*対話的学習教材の作成支援環境について

林 康史　　胡 振江　　武市 正人
東京大学大学院 情報理工学系研究科
〒113-8656 東京都文京区本郷 7-3-1
e-mail: {hayashi, hu, takeichi}@mist.i.u-tokyo.ac.jp

概要

本研究では，ジャストシステム社の汎用 XML 作成・編集環境である xfy テクノロジーを用いた情報教育のための教材作成支援環境について考察する．学習者の入力に応じて文書を動的に変化させるようなコードを文書に埋め込み，学習者が対話的に学習できるような教材を作成することを目的としている．この環境は我々が提案している PSD (Programmable Structured Documents)の考えに基づき実装された PSD 処理インタフェースをプラグインとして xfy に組み込むという方法により実現された．教材は XML を元文書とするが，その作成・編集とコード埋め込みは Web ブラウザ上で効率よく作業をすることができる．電子的なチュートリアルや試験用紙のような例を用いて環境の有効性を実証する．さらに個々の対話的文書を埋め込みコードによって連携・参照させることにより，より大きな複合的対話的文書に進化させる枠組みについての提案を行う．

1. はじめに

Web 教材は近年増々広く使われるようになってきている．この種の教材には教育上様々な利点があるが，なかでも学習者が対話的に学習できる教材は主体的な学習を促す点で特に有効であると考えられている．このような教材は通常 HTML と JavaScript のようなスクリプト言語で書かれているが近年では，XSLT のような変換言語を用いて XML から HTML を作る方法も広く使われてきている．

これらの Web ブラウザ上で表示された教材を作るためには通常 HTML のタグや CSS 等を知っている必要があり，XML から変換する場合には XSLT のような変換言語を習う必要がある．さらに動的な文書にするためには JavaScript 等のスクリプト言語も学習する必要がある．これらの技術を習得することは一般の教材作成者にとっては必ずしも容易ではなく，既に習得している者にとっても，何らかの支援ツールを使わずに，この種の文書を作成するにはかなりの手間と時間を必要とする．したがってこれらの作業を支援するための環境を提供するツールの開発は，ウェブ教材がより広く有効に使われるためには極めて重要である．このようなツールに望まれる特長として次のことが考えられる．

- 構造化文書の要素の定義を知る必要なく，ウェブの表示画面上で通常のワープロと同じ感覚で作成できる．
- 対話的な性質を付加するためのコードがウェブの表示画面上で容易に書き込むことができる．

本研究の目的は，このような特長を持つ環境をジャストシステム社が開発した xfy テクノロジー[4]に，我々が提案している PSD(Programable Stractured Documents)[3]の考えを組み入れることにより構築することである．この環境を用いると次のような特長をもった教材を効率よく作成することができる．

- 学習者の入力に基づいて文章の内容が動的に変化する．
- 教科書やチュートリアルが学習者に対

* 本研究は文部科学省リーディングプロジェクトである「e-Society 基盤ソフトウェアの総合開発」の一環である「高信頼構造化文書変換技術」の中で行われた．

して個別化可能である．

- 教科書等に記述されたプログラムが，文書上でそのまま実行することができる．

本論文の構成は次のとおりである．2 節では，XML 文書の作成・編集に利用するジャストシステム社の xfy テクノロジーを紹介する．3 節では我々が提案する PSD の考えとその処理システムについて解説する．4 節で文書作成を容易にするために予め用意される構築ブロックについて説明し，さらに埋め込みコードの記述の仕方について解説する．5 節では対話的文書の作成方法を，具体例を用いて説明する．6 節では対話的文書のアプリケーションを組み合わせて一つのより大きな複合的アプリケーションを構築・管理する考えをまとめている．7 節では関連研究との比較について述べ，最後に 8 節で本研究のまとめと今後の課題について述べる．

2. XML 文書の作成・編集

XML 文書の作成・編集には，文字を編集する単純なエディタを用いることが考えられるが，変更に際しても構造を維持するためには，XML 文書に特化した XML エディタを利用することが望まれる．ジャストシステムの xfy テクノロジーは XML 文書用の作成･編集環境であり，2004 年 11 月 17 日に TP1(テクノロジー・プレビュー)版が発表され， 現在(2005 年 6 月)は TP2 版が無料でダウンロード可能である．xfy は複数の XML ボキャブラリを 1 つの文書上でシームレスかつ自然な方法で作成・編集することを可能にする．また柔軟なプラグインシステムによって拡張性に優れ，アプリケーション開発機能を持つ．

xfy は汎用的なソフトウェア基盤技術であり，特定の分野のアプリケーションを想定した物ではないが，情報教育の分野において，自分で簡単なアプリケーションが容易 に作れる等，教育的に非常に有益な技術であると思われる．電子的な教材開発に関して，xfy が持つ特に有用な特長は次の 2 つである．1 つは Vocabulary Connection Descriptor (VCD)というスクリプト言語によって XML 文書の HTML 文書への変換，表示だけでなくその Web ブラウザ表示画面上で直接 XML ソースの編集ができることである．もう 1 つは VCD ファイル内に，テンプレートや，文書の構成ブロックで内容が空の物を予め定義しておき，それを xfy ウィンドウのメインメニューから選択して追加し，内容を入力することにより容易に構成要素が追加できることである．これらの機能によって，文書の作成やアプリケーションの開発が効率よくできるようになっている．

3. PSD

3.1. PSD とは

構造化文書はプログラミング言語のデータ構造と共通の性質を有しており，関数型言語によるアルゴリズム記述が文書処理に適している．我々が提案している Programmable Structured Documents (PSD)は，自分自身を処理するプログラムを含む文書であり，プログラムの実行を手段として構造化文書処理を行う．すなわち，PSD においては，構造化文書をプログラミングにおける構造化データであるとみなし，文書の一部として埋め込まれているプログラムコードを何らかの方法で実行することにより，文書自体が編集・操作される．

PSD は例えばあるプログラムが埋め込まれた XML 文書であり，HTML に変換・表示した時に評価式は，評価ボタンとして表示され，そのボタンを押すと評価が実行され，結果として得られた XML 文書が Web ブラウザ表示されるというような処理が行われる．このような見方により，PSD 研究は構造化文書処理の分野にプログラミング言語の理論を適用し，信頼性の高い構造化文書処理のための基盤技術を開発することを目的にしている．

3.2. iDocument

PSD アプリケーションの一つとして我々が開発している iDocument は対話的に利用者が与えた入力に基づいて内容が動的に変化する文書であり，その応用先として教科書等の教育用アプリケーションを想定している．iDocument は PSD エディタ上で作成された XML 文書に対して，ユーザの入力を受け取りそれに応じた内容を動的に作成するコードを埋め込むことで対話的文書の作成を実現している．iDocument と併せて，文書をその構成部品である構築ブロックから容易に作成できる環境も開発しており，従来のチュートリアル，教科書にはない新しい有用性を備えた対話的文書の開発に用いることが可能である．我々は iDocument の応用事例として，

- iTutorial: 利用者が与えた入力に基づいて説明文が動的に変わるチュートリアル
- iTextBook: 演習問題を本文の説明の中に埋め込み，学習者が対話的に理解できる教科書
- iExam: 採点機能や，解答に応じたコメントの表示の機能を備えた電子的試験用紙

等を開発している．

3.3. PSD 評価機構

現在，iDocument を可能にする PSD 処理システムは PSD エディタと外部評価系を分離させ，PSD の処理を実現している．剪定木 DOM インタフェース[1]は，この処理を効率化するために我々が提案した機構の一つである．図 1 に剪定木 DOM インタフェースを使った PSD 処理の流れを示す．

まず PSD エディタは，評価式中の，XML 木の他の部分への参照を表す XPath 式を元に，必要な部分木だけを集めて 1 つの木（剪定木）を作成する．さらに評価式自体を含む

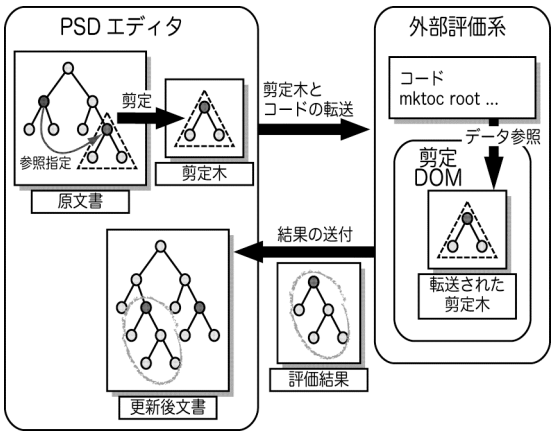

図 1. 剪定木 DOM インタフェース

要素と PSD 上で定義されたユーザ定義関数の定義された要素を含めて，1 つの XML として外部評価系に送る．送られた XML は外部評価系側で，評価に用いる言語で処理できるプログラムに変換され，実際の評価が行われる．評価結果は再び PSD エディタに戻され，DOM 部分木として評価式のあった部分に挿入される．

現在の実装は PSD エディタとして xfy を用い，外部評価系としては関数型言語 Haskell [2]の処理系を，そして XML から Haskell への変換には HaXml [5]を用いている．xfy と外部評価系をつなぐ剪定木インタフェースは xfy のプラグインとして組み込むことにより種々の PSD 処理が行えるようにしている．このシステムで， iDocument の各種事例に対して動作を確認し，性能向上に大きな効果があることを確認した．

4. 対話的文書の作成

4.1. 構築ブロックの利用

作成を容易にするために，iDocument は構築ブロックと呼ばれる基本部品から構成される．xfy テクノロジーの iDocument 作成支援にとって有用な特長の 1 つは，メニューバーから選んだ構築ブロックから文書を作成する環境を作成できることである．構築ブロックは VCD ファイルに予め記述しておく．例え

ば Web ブラウザ表示上でカーソルのある場所に相当する XML 文書の場所に，<paragraph/>を挿入する命令と，その命令のメインメニューへの割り当てを定義し，このブロックを Web ブラウザ上でどのように表示するかを記述する．これらの記述をしておくと，文書の作成者がメインメニューから paragraph 追加命令を選ぶと Web ブラウザ画面上で作成者に内容を入力することを促すカーソルが現れるようになる．このように，想定される文書のタイプに適したブロックそれぞれに対して必要な記述を VCD ファイルに予め環境として用意しておくことで，文書の作者は要素や属性の定義を知ることなくメインメニューからの選択と内容の入力だけの操作によって XML 文書の作成・編集が可能になる．

現在, iDocument 作成のために我々が用意した構築ブロック群は，基本ブロック，PSD 関連ブロック，アプリケーション固有ブロックの 3 種類に分類できる．基本ブロックに属するものは，一般的な文書によく用いられるものであり，section，paragraph，list，ordered list，sample code，image，table 等がある．PSD 関連ブロックは PSD を特徴付けるもので，Function Definition，PSD Expression 等である．Function Definition は <psdfun/> であり, これを XML に挿入する命令が選択されるとユーザ定義関数の定義の入力を促すカーソルと，定義を入力した後必要に応じて定義を見えないようにするためのボタンが Web ブラウザ表示画面上で表示される. PSD Expression は <psdexpr/>であり，これを XML に挿入する命令が選択されると埋め込みコードの入力を促すカーソルと共に，コードを実行するためのボタンと埋め込みコードを見えないようにするためのボタンが Web ブラウザ表示画面上に表示される．アプリケーション固有ブロックは，特定のアプリケーションで必要となるブロックであり，例えば，iExam においては options，answer 等がある．

4.2. 埋め込みコードの記述

現在の PSD システムでは，上で述べたように psdexpr タグで囲まれたテキストは埋め込みコードと認識され，評価ボタンと共に Web ブラウザ上で表示される. 評価ボタンを押すことで，評価式は外部評価系に送られ，そこで評価された結果が xfy に戻されて元文書に反映される. iDocument では利用者によって入力された内容を参照し,それに基づいた内容を動的に作成するコードを Web ブラウザ表示上で入力して XML 文書に埋め込むことで，文書に対話的性質を付加する．3 節で説明した PSD 処理システムは原理的には言語独立であるが現在は実験用として埋め込みコードのためのプログラミング言語として Haskell を用いている．XML 要素の Haskell によるデータモデリングは次のとおりである．

```
data Element =    Elem Name [Attribute] [Content]
data Content =     CElem Element
                 | CString String
```

評価式は予め定義されているライブラリー関数または PSD の中で定義されたユーザ定義関数を用いることができる．これらの関数は他の Haskell 評価式や XML ツリーの他の部分を引数としてとることができる．例えば選択式の問題などで用いる関数 grade は，正解，利用者の選んだ答え，正解ならば利用者に与えられる点数の 3 つの引数をとり，結果として利用者が得た点数を生成する関数とする．評価式は例えば grade(2, ../answer, 10) のような形をとる．ここで 2 は正解，../answer は利用者が与えた答えを内容として持つ要素の相対パス, 10 が正解の時の得点である. grade の定義は利用者により，次のように与えられる．

```
grade(x, (Elem “arg” a0 [CElem (Elem “answer” a1 [CString ans])]), y)
```

```
= (if x==ans
    then (Elem “psdresult” [] [CString y])
    else (Elem “psdresult” [] [CString “0”]))
```

ここで，(Elem “arg” a0 [CElem (Elem “answer” a1 [CString ans])]) は参照 ../answer から得られた Haskell で表現された剪定木で，ans は利用者によって与えられた解答である．grade 関数は x と ans を比較して等しければ y を内容に持つ psdresult 要素を，等しくなければ “0”を内容として持つ psdresult 要素を返す．

5. 具体例

5.1. iExam

我々は iDocument の事例として iTextBook，iTutorial，iExam 等を開発している．ここでは iExam を例に解説する．まず VCD の中に予め定義しておいた iExam のテンプレートをメニューから呼び出す（図 2）．

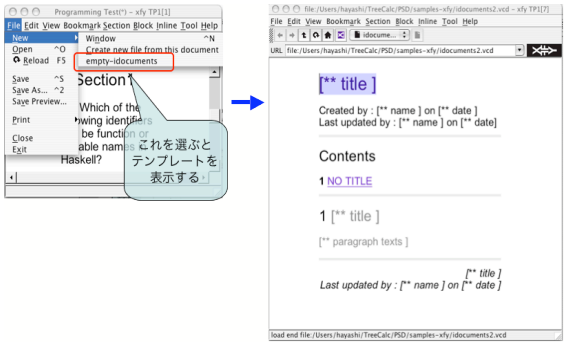

図 2. 新規文書の作成

次にテンプレートにタイトル等を入力する．メニューから構築ブロック関連のコマンドを選ぶと入力を促す言葉と共にカーソルが，その内容が表示されるべき場所に現れるので，内容を入力することにより，文書を作成していく．ここではいくつかの選択肢から正解を選ぶタイプの問題を作成する．問題，選択肢，解答欄を作成した後，PSD Expression コマンドを選ぶと Evaluate ボタンと Hide Code ボタンの 2 つのボタンと共に，埋め込みコードの入力を促す言葉とカーソルが表示される．そこに解答欄に入力された解答を受け取り，採点を行う評価式を Haskell で記述する（図 3）．Hide Code ボタンを押すとコードが見えなくなり，ボタンの表示が Edit Code に変わる．このコード中で用いた関数の定義は，Function Definition 命令で定義される．この命令を選ぶと，Hide Program ボタンと共に関数定義の入力を促す言葉とカーソルが表示されるので関数の定義を入力する．利用者は

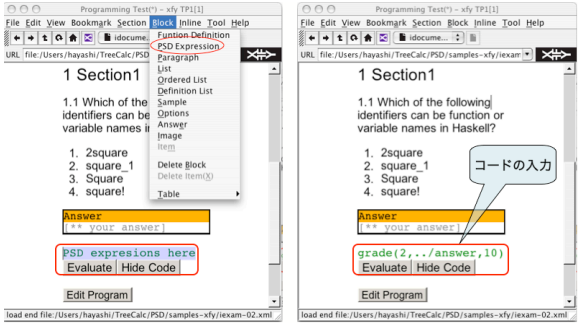

図 3. PSD コードの埋め込み

解答欄に解答を入力して Evaluate ボタンを押すことで採点結果を知ることができる（図 4）．

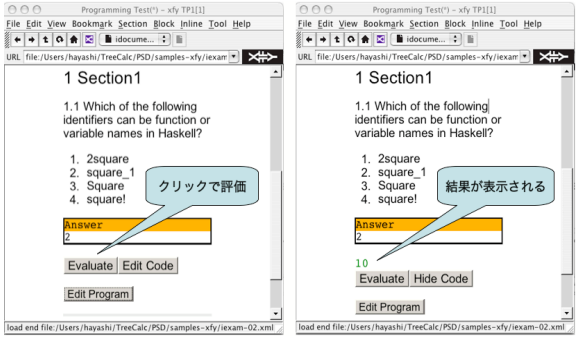

図 4. PSD コードの評価

ここでは簡単化したもので説明したが，解答内容に応じたコメントの表示をしたり，総得点計算をする機能も埋め込みコードによって容易に実装できる．

5.2. iTutorial

ここでは，ユーザの入力に応じて動的に説明文が変化する電子的チュートリアルである

iTutorial について説明する．まず，iTutorial 用のテンプレートを選び，文書タイトル，章のタイトル，箇条書き等の項目を，準備された構築ブロックを追加して内容の入力により，作成していく．例として次のような数列上のアルゴリズムに関する問題を用いる．ある数列が与えられた時，その連続した部分列の中で数の和が最大となるような部分列の和を求める問題を解説している．その解法として 3 つの段階に分け，各段階を独立に考えて問題の解決にあたる方法を使っている．ここで例として学習者にある数列を与えてもらい，それを受け取って各段階の処理を段階的に行うプログラムを埋め込んでおく．各段階とは，1. 全ての部分列を計算する．2. それぞれの部分列の和を計算する．3. 部分和の中で最大のものを選ぶ．の 3 段階である．第 2 段階と第 3 段階のプログラムは一つ前の段階の評価式の評価結果を受け取ってその段階の処理を行う評価式を用いている．

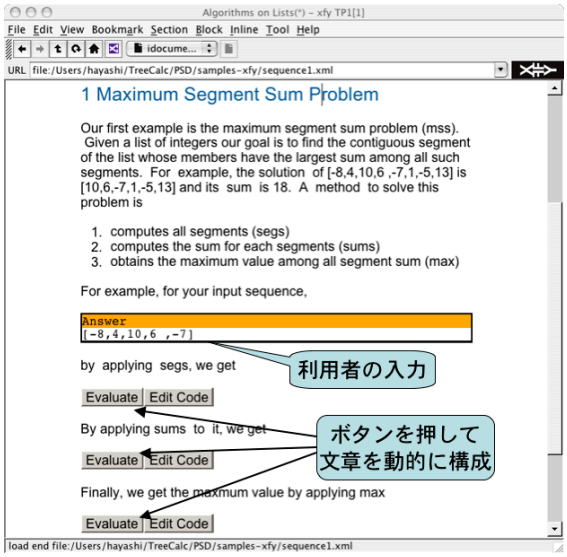

図 5. ボタンを押す

学習者がある数列を与えて各評価ボタンを順番に押していくと(図 5)，各段階での処理の結果が動的に計算されて表示される(図 6).

このように学習者が与えた何らかの情報に応じて，文章自体が変化するような仕組みを埋め込んでおくことにより，対話的に学習できるチュートリアルが作成できる．

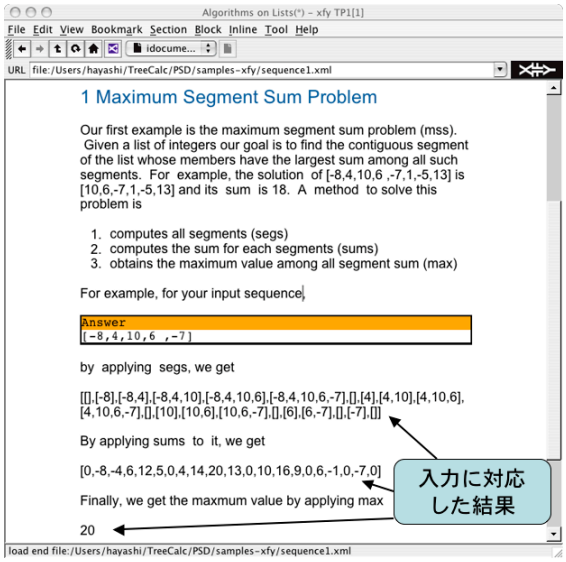

図 6. 評価結果を表示

類似の例としては，各種ソートのアルゴリズムを説明した後に，学習者が与えた数列がソートされていく過程が同じボタンを繰り返し押していくことにより，順番に表示されていくような仕組みをつくることも可能である．この場合は，各評価式が次の中間結果と，その次の結果を評価する評価式を自動生成するようになっているため，各ボタンを押すごとにボタンの背後にある評価式は変わっている．このように数値や文章だけでなく，評価式も結果として生成するようなコードを書くことが可能である．

その他，プログラミングの教科書に記述されたサンプルコードはそのまま，教科書上で編集，実行可能であるので，プログラムの結果が確認できるだけでなく，一部を変更してから実行することや，予め間違いをコードに含ませておいて，学習者に正しいコードに直させるようにすることも可能である．

6. 複合型アプリケーションの管理

xfy は XML のデータを処理するための表示編集環境自体を構築・カスタマイズして様々なアプリケーションを自分で作り出せるようになっている．個々のアプリケーション

に UI や操作を特化しても最終的なデータは XML なので相互の連携･参照が容易である．そこで個々の PSD サブアプリケーションを埋め込みコードによるデータの参照により結びつけて，より高度な複合型の PSD アプリケーションを構築することが可能である．

例えば, iTutorial, iExam はそれ自体 PSD アプリケーションであるがそれらに加えて，問題を集めて編集するための問題データベース, 得点を参照して表やチャート等を自動的に生成して表示させるためのデータ解析アプリケーションを構築してそれらを交互に連携させ, 1 つの複合型教育アプリケーションとして機能させることができる．問題データベースから自動的に問題を選んで iExam を自動生成し，試験結果はデータ解析アプリケーションによって視覚的に分かりやすく表示し，間違った問題に対しては対応する iTutorial の説明箇所を参照できるようにする．逆に iTutorial の説明文の途中でその時点での理解度をチェックするための練習問題を問題データベースから選び, iExam の機能を追加して説明文の間に挿入することも可能である．

このような連携を全て Web ブラウザ上の表示画面の中でコードを記述することにより容易に管理できるようにする．このように xfy と埋め込みコードを併用することにより, 複合的かつ拡張性に優れた対話的教育教材を作成する環境を構築できる．

7. 関連研究との比較

HTML や XML と XSLT を用いた学習教材の作成支援に関しては多数の研究(例えば, [6],[7])や商用ソフトウェアがあるが，そのほとんどは，メタデータから XML を自動生成して XSLT で表示させるものや，HTML のオーサリングツールを用いている．データとしては再利用性や可搬性に優れた XML のデータ形式を用い，文書の編集と動的変化の仕組み，さらにアプリケーション間の連携・参照の記述が実際の Web ブラウザ表示画面上で行うことができるのが本システムの類似の研究にない特長である．また，PSD の埋め込みコードは，新しい評価式を生成して結果に含むようなコードを記述でき，さらにその評価式を評価するという操作が可能である．その点も HTML と共に用いる JavaScript 等の他のスクリプト言語とは異なる特長である．

8. まとめと今後の課題

xfy は汎用的 XML 環境であり幅広い分野への応用が可能であるが, 電子的な情報教育教材の基盤として大きな可能性があると思われる．PSD の考えを実現するための PSD 評価機構を実装しプラグインとして xfy に組み込んだシステム上で, 利用者の入力を受け取り教材の内容を動的に変化させるような対話的教材を作成できる. また文書の構成要素をブロックとして用意しておくことにより, これらの教材作成を支援する環境を構築した. さらには個々の教材や補助ツール自体を構築，個別化し，相互に連携・参照させることでより高度な複合的教材･管理システムとして進化させることができることを説明した．

今後の課題の 1 つは，埋め込みコードの記述を支援する環境を整えることである. 現在, Haskell の有用な関数を予め定義したものからなる関数ライブラリーを作成中である．これにより Haskell プログラミングの知識がなくても，引数の記述とライブラリーからの関数の選択だけで，多くの処理ができるようになる．またシステム構成上は言語独立であるので，いくつかの言語をサポートして利用者が好きな言語を選択できるようにすることを計画している．

もう 1 つの課題は，より高度な処理が行えるように機能を拡充することである．例えばプログラミングの練習問題の解答として利用者が書いたプログラムの正しさを評価したり, 適切な助言を自動生成する機能等が考えられ

る．

また，現状のシステムでは，外部処理系に送られる XML に日本語が含まれると評価が正常に行われない場合があるため，日本語をサポートしていないことにしている．この問題を解決し完全に日本語を使えるようにすることも課題の一つである．

9. おわりに

現在(2005 年 6 月)，開発した環境を Web 上(http://www.psdlab.org/jp)で公開している．公開によって多くの人に利用していただき，その意見・感想をシステムの改良に反映していきたいと考えている．

参考文献

[1] Y. Hayashi, Z. Hu, M. Takeichi, N. Wake, M. Hara, N. Oshima: Pruning DOM Trees for Structured Document Processing. In proceedings of JSSST Annual Conference, 2004.

[2] S. P. Jones. Haskell 98 Language and Librarie. Cambridge University Press, 2003.

[3] M. Takeichi, Z. Hu, K. Kakehi, Y. Hayashi, S.-C. Mu, and K. Nakano. TreeCalc: Towards Programmable Structured Documents. In Proceedings of JSSST Annual Conference, 2003.

[4] N. Matsumoto, M. Tsujii and J. Bennett xfy Technology. In proceedings of Xtech 2005, 2005.

[5] M. Wallace and C. Runciman. Haskell and XML: Generic Combinators or Type-based Translation?. In Proceedings of the 1999 ACM SIGPLAN International Conference on Functional Programming. ACM Press, 1999.

[6] 白田由香利．「XSLT 技術を用いた教材自動生成システムとその使用拡張に関する考察」電子情報通信学会データ工学研究会主催第 15 回データ工学ワークショップ(DEWS2004) 予稿集 CD-ROM, 2004.

[7] 小山幸治，尾崎正弘，森屋裕治，武岡さおり，足達義則．「動的な学習教材作成オーサリングシステムの開発について」名古屋女子大学紀要人文・社会編, No.50, 2004.