SmartDocを中心とした文書フォーマット変換

白石 啓一* 桐山 和彦† 原 元司‡ 山本 喜一§ 本間 啓道¶ 白濱 成希‖
岡田 正**

Document Format Translation Centering on SmartDoc

Keiichi SHIRAISHI Kazuhiko KIRIYAMA Motoshi HARA Kiichi YAMAMOTO
Yoshimichi HONMA Naruki SHIRAHAMA Tadashi OKADA

Synopsis

We discuss about document format translation centering on SmartDoc to extend c-Learning system – the system to collect and process documents. MediaWiki format to SmartDoc format translation and SmartDoc to WebClass importation are discussed. The translator is implemented with programming language AWK. The translator can process headings, bullet lists and numbered lists.

1 はじめに

コンピュータやネットワークを利用した教育であるeラーニングが普及してきており，筆者等も講義に利用している．eラーニングを利用すると，教員自らが解説や問題などのコンテンツを作成することも多い．これらのコンテンツは，個人的にあるいは組織内で再利用されるが，組織を超えて再利用されるものは少ない．我々は，コンテンツの再利用を，組織を超えて行うためのシステムを開発している[1, 2]．本システムをc-Learningシステムと呼んでいる．

コンテンツの再利用を進めるには，コンテンツの文書フォーマットの統一が必要である．筆者等は文書フォーマットだけでなく文書を扱うツールまで仕様が公開されている，文書の一部を再利用しやすくなるよう構造化されている，という条件を満たすSmartDoc[3]を選択した．SmartDocはXMLベースの文書フォーマットであり，トランスレータによりHTMLやLaTeXへフォーマット変換できる．

c-Learningシステムは，SmartDocを中心にした文書フォーマット変換機能，画像フォーマット変換機能，バージョン管理機能，文書共有機能を実現している．しかし，SmartDoc文書は，HTMLやLaTeXと同様にタグにより文書構造が書かれているので，ワードプロセッサのように簡単に文書作成できない．つまり，広く普及させるには難があり，共有機能はあっても，広く使われないことになる．一方，インターネット上にはWikipedia[4]などのフリー百科事典が登場した．フリー百科事典は，信頼性に難があると言われているとは言え，そのコンテンツの利用を検討する価値がある．また，Wikipediaでコンテンツ管理に使われているWikiのような，簡単に使える共同編集システムがあれば，c-Learningシステムの広まると考えられる．つまり，Wikiのコンテンツのc-Learningシステムへの導入，c-Learningシステムのユーザインターフェースの改良という二つの意味でWikiフォーマットからSmartDocフォーマットへのフォーマット変換は重要である．

本稿では，Wiki(MediaWiki[5])フォーマットからSmartDocフォーマットへのフォーマット変換を議論する．既に，PukiWiki用のSmartDoc変換プラグイン[6]は開発されているが，Wikiはシステムによりフォーマットが異なっているので，Wikipediaに使われているMediaWiki用のフォーマット変換を議論することは有用であろう．

c-Learningシステムでコンテンツ再利用ができた

*電子制御工学科
†鳥羽商船高等専門学校
‡松江工業高等専門学校
§OpenEdu プロジェクト
¶奈良工業高等専門学校
‖北九州工業高等専門学校
**津山工業高等専門学校

表 1: SmartDoc のタグ

タグ	意味
<section> </section>	節の開始 節の終了
<subsection> </subsection>	小節の開始 小節の終了
<subsubsection> </subsubsection>	小小節の開始 小小節の終了
<title> </title>	節などの見出しの開始 節などの見出しの終了
   空行	段落の開始 段落の終了 改段落 (, の代わりに使える)
 	箇条書の開始 箇条書の終了
 	番号付き箇条書の開始 番号付き箇条書の終了
 	箇条書の各項目の開始 箇条書の各項目の終了
	箇条書は入れ子にできる.

としても，学習管理システムへのコンテンツ登録が容易でなければ価値が減少する．近年の学習管理システムには，HTML 文書や広く普及しているオフィスソフトウェアの文書を取り込む機能を持つものが多い．筆者等が利用している WebClass[7] にも，外部で編集した HTML 文書や PDF 文書，オフィスソフトウェア文書を取り込む機能がある．本稿では，c-Learning システム (SmartDoc) で生成した HTML 文書を WebClass へ登録する方法を述べる．

2 文書フォーマットと学習管理システム

本節では，本稿で扱う文書フォーマットと学習管理システムの特徴を述べる．

2.1 SmartDoc

SmartDoc は，XML をベースとした文書フォーマットである．トランスレータにより，HTML, LaTeX, プレインテキストなどのフォーマットへ変換できる．文書構造を SmartDoc タグにより指定できる．XML なので，開始タグと終了タグがある (表 1)．節見出し・本文・小節が節に属し，小節見出し・本文が小節に属するなどの階層構造を持つ (図 1)．

2.2 MediaWiki

MediaWiki は，Wikipedia のために書かれた Wiki システムである．Wiki は，WWW ベースのコンテンツ管理システムであり，誰でもそのコンテンツを閲覧・編集できる．文書構造を簡単なマークアップ

```
<section>
<title>1 節の見出し</title>
本文
  <subsection>
  <title>1.1 節の見出し</title>
  本文
  </subsection>
  <subsection>
  <title>1.2 節の見出し</title>
  本文
  </subsection>
</section>
```

図 1: SmartDoc 文書構造 (例)

表 2: MediaWiki のマークアップ

マークアップ	意味
== 見出し ==	レベル 2 見出し
=== 見出し ===	レベル 3 見出し
==== 見出し ====	レベル 4 見出し
空行	改段落
*	箇条書
**	(1 レベルインデントした) 箇条書
***	(2 レベルインデントした) 箇条書
#	番号付き箇条書
##	(1 レベルインデントした) 番号付き箇条書
###	(2 レベルインデントした) 番号付き箇条書
	, #は#*のように混在できる.

言語により指定できる (表 2)．このマークアップ言語は，SmartDoc より簡単である．マークアップは，基本的に行指向であり，各行の先頭にある記号 (マークアップ) により書式が決まる．例えば，==はレベル 2(節) の見出し，===はレベル 3(小節) の見出しを示すマークアップである．SmartDoc と違い，終了を示すマークアップがないので，次に同じレベルの見出しが出て来るまでが一つのまとまりであると言える (図 2)．

2.3 WebClass

WebClass は，(株) ウェブクラスが開発した学習管理システムであり，教材の提示，問題の提示，回答の収集，半自動採点，採点結果の集計などの機能

```
==1 節の見出し==
本文
  ===1.1 節の見出し===
  本文
  ===1.2 節の見出し===
  本文
```

図 2: MediaWiki 文書構造 (例)

```
== 見出し ==
本文
```

図 3: MediaWiki 文書例 1

を持つ．教材については，WebClass 上で編集することも，外部で編集したものを取り込むこともできる．複数ファイルからなる場合，zip や lzh などの形式でアーカイブすることにより取り込むことができる．

3 MediaWiki-SmartDoc 変換

本節では，MediaWiki フォーマットから SmartDoc フォーマットへのフォーマット変換について述べる．以下，MediaWiki フォーマットの文書を MediaWiki 文書，SmartDoc フォーマットの文書を SmartDoc 文書と呼ぶ．

MediaWiki フォーマットは基本的に行指向なので，MediaWiki 文書を 1 行ずつ読み，適切な SmartDoc タグを出力すれば，MediaWiki 文書を SmartDoc 文書へ変換できる．

例えば，図 3 に示した MediaWiki 文書は，図 4 に示した SmartDoc 文書へ変換できる．SmartDoc 文書において，行頭のインデントと改行は読みやすくするためにあり，なかったとしても同じ文書構造を意味する．本稿では，変換後の SmartDoc 文書の行頭のインデントと改行を考慮しないことにする．

この例から分かるように，

1. MediaWiki 文書から 1 行読み，line へ保存する
2. line 行頭に==があれば，以下を実行する
   (a) <section>を出力する
   (b) line から行頭の==と行末の==を除く
   (c) <title> line </title>を出力する
   (d) 4 へ
3. line を出力する
4. MediaWiki 文書の最終行ならば</section>を出力して終了，そうでなければ 1 へ

```
<section>
  <title>見出し</title>
  本文
</section>
```

図 4: SmartDoc 文書例 1

という手順で変換できる．これを手順 A と呼ぶ．

次に，図 5 を考える．図 5 は図 6 へ変換されるべきだが，手順 A では適切な位置で終了タグを出力しないために，図 6 は出力されない．==が現れたことを記録しておき，2 度目の==が現れた時点で，終了タグを開始タグと共に出力する必要がある．つまり，

1. SectionFlag=0
2. MediaWiki 文書から 1 行読み，line へ保存する
3. line 行頭に==があれば，以下を実行する
   (a) SectionFlag=1 ならば，</section>を出力する
   (b) SectionFlag=1
   (c) <section>を出力する
   (d) line から行頭の==と行末の==を除く
   (e) <title> line </title>を出力する
   (f) 5 へ
4. line を出力する
5. MediaWiki 文書の最終行でなければ 2 へ
6. SectionFlag=1 ならば，</section>を出力して終了

という手順で変換する必要がある．これを手順 B と呼ぶ．

次に図 7 を考える．図 7 は図 8 へ変換されるべきなので，手順 B に===の処理を加える必

```
== 見出し 1 ==
== 見出し 2 ==
```

図 5: MediaWiki 文書例 2

```
<section>
  <title>見出し 1</title>
</section>
<section>
  <title>見出し 2</title>
</section>
```

図 6: SmartDoc 文書例 2

要がある．また，<section>, <subsection>, <subsubsection>は入れ子になるので，==が現れたら，<subsection>, <subsubsection>を閉じる必要がある．同様に，===が現れたら，<subsubsection>を閉じる必要がある．つまり，より下位のタグを閉じる必要がある．そのために，手順 B の SectionFlag のようなフラグを複数個使っても良いが，スタックで管理することにする．

1. スタックを空にする
2. MediaWiki 文書から 1 行読み，line へ保存する
3. line 行頭に====があれば，以下を実行する
   (a) スタックのトップが subsubsection ならば，</subsubsection>を出力し，スタックから 1 個取り出す
   (b) subsubsection をスタックに積む
   (c) <subsubsection>を出力する
   (d) line から行頭の====と行末の====を除く
   (e) <title> line </title>を出力する
   (f) 7 へ
4. line 行頭に===があれば，以下を実行する
   (a) スタックのトップが，subsubsection でも subsection でもなくなるまで，以下を実行する
      i. スタックのトップが subsubsection ならば，</subsubsection>を出力し，スタックから 1 個取り出す
      ii. スタックのトップが subsection ならば，</subsection>を出力し，スタックから 1 個取り出す
   (b) スタックに subsection を積む
   (c) <subsection>を出力する
   (d) line から行頭の===と行末の===を除く
   (e) <title> line </title>を出力する
   (f) 7 へ
5. line 行頭に==があれば，以下を実行する
   (a) スタックのトップが，subsubsection でも subsection でも section でもなくなるまで，以下を実行する
      i. スタックのトップが subsubsection ならば，</subsubsection>を出力し，スタックから 1 個取り出す
      ii. スタックのトップが subsection ならば，</subsection>を出力し，スタックから 1 個取り出す
      iii. スタックのトップが section ならば，</section>を出力し，スタックから 1 個取り出す
   (b) スタックに section を積む
   (c) <section>を出力する
   (d) line から行頭の==と行末の==を除く
   (e) <title> line </title>を出力する
   (f) 7 へ
6. line を出力する
7. MediaWiki 文書の最終行でなければ 2 へ
8. スタックが空になるまで以下を実行し，終了する
   (a) スタックのトップが subsubsection ならば，</subsubsection>を出力し，スタックから 1 個取り出す
   (b) スタックのトップが subsection ならば，</subsection>を出力し，スタックから 1 個取り出す
   (c) スタックのトップが section ならば，</section>を出力し，スタックから 1 個取り出す

```
== 見出し 1 ==
=== 見出し 2 ===
```

図 7: MediaWiki 文書例 3

```
<section>
  <title>見出し 1</title>
  <subsection>
    <title>見出し 2</title>
  </subsection>
</section>
```

図 8: SmartDoc 文書例 3

これを手順 C と呼ぶ.

同様に，段落についても，MediaWiki の改行と SmartDoc の p タグを対応付け，適切に変換できる．

次に，図 9 から図 10 への変換を考える．箇条書の処理は*または#が行頭に現れたときに行う．前の行とマークアップ列が異なっていたら，終了タグや開始タグを出力する必要がある．ここでは，箇条書の処理のみ考える．

1. s, t を空文字列にする
2. MediaWiki 文書から 1 行読み，line へ保存する
3. line からマークアップ列を切り出し，s へ代入する
4. i=1, j=1
5. 「s の i 文字目」か「t の j 文字目」が空文字でなければ以下を実行する，両方とも空文字であれば 6 へ
   (a) 「s の i 文字目」と「t の j 文字目」が等しければ，5f へ
   (b) t の j 文字目が*なら
を出力し，#なら
を出力し，空文字なら 5d へ
   (c) j=j+1，5b へ
   (d) s の i 文字目が*なら<ul>を出力し，#なら<ol>を出力し，空文字なら 6 へ
   (e) i=i+1，5d へ
   (f) i=i+1, j=j+1，5 へ

```
* 箇条書 1
* 箇条書 2
** 箇条書 2-a
** 箇条書 2-b
*# 箇条書 2-1
*# 箇条書 2-2
* 箇条書 3
```

図 9: MediaWiki 文書例 4

```
<ul>
<li>箇条書 1</li>
<li>箇条書 2</li>
  <ul>
  <li>箇条書 2-a</li>
  <li>箇条書 2-b</li>
  </ul>
  <ol>
  <li>箇条書 2-1</li>
  <li>箇条書 2-2</li>
  </ol>
<li>箇条書 3</li>
</ul>
```

図 10: SmartDoc 文書例 4

6. line 行頭から s を除く
7. line を出力する
8. t=s
9. MediaWiki 文書の最終行でなければ 2 へ
10. t の文字列長を i とし，i>0 の間，以下を実行し，終了する
    (a) t の i 文字目が*なら
を出力し，#なら
を出力する
    (b) i=i-1
    (c) 10a へ

これを手順 D と呼ぶ.

手順 C と手順 D を組合せた変換処理をプログラミング言語 AWK[8] で実装した．図 11，図 12 に示した変換例の通り，期待通りの動作をしている．

```
= 1 章 =

1 章の本文

== 1.1 節 ==

1.1 節の本文
*箇条書 1
*箇条書 2

=== 1.1.1 節 ===

1.1.1 節の本文
#箇条書 1
#箇条書 2
次の文

=== 1.1.2 節 ===

1.1.2 節の本文

== 1.2 節 ==

1.2 節の本文

*箇条書 1
*箇条書 2
**箇条書 3
**箇条書 4

=== 1.2.1 節 ===

1.2.1 節の本文

*箇条書 1
*#箇条書 3
*#箇条書 4
*箇条書 4

==== 1.2.1.1 節 ====

1.2.1.1 節の本文

#箇条書 1
##箇条書 2
##箇条書 4
#箇条書 3


==== 1.2.1.2 節 ====

1.2.1.2 節の本文

=== 1.2.2 節 ===

1.2.2 節の本文
(以下省略)
```

図 11: MediaWiki 文書例 5(一部)

```
<chapter>
<title>1 章</title>
<p>
1 章の本文
</p>
<section>
<title>1.1 節</title>
<p>
1.1 節の本文
<ul>
<li>箇条書 1</li>
<li>箇条書 2</li>
</ul>
</p>
<subsection>
<title>1.1.1 節</title>
<p>
1.1.1 節の本文
<ol>
<li>箇条書 1</li>
<li>箇条書 2</li>
</ol>
次の文
</p>
</subsection>
<subsection>
<title>1.1.2 節</title>
<p>
1.1.2 節の本文
</p>
</subsection>
</section>
<section>
<title>1.2 節</title>
<p>
1.2 節の本文
</p>
<p>
<ul>
<li>箇条書 1</li>
<li>箇条書 2</li>
<ul>
<li>箇条書 3</li>
<li>箇条書 4</li>
</ul>
</ul>
</p>
<subsection>
<title>1.2.1 節</title>
<p>
1.2.1 節の本文
</p>
<p>
<ul>
<li>箇条書 1</li>
<ol>
(以下省略)
```

図 12: SmartDoc 変換結果 (一部)

4 SmartDoc 文書の WebClass への登録

WebClass は，外部で編集した HTML 文書を取り込む機能を持つ．複数ファイルからなる場合，それらのファイルとともに

(章見出し),,(ファイル名),

または

,(節見出し),(ファイル名),

という形式の行を取り込みたいファイル数分，並べた csv ファイルをアーカイブすることで，取り込むことができる．csv ファイル名は任意であるが，本稿では，List.csv と呼ぶ．

SmartDoc 文書は，-split:section オプションを付けて変換することで，節ごとに別の HTML ファイルに分けられる．そのファイル名は，章の場合，

(元のファイル名)_c(章番号).html

になり，節の場合，

(元のファイル名)_c(章番号)_s(節番号).html

になる．SmartDoc 文書中の章見出し・節見出しは，title タグを探すと見つけられるので，List.csv の生成は自動化できる．

図 13 に示した SmartDoc 文書を HTML 文書へ変換すると，ex.html, ex_c1.html, ex_c1_s1.html, ex_c1_s2.html, ex_c2.html, ex_c2_s1.html, ex_c2_s2.html の 7 個のファイルが得られる．ex.html は目次ページなので，WebClass へ取り込む必要はない．図 14 に示した List.csv とともに lzh アーカイブし，WebClass へ取り込んだ結果，図 15，図 16 に示す通り，取り込むことができた．

5 おわりに

SmartDoc を中心とした文書フォーマット変換について述べた．

MediaWiki フォーマットから SmartDoc フォーマットへの変換方法を示し，AWK で実装した．本実装により，節・段落・箇条書のフォーマット変換ができる．これにより，Wikipedia など MediaWiki を利用しているフリー百科事典の再利用性が高まると考えられる．Wikipedia には，本稿で述べた節・段落・箇条書以外にもマークアップがあるので，それらへの対応が課題である．

```
<?xml version='1.0'
 encoding='iso-2022-jp' ?>
<doc xml:lang="ja">
<head>
<title>WebClass 解説登録実験</title>
</head>
<body>
<chapter id="chap:about">
<title>この実験について</title>
<p>
この実験は，SmartDoc 文書を WebClass の解説へ
登録する実験である．
</p>
<section id="sec:a">
<title>節 1</title>
<p>
節 1 の本文
</p>
</section>
<section id="sec:i">
<title>節 2</title>
<p>
節 2 の本文
</p>
</section>
</chapter>
<chapter id="chap:base">
<title>章 2</title>
<p>
章 2 の本文
</p>
<section id="sec:u">
<title>節 3</title>
<p>
節 3 の本文
</p>
</section>
<section id="sec:e">
<title>節 4</title>
<p>
節 4 の本文
</p>
</section>
</chapter>
</body>
</doc>
```

図 13: SmartDoc 文書例 (ex.sdoc)

```
この実験について,,ex_c1.html,
,節 1,ex_c1_s1.html,
,節 2,ex_c1_s2.html,
章 2,,ex_c2.html,
,節 3,ex_c2_s1.html,
,節 4,ex_c2_s2.html,
```

図 14: List.csv(ex.sdoc 用)

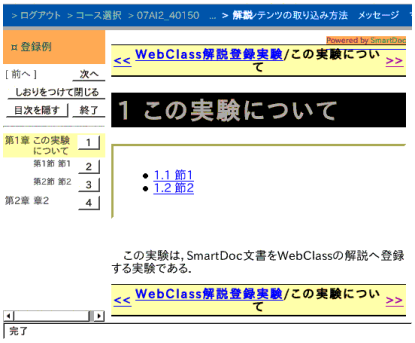

図 15: WebClass 取り込み例 (第 1 章)

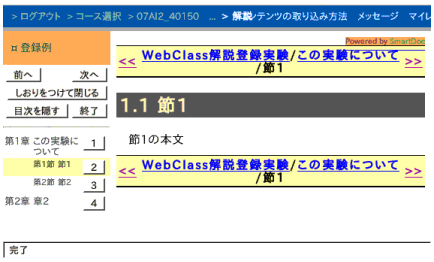

図 16: WebClass 取り込み例 (第 1 章 第 1 節)

また，SmartDoc 文書の WebClass への登録についても述べた．方法を示し，手作業で確認した．作業を自動化するためのプログラム開発が必要である．

MediaWiki-SmartDoc フォーマット変換プログラム，WebClass 登録プログラムの開発は，c-Learning システムをより使いやすく，有用にするためである．c-Learning システムへこれらのプログラムを組み込むことも急務である．

なお，本研究の一部は文部科学省科学研究費基盤研究 (C)(課題番号 19500848) の助成を受けて行われた．

参考文献

1) 桐山 和彦 他, 教育用オープンコンテンツの作成とその管理システムについて, 第 6 回情報科学技術フォーラム一般講演論文集 (4), pp.389–390(2007).

2) 桐山 和彦 他, 完全にコピー自由な教育用コンテンツ配信システムの構築に向けて, 情報処理学会第 70 回全国大会講演論文集 (4), pp.473–474(2008).

3) 浅海 智晴, XML SmartDoc 公式リファレンスマニュアル, ピアソン・エデュケーション (2002).

4) Wikipedia, http://wikipedia.org/ .

5) MediaWiki, http://www.mediawiki.org/ .

6) 田崎 潔志, 深見 空斗, 松永 智揮, 天野 善一, 藤田 毅, 論文作成のためのウェブコラボレーションシステム, 九州産業大学工学部研究報告, **42**, pp.97–102(2005).

7) WebClass, http://www.webclass.co.jp/ .

8) A. V. エイホ, B. W. カーニハン, P. J. ワインバーガー, 足達 高徳 (訳), プログラミング言語 AWK, トッパン (1989).