powerful than existing ones in the sense that only modifications that can be reflected back to the XML data are allowed, our approach can be applied to these by integrating them with bidirectional transformation based Web site construction as discussed in Section 5.

The Web site construction based on bidirectional transformation may remind readers of wiki or blogs, which allows their contents to be updated from Web browsers. Most of these only provides restricted editing functions and a limited number of templates for layouts. Furthermore, what these Web pages can share is restricted information, e.g., the title of the Web site and an index of recent entries between multiple Web pages in the Web site. The Vu-X system allows us to change the layout of Web pages and to easily specify what kind of information should be shared by describing bidirectional transformation.

Our design of the Web site maintenance system, Vu-

X, based on bidirectional transformation was greatly inspired by pioneering work [10, 4] on data synchronization based on bidirectional transformation, which originated from work on view updating [3, 5, 11, 12, 16] in the database community, where modifications to view could be reflected back to the original database. We borrowed this technique to enable Web site maintenance with one significant extension: editing operations can not only modify the view but also the transformation code, which has not been exploited before.

7 Conclusion

This paper has proposed a novel approach to Web site construction based on bidirectional transformation and presented an implementation of a practical updating system called Vu-X, which can keep the content of Web sites consistent by updating Web pages in WYSIWYG style on Web browsers. Since the Vu-X system employs the Bi-X bidirectional transformation language, it suffices to only specify forward transformation and the corresponding backward transformation is automatically derived. Furthermore, the Vu-X system provides a WYSIWYG tool to enable Bi-X programs to be easily constructed.

Even though the Vu-X system facilitates consistent Web site updating, it may be too difficult to introduce it to existing Web sites, because it would require them to be restructured with XML data and Bi-X programs from scratch to benefit from the Vu-X system. We are considering a "switchover" assistant tool for existing Web sites, which we intended to develop in future work.

Acknowledgments

We would like to thank the staff members of Dong Wu Information System Co. Ltd. who mainly implemented the Vu-X system following our design decisions. We are also grateful to Dongxi Liu, Yasushi Hayashi, Kento Emoto, Kazutaka Matsuda and Akimasa Morihata for their cooperation, and Hideya Iwasaki for his constructive comments on this paper.

References

[1] Aptana Studio. `http://www.aptana.com/studio/`.

[2] P. Atzeni, G. Mecca, and P. Merialdo. To weave the Web. In *VLDB '97*, pages 206–215, 1997.

[3] F. Bancilhon and N. Spyratos. Update semantics of relational views. *ACM Transactions of Database Systems*, 6(4):557–575, 1981.

[4] A. Bohannon, J. N. Foster, B. C. Pierce, A. Pilkiewicz, and A. Schmitt. Boomerang: resourceful lenses for string data. In *POPL '08: Proceedings of the 35th ACM SIGPLAN-SIGACT symposium on Principles of Programming Languages*, pages 407–419. ACM Press, 2008.

[5] U. Dayal and P. A. Bernstein. On the correct translation of update operations on relational views. *ACM Transactions of Database Systems*, 7(3):381–416, 1982.

[6] Adobe Dreamweaver CS3. `http://www.adobe.com/products/dreamweaver/`.

[7] Microsoft Expression Web. `http://expression.microsoft.com`.

[8] M. F. Fernandez, D. Florescu, J. Kang, A. Y. Levy, and D. Suciu. Overview of strudel - a Web-site management system. *Networking and Information Systems*, 1(1):115–140, 1998.

[9] M. F. Fernandez, D. Florescu, A. Y. Levy, and D. Suciu. Declarative specification of Web sites with strudel. *VLDB Journal*, 9(1):38–55, 2000.

[10] J. N. Foster, M. B. Greenwald, J. T. Moore, B. C. Pierce, and A. Schmitt. Combinators for bi-directional tree transformations: a linguistic approach to the view update problem. In *POPL '05: Proceedings of the 32nd ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages*, pages 233–246. ACM Press, 2005.

[11] G. Gottlob, P. Paolini, and R. Zicari. Properties and update semantics of consistent views. *ACM Transactions of Database Systems*, 13(4):486–524, 1988.

[12] S. J. Hegner. Foundations of canonical update support for closed database views. In *ICDT '90: Proceedings of the Third International Conference on Database Theory*, pages 422–436. Springer-Verlag, 1990.

[13] Z. Hu, S.-C. Mu, and M. Takeichi. A programmable editor for developing structured documents based on bidirectional transformations. In *PEPM '04: Proceedings of the 2004 ACM SIGPLAN Symposium on Partial Evaluation and Semantics-based Program Manipulation*, pages 178–189. ACM Press, 2004.

[14] R. Lämmel. Coupled software transformations (extended abstract). In *First International Workshop on Software Evolution Transformations*, pages 31–35, Nov 2004.

[15] T. Lau and J. Staczek. A contextual inquiry-based critique of the strudel Web site maintenance system. Technical Report TR-99-01-01, Department of Computer Science and Engineering, University of Washington, 1999.

[16] J. Lechtenbörger and G. Vossen. On the computation of relational view complements. *ACM Transactions of Database Systems*, 28(2):175–208, 2003.

[17] D. Liu, Z. Hu, and M. Takeichi. Bidirectional interpretation of XQuery. In *Proceedings of the 2007 ACM SIGPLAN Workshop on Partial Evaluation and Semantics-based Program Manipulation*, pages 21–30, 2007.

[18] L. Meertens. Designing constraint maintainers for user interaction. `http://www.cwi.nl/~lambert/`, Jun 1998.

[19] XSL transformations (XSLT) version 2.0. `http://www.w3c.org/TR/xslt20/`, 2006.