

BIB_{T_EX} を基にした研究者のための原稿執筆支援システム

BIB_{T_EX}-based manuscript writing support system for researchers

轟 眞市 Shin-ichi Todoroki

物質・材料研究機構,
茨城県つくば市並木 1-1, 305-0044
Todoroki.Shin-ichi@nims.go.jp

小西 智也 Tomoya Konishi

阿南工業高等専門学校,
徳島県阿南市見能林町青木 265, 774-0017
konishi@anan-nct.ac.jp

KEYWORDS BIB_{T_EX}, Ruby, Publication list

ABSTRACT 研究者が書き物仕事をする際、自分の論文リストの中にパソコンに保存してある原稿ファイルへのリンクが埋め込まれていると重宝する。なぜなら、研究発表した記憶を頼りに過去の文書や図面、写真を素早く探し出すことができるし、それらを再利用すれば時間の節約になるからである。この様なリストをウェブ閲覧ソフト上に表示させるシステムを Ruby と BIB_{T_EX} の bib ファイルを使って構築した。このシステムの特徴は、研究者が自分の論文データベースを常に最新のものにしておく気になるように、論文リストを様々な形式で出力したり、現在抱えている原稿を管理する機能を有していることである。

(本論文は、The Asian Journal of T_EX 誌 (4 [x] pp. xx-xx, 2010) に掲載された英語論文の著者訳である。)

1 はじめに

研究論文を執筆する際には、過去に作成して保存した原稿や図面、写真等を参照したくなることが多い。よって、パソコンに保存したファイルを、必要な時にはいつでも検索できるようにしておくことは重要である。必要な素材をすぐに見つけ出すことができれば、その一部を今書いている原稿に再利用するなどして時間を浮かせることができる。このことは年齢を重ねるにつれてより切実になる。ドキュメントは溜まっていくのに、執筆に集中できる時間は減っていくのだから。

その一方で研究者は、自らの研究発表の記録を一覧表にまとめておく必要がある。なぜなら、研究費や求職・昇格を申請するには常に参照する必要があるからである。ということは、このリストをパソコンに保存したファイルを見つけるポータル(過去の原稿の検索ツール)として使うのは道理にかなったことである。研究者のパソコンに保存されたファイルのほとんどは、彼らの研究業績リストに記載の項目のどれかと関係している

はずなので、関係する発表の経緯を思い出せば、目的のファイルに容易にたどり着けるからである [9]。つまり、研究業績リストの各項目に、関係する手元のファイルへのリンクが埋め込まれていれば、研究者は重宝するはずである。

このような発表リストを編纂するには、引用管理ソフトウェア [6] を使うのが良さそうに思える。しかしそれらは、パソコンに保存した原稿ファイルを管理する様に設計されているとは限らない。また、我々は日々新しいファイルを作成しているのであるから、それに関連するリストは最小の手間で、かつできるだけ頻繁に更新できる様になっているべきである。

そこで本報では、BIB_TE_X [4] と Ruby [1] で構築した執筆支援システムを紹介する。その中身は、Web ユーザーインターフェース上で、BIB_TE_X の bib ファイルを操作する Ruby スクリプトである。以下、第 2 節と第 3 節でその設計と実装を説明し、第 4 節で結論を述べる。

2 管理し易い研究業績リストの設計

2.1 方針

引用管理ソフトウェアに追加すべき機能は以下の 2 つである。

- ソフトウェアのデータベースに登録できる項目を増やす。例えば、原稿ファイルへのパス等である。
- 出力するレイアウトをカスタマイズして、手元に保存した素材を開くためのボタンあるいはリンク (以下、リンクと表記) を埋め込めるようにする。

さらにその更新手順に関しては、(1) 日々進めていく仕事の流れを中断しない程度の手間で更新でき、かつ (2) その手間を惜しむ気が起こらない位の見返りがあること、が求められる。

後者の条件については、業績リストを更新する際に、併せて別のバージョンのリスト (例えば研究者個人のホームページに公開しているもの) を一括して自動で更新するのが良い [9]。その他に、筆者のひとりが 6 年間続けてきて効果があると感じている機能は、原稿締切一覧表の自動生成である。その流れを以下に述べる。

1. 原稿執筆に着手する際、引用管理ソフトウェアが管理するデータベースにその項目を登録する。その際、原稿締切日と原稿執筆に用いる作業フォルダへのパスも登録する。必要に応じて、執筆時に参照すると役に立つサイト (例えば、スタイルガイドや学会会場情報の供給元) の URL も登録しておく。
2. 次に、データベースの中から原稿締切日が設定されているエントリを選び出して時間順に並べ替えた一覧ウインドウを表示させる。各項目には、以下のリンクを含ませる (図 1 右参照)。

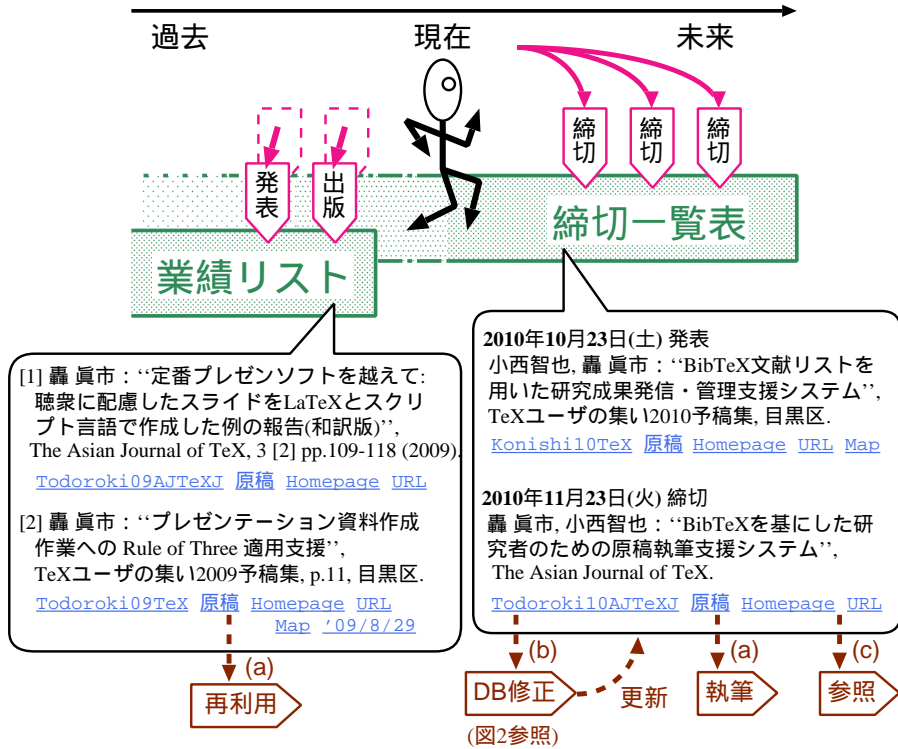


図 1. 本システムにおける原稿縮切一覧表と業績リストとの関係。

- (a) 原稿ファイルが保存されたフォルダにおいてシェル端末 (あるいはフォルダ・ウィンドウ) を起動させるリンク。ここをクリックして執筆作業を開始する。
- (b) データベース内の当該エントリを修正するウィンドウを開くリンク。論文リストを更新する前には、ここをクリックして編集する。
- (c) 登録してある URL に接続するリンク。

つまりこの一覧表には、執筆作業を始めるための入り口と、その文書に関する情報を更新するための入り口を設けておくのである。執筆した原稿が出版されれば、その書誌情報を登録した上で業績リストに移動させる (図 1 左上参照)。2つの入り口は業績リストにも引き継がれ、後に原稿ファイルを再利用する時や、出版後に判明した書誌情報の修正の際に利用する。

これらの機能が使い易く実装されていれば、上述の条件 (1) が満たされる。

2.2 引用管理ソフトウェアの選択

著者らが文献データの収納先として、BIB_TE_X bib ファイルを選んだ理由は、BIB_TE_X を長年使い続けていること他に、以下の点を挙げることができる。

単純な形式: テキストベースの形式は、人間にとって読み易く、編集し易い。さらに、パー

サーとして、BibT_EX 以外にも、様々なものが利用可能であり、Ruby, Python, Perl などの軽量スクリプト言語で書かれたものもある。

拡張性: 予約語の数が限られており、ユーザが定義したフィールドを BibT_EX は無視するので、フィールドを独自に拡張してもコンパチビリティが失われない。

データ可搬性: 事実上、文献書誌情報用の標準形式の一つであるから、適切な変換手段を介して外部のデータベースからデータをインポートできる¹。

3 実装例

次に示す三つの基本方針に基づいて、BibT_EX と Ruby を使って実装した例を、以下詳しく説明していく。

1. BibT_EX の bib ファイルに必要な情報をすべて記録する。
2. bib ファイルに登録されている全てのエントリを Ruby のオブジェクトに収納し、それを操作して必要なエントリを必要な表現で出力する。
3. すべての操作はウェブ閲覧ソフトを通じて行う。ローカルなファイルにアクセスするためのアプリケーションの実行もそこを経由して行う。

3.1 フィールドの追加

原稿管理機能と締切り一覧作成機能を追加するために、BibT_EX の bib ファイルに追加したフィールドを以下に示す (図 2 参照)。

keywords カンマで区切ったキーワードのリスト

deadline 原稿提出締切日

path 原稿執筆に用いる作業フォルダへのパス

homepage 出版に関する情報を提供しているサイトの URL

url オンライン出版 (全文テキスト) の URL

なお、フィールドの命名の際には、biblatex パッケージ [3] が規定しているフィールド名を参考にして、衝突のないように配慮した。

3.2 Ruby による処理

文献データを収録するために、図 3 中央に示す 2 種類のクラスを定義した。Entry クラスは文献エントリを収納するクラスであり、フィールド値をハッシュで保持し、その内容を HTML や L^AT_EX、あるいはプレーンテキストで出力するメソッドが定義されてい

1. これは、文献データを他の標準形式 (例えば XML) で管理することも可能であることを意味する。その選択は、利用者のスキル、経験、嗜好性による。

```

1 @Unpublished{Todoroki10AJTeXJ,
2   keywords = {論文, 情報関連, TeX},
3   author = {轟 真市 and 小西 智也},
4   title = {\BibTeX を基にした研究者のための原稿執筆支援システム},
5   journal = {The Asian Journal of \TeX},
6   year = 2010,
7   volume = 4,
8   number = 2,
9   pages = {xx-xx},
10  deadline = {Oct 23 2010},
11  path = {work/2010/10/AJTeXJ},
12  homepage = {http://ajt.ktug.kr/},
13  url = {http://oku.edu.mie-u.ac.jp/texconf10/proceedings.pdf}
14 }

```

図 2. 拡張したフィールド (斜体) を含む bib ファイルの例。

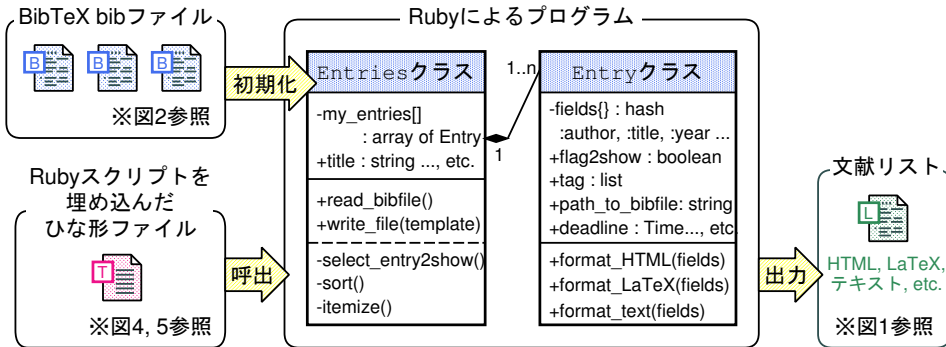


図 3. Ruby による処理。

る。Entries クラスはEntry インスタンスを配列として保持するクラスであり、bib ファイルを読み込んで²初期化され、文献エンリ群を選択・加工・整形して出力するメソッドが定義されている。

これら二つのクラスを用いて、原稿締切り一覧表を HTML 形式で得る例を説明する (図 3 の下部参照)。図 4 に示すひな形ファイルを用意する。その中には、あらかじめ生成しておいた Entries インスタンスを処理して表を取り出す Ruby スクリプトを埋め込んでおく (二重山括弧で囲んだ部分)。つまり、ここでリストの構造を制御するのである。例えば、どのエンリ群を選択するか、選択したエンリ群をどのように並べるか、各アイテムにどのフィールドを含ませるか、それぞれのフィールドをどのような形式で表現するか、等である。

さらに、Entry クラスにメソッドを追加することで、便利なリンクを生成させることができる。例えば、path フィールドで指定されたフォルダにある原稿ファイルを開くリンク (次節で詳しく説明する)、address フィールドで指定された場所を GoogleMaps で開

2. bibtex_parser [5] を利用した。

```

1 <html>
2 <head>原稿締切一覧表</head>
3 <body>
4   <%=
5     《締切日が過去のエントリを削除》
6     《締切日順で並べ替え》
7     《表示するフィールドの指定 (deadline, title, path, ...)》
8     《HTML でリスト出力》
9   %>
10 </body></html>

```

図 4. 原稿締切一覧表を HTML 形式で得るためのひな形ファイルの例。二重山括弧で囲んだ行には Ruby スクリプトが入る。

```

1 \documentclass{jsarticle}
2 \begin{document}
3 ...
4 \section{原著論文}
5   <%=
6     《締切日が未来のエントリを削除》
7     《keywords に「論文」を含むものを選択》
8     《新しい順で並べ替え》
9     《表示するフィールドの指定 (author, title, journal, ...)》
10    《LaTeX でリスト出力》
11  %>
12 \section{学会発表}
13   <%=
14   ...
15     《keywords に「学会」「国内会議」等を含むものを選択》
16   ...

```

図 5. 発表形式別研究業績リストを L^AT_EX 形式で得るためのひな形ファイルの例。二重山括弧で囲んだ行には Ruby スクリプトが入る。

くリンク、year および month フィールドで指定された日付での実験ノートブログ [10]³を開くリンクなどである (図 1 の下中央を参照)。

最後に、このひな形ファイルは、埋め込み Ruby コードを変換する機能を持つ eRuby [8] で処理して、HTML 形式の完成形となる。これをウェブ閲覧ソフトで表示させると、図 1 の右側の様になる。

この仕組みを使えば、ひな形ファイルを用途に応じて用意し、様々な表現 (テーマ別、発表形式別、著者別、等) の文献リストを作成することができる。図 5 は L^AT_EX 形式でテーマ別の発表リストを得るためのひな形ファイルの例である。

3. ユーザ認証機能を持たせたブログサーバで運用する電子ノートブック。

3.3 Web インターフェース

図 1 で説明したように、ウェブ閲覧ソフトからアプリケーションを起動するには、ローカルからの CGI (Common Gateway Interface) 処理実行要求を受け入れて所定のアプリケーションを実行するような HTTP サーバをあらかじめ立ち上げておく必要が有る。図 4 は、汎用 HTTP サーバフレームワーク WEBrick [2] を利用して書いた Ruby スクリプトの例である。発表リスト内にある CGI 実行用のリンクは、すべて Entry クラスに定義されたメソッドの中のどれかから生成されている。言い換えれば、このサーバが論文リスト内に埋め込まれた CGI 実行要求を解釈できる様に、Entry クラスに必要なメソッド群を定義しておくのである。

例えば、シェル端末 (あるいはフォルダウインドウ) を立ち上げる CGI リンクには、オプション引数に `path` フィールドの値を渡す様にする (図 4 の 30 行目と 35 行目参照)。指定した文献のデータベースエントリを編集する画面を開く CGI リンクには、編集すべき bib ファイル名とエントリが存在する行を特定して渡すようにする (図 4 の 41 行目参照)。

なお、図 4 に示した HTTP サーバは第三者にも実行され得ることを認識しておかなければならない。複数のユーザが利用する計算機上では、認証機能や厳格な引数文法チェックを盛り込んでおく必要が有る。

4 おわりに

研究者向けの原稿執筆支援システムとして、研究者が過去の原稿を見つけるのに役立つウェブベースの個人向け発表リストを提案した。これは、原稿執筆に役立つのみならず、研究者の業績リストが常に最新の状態に保たれ、さまざまな形式で出力できると言う点で、研究者の自己アピールにも役に立つ。第 2 節で述べた基本的機能は、BIB_{TEX} 以外の引用管理ソフトウェアにも実装可能なものであるから、このアイデアが T_{EX} コミュニティーの枠を越えて広まることを期待している。なお、本稿で紹介した実装例のサブセットをネット上で公開する予定である [7]。

謝辞

第 2.2 節の記述に関してご教示いただいた査読者のひとりに心より謝意を表します。

参考文献

1. Ruby ホームページ, <http://www.ruby-lang.org/>.
2. Webrick, <http://www.ruby-lang.org/ja/man/html/webrick.html>.
3. Philipp Lehman, *biblatex*, CTAN:macros/latex/exptl/biblatex/.
4. Oren Patashnik and Leslie Lamport, *BibTeX*, CTAN:biblio/bibtex/.

```

1  #!/usr/bin/env ruby
2  require 'webrick'
3  include WEBrick
4
5  s = HTTPServer.new(:BindAddress => '127.0.0.1', :Port => 8000,
6                    :DocumentRoot => File.join(Dir::pwd, "public_html"))
7
8  # /cgi-bin/* へのアクセスは、すべてここに委譲される
9  s.mount_proc('/cgi-bin') { |req, res|
10     res.body =<<"EOS"
11     <html>
12     <body>
13     <script language="JavaScript" type="text/javascript"><!--
14     history.back()
15     // --></script>
16     <noscript>
17     ブラウザの「戻る」ボタンを押してください。
18     <a href="#{req["referer"]}">戻る</a>
19     </noscript>
20     </body></html>
21     EOS
22     res.header["Content-Type"] = "text/html"
23
24     # アクション: 実際に起動するプログラムはここに記述されている
25     key,val = req.unparsed_uri.gsub(/^[^?]*\?cmd=/, "").split(/:/,2)
26     case key
27     when "term"
28         val = File.expand_path("~/") + "/" + val
29         if File.exist?(val) then
30             system("gnome-terminal --working-directory=#{val}")
31         end
32     when "gui"
33         val = File.expand_path("~/") + "/" + val
34         if File.exist?(val) then
35             system("nautilus #{val}")
36         end
37     when "edit"
38         file_name , line_num = val.split(/:/)
39         if File.exist?(file_name) then
40             system("env XMODIFIERS=\"@im=NONE\" " +
41                   "emacs +#{line_num} #{file_name} 2> /dev/null&")
42         end
43     end
44 }
45 Signal.trap(:INT){ s.shutdown }
46 s.start

```

図 6. 所定のアプリケーションを CGI 経由で起動する Ruby でかかれた HTTP サーバのスク립ト。http://localhost:8000/cgi-bin/runcmd.cgi?cmd=key:val と指定すれば、25 行目以降の記述にしたがって起動される。

-
5. Jeff Shantz, *bibtex_parser*, http://rubygems.org/gems/bibtex_parser.
 6. Wikipedia, *Comparison of reference management software*, http://en.wikipedia.org/wiki/Comparison_of_reference_management_software.
 7. 小西 智也, *MyBibList*, <http://www.anan-nct.ac.jp/material/mybiblist/>.
 8. 西山 和広, 標準添付ライブラリ紹介 【第 10 回】 *ERB*, *Rubyist Magazine* 17 (2006), <http://jp.rubyist.net/magazine/?0017-BundledLibraries>.
 9. 轟 眞市, 研究業績リストの電子化 研究者のための執筆・発表支援システム, *セラミックス* 42 (2007), no. 7, 520–524, <http://pubman.mpdl.mpg.de/pubman/item/escidoc:33096>.
 10. 轟 眞市, 小西 智也, 井上 悟, ブログを基にした実験ノート: 個人の研究活動を効率化する情報環境, *Appl. Surface Sci.* 252 (2006), no. 7, 2640–2645 の和訳, <http://pubman.mpdl.mpg.de/pubman/item/escidoc:33128>.