LaTeX, metadata, and publishing workflows

Joppe W. Bos [1] and Kevin S. McCurley [2]

[1] NXP Semiconductors
[2] Self

Abstract. The field of scientific publishing that is served by LaTeX is increasingly dependent on the availability of metadata about publications. We discuss how to use LaTeX classes and BibTeX styles to curate metadata throughout the life cycle of a published article. Our focus is on streamlining and automating much of publishing workflow. We survey the various options and drawbacks of the existing approaches and outline our approach as applied in a new LaTeX style file where we have as main goal to make it easier for authors to specify their metadata only once and use this throughout the entire publishing pipeline. We believe this can help to reduce the cost of publishing, by reducing the amount of human effort required for editing and providing of publication metadata.

Keywords: publishing · LaTeX · metadata

1 Introduction

The original goal of TeX was focused primarily on typesetting, and the appearance of the output on paper. The later invention of LaTeX was focused on "letting the user concentrate more structure of the text rather than the formatting commands" [Lam94]. Users were encouraged to write their papers using high-level macros like `\section`, and leave the decisions like how much space to put before or after a section to the style that is used. As a result, an author does not have to worry so much about how the paper looks, but only has to worry about how the paper is logically structured. This "separation of concerns" is an example of a more general concept from computer science where a program would be separated into distinct units that focused on a restricted part of the task for the program.

The separation of concerns about appearance versus structure has proved to be very effective and most, if not all, scientific publishers now have their own LaTeX styles (overleaf lists about 400 different styles for academic journals[1]). These styles make it easy for an author to conform to a common look and feel in a journal. This can streamline the production steps for a journal if authors comply with the style.

There is at least one area in which the LaTeX community has been slow to adapt to the needs of modern publishing workflows, namely in the *curation of metadata about publications*. This is the main focus of this article. Some of this metadata is fairly obvious, such as the title, list of authors, affiliations, funding sources, and references. Most of this metadata should be supplied by the authors, who best understand the relationships between the different entities. We believe that a LaTeX class should provide a convenient mechanism for authors to enter the metadata only once, in a way that encodes relationships between the entities, so that this LaTeX class can generate appropriate machine-parsable formats which can be used at every phase in the publishing pipeline.

In this paper we survey the various requirements for metadata and the options one can use to collect this information conveniently for the authors. We outline possibilities and

E-mail: joppe.bos@nxp.com (Joppe W. Bos), iacrcc@digicrime.com (Kevin S. McCurley)

[1] See https://www.overleaf.com/latex/templates/tagged/academic-journal.

drawbacks of some of the LaTeX engines and introduce our approach to solve this problem in a new LaTeX class file.

2 The importance of metadata

In the increasingly competitive world of academic publishing, there are several reasons why metadata has become important:

- Metadata provides an important taxonomy for navigating in the world of academic literature.
- Authors, institutions and funding agencies need to receive proper attribution.
- The proliferation of publications has increased the need to rank publications, authors, universities, and departments. The primary methodology for measuring scientific impact and ranking of publications is based on citation analysis, to determine which publications have the greatest influence on (an area of) science. Researchers themselves need to focus their attention on the most important works in a field. Quality metrics can help, even if they are imperfect.
- Papers often end up being posted in multiple places on the web in various forms, so duplicate elimination is a serious problem. Publishers often exacerbate this problem by trying to keep an "officially published" version behind a paywall, while allowing authors to freely post unofficial versions on their home pages and eprint servers.

Academic publications are now being tracked by numerous indexing services including Google Scholar, Clarivate (Web of Science), Scopus, PubMed, and many others. Their task is complicated by the poor state of metadata within publications. The problem of inferring metadata from publications is itself a complicated subject [TXM+20] that could have been simplified if the metata had been provided with the publication itself in a machine-parsable format. This is one of the goals we set out to achieve in this work.

3 Publishing workflow

Part of our motivation arises from our involvement in trying to launch a new open access journal for the professional non-profit society International Association for Cryptologic Research (IACR)[2]. The society already runs two diamond open access journals, but experience from running these has shown that on average each published paper requiries about an hour of human effort for production and metadata handling.

Another study [GB21] estimated the amount of human labor for editing and production to be 7.5 person-hours for each published paper. We believe that most of this should and could be automated, and this can help to lower the cost of publishing. This is particularly important for open access publishing, which is heavily dependent on volunteer labor [BFK+21] as a way to control costs. It can also be used to improve the profitability of commercial publishers.

Publishing workflows vary somewhat from one organization to another, but will at least include the steps of authoring, submission, review, revision, submission of a final version, copy editing, production of readable output formats (e.g., PDF and HTML), DOI registration, indexing, and hosting. Metadata can be important even during the reviewing phase, because it can be used to automatically identify potential conflicts of interest. For large scale operations such as the 2022 NeurIPS conference that received

[2] See https://iacr.org.

9,634 full paper submissions with 10,406 reviewers[3], the metadata can help greatly in automating assignment of reviewers and avoiding conflict of interests.

In some systems like Open Journal Systems [OJS], the submission and curation of metadata is treated as a separate task from submission of the Word, LaTeX or PDF document. That imposes an extra burden on authors, but also renders the workflow vulnerable to inconsistencies with metadata in two places. In our experience, by the time an article has been revised and accepted, there are often changes in titles, abstracts, affiliations, email addresses, etc. Checking and correcting these inconsistencies ends up costing human time by authors and editors.

Metadata becomes increasingly important in the publishing workflow when the author submits their final version. Once the final version is ready, it will typically be registered for a DOI, and this requires supplying a considerable amount of metadata. Moreover, the web "landing page" for a paper typically has to be created from the metadata. Indexing agencies then step in, either by crawling the data or by metadata feeds from the publisher.

4 Our approach at a high level

We automate the capture of metadata during the publishing workflow through the use of a LaTeX class `iacrcc.cls` and a BibTeX style `iacrcc.bst`[4].

- Authors prepare their LaTeX in the usual fashion using the provided `cls` file. Almost all metadata is supplied by authors using structured macros from the class file. The function of the `cls` and `bst` files is to both display the metadata in the output format, but to also extract the metadata during the compilation process, producing an easily parsable external format as a side product.
- When authors supply their final versions, they do so by uploading their LaTeX source to a cloud server, which compiles their sources and extracts all metadata from their sources into a structured format text file. The submission process does not require authors to enter any additional metadata, because it is all encoded into the LaTeX source. The DOI suffix is assigned by the server and the DOI is compiled directly into the PDF at time of submission (the DOI will be registered with the DOI registration agency later).
- A post-compilation step is used to parse the structured metadata and convert it into other formats, including JSON and XML.
- Once the paper has been approved by an editor, the extracted metadata is further used to register the DOI. The extracted metadata is also used to produce various web pages for the journal site, RSS feeds, OAI-PMH feeds, and register with various indexing services.

The metadata output we require is necessarily *text*, and the lingua franca of text encoding is UTF-8. This causes a few problems in the LaTeX world, which is oddly not well suited to text manipulation. We chose an implementation based on use of the `\write` macro during the LaTeX compilation process to write an external file containing metadata.

In our discussion, it is important to distinguish between text *input* encodings and text *output* encodings in the TeX world. All TeX engines are now capable of dealing with UTF-8 *input* encodings, but the support for *output* encodings is much more complicated. The problem here is fonts, which are of course necessary for typesetting. TeX engines are generally concerned with producing typeset material with fonts and hyphenation, and this overloads all discussion about TeX output encoding. LaTeX also has an internal

[3]See `https://media.neurips.cc/Conferences/NeurIPS2022/NeurIPS_2022_Fact_Sheet.pdf`.
[4]The authoritative place to download these is publish.iacr.org/iacrcc.

representation for characters called the "LICR". A complete discussion of this is beyond the scope of the paper, but see [MFL16] and [MGB+04, § 7] for more details.

LaTeX is not really intended for text manipulation, and the TeX input file format predates all modern text encoding methods. Authors are encouraged to write their input in 7-bit ASCII, using things like `\"u` to represent the character ü. This has improved somewhat over the years, and most LaTeX engines can now process input files that are encoded in UTF-8 (but that doesn't mean LaTeX can handle all unicode characters, because it may lack a font to render the character).

Some characters like `# $ % & _ ^ ~ \ { } %` and spaces require special effort by authors to encode these as text, because they have special meaning to the TeX parser. When LaTeX reads an input file, the contents are converted to a sequence of tokens, some of which represent text characters. This internal representation makes it inconvenient to access what would commonly be thought of as "text" in any other computer language. Moreover, the core functionality of LaTeX is user-defined macros, so an author might define `\thatfellow` to represent the text string "Paul Erdős". We only discover this during the LaTeX output process when the macros are expanded into tokens. We rely upon the TeX expansion process, but it is very complex and has many special cases.

The function of the `\write` macro is to expand a list of tokens and write a parsable representation of these tokens into a file. It was originally intended for the writing of auxiliary files like `.bbl`, `.aux`, etc that would be read into the LaTeX source during subsequent compilation. The fact that `\write` performs expansion is very useful to us, because it expands user-defined macros. Unfortunately `\write` also causes a few problems when we use it to produce metadata. The example in Figure 1 illustrates several of these problems:

1. If a space is desired after a user-defined macro like `\Insert` as in Figure 1 then it must be inserted with \␣. Unfortunately the extra backslash is preserved by `\write` in line 18.

2. The braces {} in line 19 come through in the output from `\write`, though they are only meaningful to LaTeX and will be discarded when LaTeX produces typeset output.

3. The ~ character on line 20 gets expanded as an inconvenient token representation, because it has special meaning to TeX. From the point of view of metadata, it should be replaced by a non-breaking space, which is really only relevant to typesetting.

4. The output from line 21 is `With math $\alpha $` with an extra space in it. If we change the $ math delimiters to `\(` and `\)`, then `\write` fails, and part of the errors go to the `output.pdflatex` file. Note that we do not mind having inline mathematics within the metadata, since other metadata formats support either this or MATHML to encode mathematics.

5. Text processing in LaTeX is conflated with typesetting. If we omit line 7 in the example, then it won't compile with `pdflatex` because `\DJ` is unavailable in the default output encoding OT1. If we compile it as is with `pdflatex`, then the PDF output is ok, but then the file `output.pdflatex` has a last line of

   ```
   Đ and \T1\DJ  Āā Ēē Īī Ōō Ūū and �
   ```

   This contains characters with mixed encodings: The Đ is encoded as UTF-8 `C390` but the ü is encoded as the T1(Cork) encoding of `FC` rather than the UTF-8 encoding of `C3BC`. The `\DJ` macro is produced with something that appears to invoke the `T1` encoding.

```
1  % example document that uses \write
2  \documentclass{article}
3  \usepackage{expl3}
4  \usepackage{iftex}
5  \newwrite\afile
6  \ifpdftex
7  \immediate\openout\afile=output.pdflatex
8  \usepackage[T1]{fontenc}
9  \else
10 \immediate\openout\afile=output.lualatex
11 \fi
12 \begin{document}
13 Some unicode: Đ and \DJ\ Āā Ēē Īī Ōō Ūū and \"u and math: \(\alpha^2\).
14 \ExplSyntaxOn
15 \newcommand\justtext[1]{Just text: \text_purify:n{#1}}
16 \ExplSyntaxOff
17 \newcommand\Insert{Insert}
18 \immediate\write\afile{With space:\Insert\ stuff.}
19 \immediate\write\afile{With braces:\Insert{} stuff.}
20 \immediate\write\afile{With tilde: \Insert~stuff.}
21 \immediate\write\afile{With math $\alpha$}
22 \immediate\write\afile{\justtext{na\"ive \textbf{Math}} with $a=b$}
23 \immediate\write\afile{Đ and \DJ\ Āā Ēē Īī Ōō Ūū and \"u}
24 \immediate\closeout\afile
25 \end{document}
```

Figure 1: Example using `\write` to illustrate a few problems.

```
With space:Insert\ stuff.
With braces:Insert{} stuff.
With tilde: Insert\protect \unhbox \voidb@x \protect \penalty \@M \ {}stuff.
With math $\alpha $
Justtext:naïve Math with $a=b$
Đ and \TU\DJ \ Āā Ēē Īī Ōō Ūū and ü
```

Figure 2: Output `output.lualatex` from compiling our example with `lualatex`

Note here that our goal here is not to just “fix this document”. This is merely an example of some of the things we might encounter in an author-supplied LaTeX document when our style file attempts to extract meta-information. Our goal is to provide a system that supports whatever the author supplies to us, and to provide them with clear instructions on how to prepare it without causing any interruptions (errors) or other inconvenience to the author’s typesetting experience. Ultimately the metadata that the author encodes as text, either as TeX macros or as UTF-8 input should survive as UTF-8 output. We could not find a way to do this with `pdflatex`, so our cloud system uses `lualatex`.

We don’t really need the output file to contain only UTF-8 characters, because we can easily convert something like `\"u` to ü with postprocessing by a higher-level language like python that is better suited to text processing. What we *cannot* support is mixed encoding on the output file. The use of `\text_purify:nn` on line 22 generates the line `Justtext:naïve Math with $a=b$`, so it converts the LaTeX accented character and removes the `\textbf` cleanly.

A good example of a LaTeX package with requirements for text output encodings is the `hyperref` package, which needs to produce PDF bookmarks encoded in a text encoding supported by PDF. PDF originally supported only the PDFDocEncoding text encoding. Starting in 1996, PDF supported the UTF-16BE encoding, which is the UTF-16 encoding with a byte order marker at the beginning of the string. The UTF-16 encoding is quite different than UTF-8, since it consists of one or two 16-bit blocks per character, whereas UTF-8 consists of between 1 and 4 bytes to represent characters. Starting with PDF 2.0 in 2017, text may be encoded in PDF as UTF-8, but as of 2022, the `hyperref` package still supports only UTF-16BE (PU) and PDFDocEncoding formats for text. The unicode implementation depends on a 2000-line file with lines like

```
\DeclareTextCompositeCommand{\`}{PU}{u}{\80\371}% U+00F9
```

to define the encoding a 'Latin Small Letter U with Grave' ù. Building such a thing is a tedious process.

Another example of a LaTeX package with requirements for text processing is the `pdfx` package that produces UTF-8 encoded XML inside the PDF output from LaTeX. They defined a pseudo-encoding called `l8u` [RTMS, § 4.2]. They mention that such an encoding could be repackaged into a full package to produce UTF-8 encoded strings, and that might have been useful to us. The `pdfx` package's purpose is similar to ours, in that their objective is to *inject* a subset of metadata into the PDF. Our goal is to *extract* a larger set of metadata from the LaTeX sources (e.g., ORCID IDs).

4.1 Alternative approaches

We considered several ways to implement this metadata extraction, but we settled on an implementation based on the `\write` macro. As the reader can tell, this introduces some complications. One alternative approach would be to use a LaTeX parser to extract the metadata directly from the LaTeX. The problem of parsing LaTeX is complicated by the need to expand macros, for which the LaTeX engines themselves are so far the only robust solution. Another approach that we considered involved using `lua` within `lualatex`. This is much better suited to text processing than using LaTeX itself, but we had an initial goal to try and make things work with any LaTeX engine. Using `lua` code doesn't completely eliminate the problem of macro expansion, and we may return to the `lua` approach again in the future.

5 What metadata is required?

There is potentially a lot of metadata that can be associated with a publication. Some fields are quite obvious, but even the obvious fields have nuance in how they are encoded. Examples include:

- Title of the work. In some fields it is commonplace to use mathematics in titles, but TeX formatting in metadata records is often changed to another format like MATHML. Titles may also encode face markup (e.g., bold face) or multiple character sets. Extremely long titles are sometimes broken up into a hierarchy, incorporating a subtitle or short versions for running titles.

- Authors of the work. Just asking for names of authors can be tricky, and readers are urged to read [McK10]. About the best one can hope for is to provide a string field for a name, but some bibliographic formats may insist upon entering a surname and/or given name.

- Authors may have different levels of contribution. In some cases this is signaled by having author names out of alphabetical order, but in other fields it is common to identify a *role* for author contributions. The CRediT taxonomy is often used to reflect this [NIS]. Authors may also be categorized as a “corresponding author”.

- Relationships between authors and affiliations and/or authors and funding agencies. It is now very common for authors to have multiple affiliations [HRL21] and multiple authors to share a subset of affiliations or funding agencies. These many-to-many relationships are best encoded as relations rather than repeating the information for each author.

- Bibliographic information (e.g., journal or conference name, volume, year, etc).

- A list of references (supplied from BibTeX).

- Licensing information.

There are numerous other fields that may be encoded into a LaTeX document or the output format produced from LaTeX. Examples include abstract, number of pages, address information for authors, email address of authors, links to ancillary works like code and data, etc. It is beyond the scope of this document to catalog all of them, but there have been numerous attempts to define schemas for publishing metadata. We mention them here because the reader may have a requirement to comply with one or more.

Metadata can of course be encoded into various formats such as the LaTeX itself, XML, JSON, or YAML. Most of the descriptions of publication metadata is done in various flavors of XML. We believe that author-supplied metadata should be encoded into the original LaTeX, and that processing of this metadata should be capable of producing other formats that are required for the publication pipeline.

Below we describe a few of the most important metadata schemas.

5.1 Crossref

crossref.org is a non-profit organization whose primary mission is the collection of metadata and the assignment of DOIs. As such, they are one of the most important formats to be compatible with, and they have own XML schema [Croa]. Their schema supports multiple affiliations, author roles, and funding agencies.

5.2 JATS

The Journal Article Tag Suite (JATS) is an XML format that has three variations for archiving & metadata, publishing, and authoring [Usd16]. The JATS format may be viewed as a complete structural representation for a publication; in many ways comparable to LaTeX but focused more on structure and less on layout. A JATS document consists of several sections, including:

front matter This is a required section where most important metadata about the article will appear. It contains metadata about the article but also about the containing serial or journal.

body This optional section is indended to contain the main content of the article itself, and consists of sections, paragraphs, etc.

back matter This optional section contains bibliographic references, a glossary, appendices, or other ancillary information for the article. One area in which JATS excels is in the encapsulation of bibliographic references.

The schema for front matter is very expressive and offers the broadest coverage of publishing metadata that we have found. JATS can accomodate both MATHML3 and inline TEX for mathematics, and as such may also be considered as a competitor to LATEX, or as an output format defined in section 7. It is however quite difficult to convert the full capabilities of LATEX into JATS, and the results are often not faithful to the original intent of the author because JATS contains only semantic structure rather than layout information. In order to convert JATS to a convenient consumable format, it requires a counterpart to LATEX styles, namely XSLT or some other means of formatting the XML.

5.3 Elsevier's Scopus

Scopus is one of the largest indexing agencies. As a commercial service, they partner with publishers to receive their content, but their preferred method is to receive PDF and XML in JATS format [Bre22].

5.4 Clarivate Web of Science

Clarivate's Web of Science is another commercial service that provides indexing services. They accept XML feeds in their own schema [Cla21].

5.5 Other formats

There are a number of less descriptive metadata formats, but they are not as comprehensive. We mention them in part because they may define namespaces that can be used within other XML formats.

DOAJ DOAJ stands for the Directory of Open Access Journals, which is is a community-curated online directory that indexes open access peer-reviewed journals. They accept XML in their own schema or crossref's schema [DOA22].

Extensible Metadata Platform (XMP) This is an ISO standard for representation of metadata associated with digital documents. It was originally created by Adobe Systems and is often used in PDF. It defines several "standard" schemas, but is extensible through inclusion of other XML schemas. The standard schemas lack many basic features that are important to describe academic literature.

Dublin Core The Dublin Core Data set version 1.1 started in 1999 as an XML schema consisting of only fifteen basic elements. It has proved to be inadequate to describe scholarly articles, but some of the elements are still used in other XML schema, and it's one of the standard schemas for XMP. Version 1.1 was improved upon in the "Qualified Dublin Core" in 2008, but this still lacks basic things like ORCID IDs or affiliations. Bibliographic references are encoded as an unstructured string and omitted entirely in Dublin Core 1.1.

PRISM PRISM [Ken] is another simple XML format to describe authors, title, url, DOI, dates, rights, etc. It seems to be less widely used than JATS. The `hyperxmp` package generates XMP in PDF with some tags from this schema.

TEI The Text Encoding Initiative (TEI) is an older XML-based format that started in 1987. It is used mainly in the humanities as an alternative to JATS, but is much less descriptive.

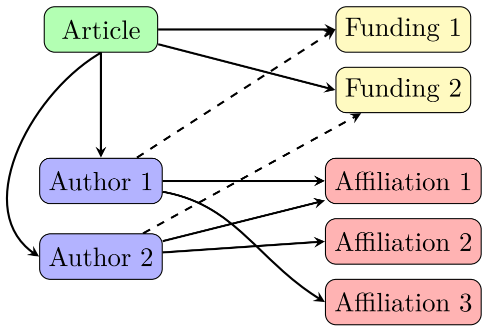

Figure 3: Relationships between major entities. Each entity is listed only once in the article. An article may have multiple authors who share relationships to affiliations. Funding agencies are related to the article in the crossref schema, so we chose to link them this way. As an alternative, relations shown with dashed arrows can link authors to their funding sources, in much the same way that we relate authors to their affiliations. We chose to use footnotes to clarify the complex relationships between funding agencies and authors or affiliations. Some funding agencies (e.g., [oH21]) have strict guidelines for how these annotations should be shown in the paper.

5.6 Conclusions about metadata

When developing a LaTeX class for a journal and/or conference series, it's important to be aware of the metadata elements that are most relevant to the field. Since our journal was in computer science, engineering and mathematics, we had no need for things like clinical trial information or chemical reaction encoding. A complete LaTeX package that supports all possible metadata for a publication would be a daunting project, and would impose an undue burden on authors to wade through all the possible metadata elements.

Our workflow supports registration of DOIs with crossref and indexing by several services. Compliance with these schemas imposes some restrictions on the relationships between articles, authors, affiliations, and funding agencies. These are shown in Figure 3. We believe these are fairly universal to all fields.

6 Using unique identifiers

Unfortunately, things like titles and human names are not unique identifiers. DBLP lists 14 authors in computer science who use the exact name "Thomas Müller", and dozens of others that are similar to this, like Thomas F. Müller. In order to perform large scale bibliometric analysis for attribution or duplicate detection, all entities associated with a publication need to be assigned a unique identifier.

Many of the XML schemas like JATS have embraced the use of unique identifiers. The most notable efforts to assign unique identifiers include:

- DOIs for publications [Pas10],
- ORCID IDs for authors [HFP+12],
- ROR IDs for research institutions [Lam20],
- Crossref funder registry for funding agencies [Crob].

Note that in each case where an organization has assigned a unique ID to an entity, there will often be competing organizations with their own ID space. For example, DOIs cost money, so some organizations have assigned IDs for articles within their own namespace. For example, `arxiv.org` has long been assigning unique IDs to the papers in their namespace. They recently also started assigning DOIs that encode their long-standing ID system.

Sometimes two nearly identical versions of an article will have two DOIs assigned to them in the same namespace (e.g., the publisher version and the arXiv version). This is not the only case when multiple DOIs might be assigned to minor variations of a paper. `arxiv.org` assigns a different DOI to each revision, no matter how small.

ORCID IDs are actually a reserved block of ISNI identifiers, but ISNI has also issued IDs to some authors[5] who don't have ORCID IDs. ISNI also maintains identifiers in other domains, including creators of media, publishers, and some academic institutions[6] as an alternative to ROR IDs. Commercial services that track bibliographic information will typically maintain their own namespace for author and/or affiliation identifiers. For example, other identifiers for authors have been issued by Clarivate Web of Science, Scopus, SciENcv, Mathematical Reviews, and DBLP.

ROR IDs have coarse granularity, so while there is an identifier for Massachusetts Institute of Technology, they don't distinguish between departments, schools, or programs of the university. By contrast, Mathematical Reviews assigns institution codes at the department level (e.g., 1-SCA-C for the department of computer science at University of Southern California).

A complete list of identifiers associated with scholarly publications is beyond the scope of this document, and we should probably expect future ID systems to emerge. Because an entity may have multiple IDs from different organizations, we strongly recommend a schema that assigns IDs with a namespace and identifier within that namespace. Thus for example, a paper on `arxiv.org` might have an identifier within the DOI namespace, but also an identifier within the arxiv.org namespace. The ability to include identifiers from multiple namespaces can help with disambiguation.

6.1 Major takeaways about identifiers

There are a *lot* of potential metadata elements that may be associated with a scholarly article, and the list continues to grow. So long as data is collected in the most granular way possible, it should be possible to reformat it into whatever XML schema is desired. We believe that the JATS format (see Section 5.2) offers the most descriptive language for metadata about publications, and we have designed our metadata macros to align with JATS.

Different participants in the publishing world will have different required elements and optional elements. It is important to collect as much information as possible in order to maximize compliance with downstream consumers of metadata. LaTeX styles tend to be written for specific journals, societies, and conferences, and they need only embrace the metadata elements that are most common to their discipline (e.g., clinical trial data is irrelevant in mathematics). Finally, you should probably expect that the schema may need to be extended in the future.

7 Output formats

An important step in the publishing workflow is to create various formats for human consumption (e.g., PDF and HTML). To the extent possible, it is desirable to embed the metadata into these output formats in a machine-readable way so that the metadata

[5] See for example https://isni.org/isni/0000000088556337.
[6] See for example https://isni.org/isni/0000000121924307.

accompanies the consumable document. Unfortunately the standards for doing so are generally lacking in comprehensiveness.

7.1 Embedding metadata in HTML

HTML has very incomplete specifications for embedding metadata. The original `<meta>` tag was intended for metadata, but this is a flat structure, and the original `name` attribute covers relatively few of the metadata elements for a scholarly publication. There are various extensions that have been proposed for HTML, including the whatwg list[7] of names for `<meta>` tags. Unfortunately this still lacks many of the elements we need.

An alternative would be to encode XML within `<script type="application/xml">`. This would work if your XML does not contain `</script>`. Another option is to use

```
<link itemprop="meta" src="xmlurl">
```

or

```
<link rel="alternate" src="xmlurl">
```

in the `<head>` of the document. Neither of these are standard.

Another schema for bibliographic information in a web page was provided by the ScholarlyArticle format[8] from `schema.org`. This is inadequate for a number of reasons, including the fact that it doesn't contain ORCID IDs. Theoretically it is possible to extend ScholarlyArticle in a way that patches the schema deficiencies, but they have failed to do so and this appears to have very little usage.

Metadata for `schema.org` is typically embedded in a page as `JSON-LD` in the `<head>` of the HTML page as

```
<script type="application/ld+json">
```

One approach that we find promising would be to embed JATS in this way, by converting the XML to JSON. Unfortunately there is no standard way of converting XML to JSON. There is clearly a need for standardization in order for this to be useful.

7.2 Embedding in PDF

While we should expect other consumable document formats to grow in popularity in the future, today's usage of LaTeX is primarily to produce PDF. There are at least two ways to embed metadata into PDF, namely the Document Information Dictionary and XMP. The Document Information Dictionary has existed since version 1.0 of PDF, but encodes only very basic elements like Title, Author, Subject, Keywords, CreationDate, etc. This is the output from the `hyperref` package, but this schema is inadequate for the needs of publishing. In particular it has a single field for author, and does not have any facility to embed unique identifiers like DOI or ORCID IDs. Due to these inadequacies, most metadata should be encoded using XMP, which is described in the next section.

7.2.1 XMP

Starting with PDF version 1.4, the preferred way to embed metadata into PDF is as "Metadata Streams", which are themselves XML documents called packets. This is referred to as the Extensible Metadata Platform (XMP) [XMP]. The XMP standard defines several standard schemas [pdfb], but the only one that appears to be relevant for academic publishing is minimal Dublin Core Schema 1.1 that we already dismissed as inadequate

[7] See `https://wiki.whatwg.org/wiki/MetaExtensions`.
[8] See `https://schema.org/ScholarlyArticle`.

(see Section 5.5). Luckily, as the name implies, this format is extensible, and the XML dictionary may use a schema from a variety of namespaces [pdfc]. The `hyperxmp` package already does this for a few metadata elements.

We believe that there is considerable room for improvement in the use of XMP metadata, including the encoding of bibliographic references, ORCID IDs, affiliations, funding information, etc. Unfortunately the XMP extension mechanism requires that if any extension schema is used, then the entire extension schema must be embedded into the PDF document. This can result in a PDF file that has more schema information than actual metadata.

7.2.2 PDF/A and archiving

PDF/A is a set of ISO standards [pdfa] that define a subset of PDF, intended for long-term archiving of electronic documents. The standards define different versions and different levels of compliance. PDF/A-1 is based on PDF version 1.4; PDF/A-4 is based on PDF version 2.0. PDF/A-3 added a feature for file attachments. XMP for metadata was introduced in PDF/A-1 and is a requirement for PDF/A-2. Unfortunately, generating a PDF file that adheres to one of the PDF/A standards can be difficult for a number of reasons:

1. there are very few open source tools to validate PDF/A compliance, perhaps due in part to the fact that the standards are proprietary ISO documents that are not freely available. Probably the best open source tool is veraPDF, but this is still in beta.

2. Fonts need to be embedded into the PDF. LaTeX engines will usually do this automatically, but authors should beware that if they include PDF graphics, then they may contain fonts that are not embedded.

3. Color spaces must be included in the PDF. This can be problematic if the PDF includes color graphics in different color spaces, because only one color space is allowed in PDF/A-1.

4. Certain PDF features such as transparency, encryption, embedded audio, and javascript are forbidden.

One requirement for XMP metadata is that the information contained there must agree with the data in the Document Information Dictionary. This has been pointed out to be problematic, since the Document Information Dictionary is vague on how to encode multiple author names [Pak, p. 13].

Some funding agencies such as the National Science Foundation, the European Commission, and members of cOAlition S now require research publications that have been funded by them to be deposited in approved repositories for long-term digital archiving. In particular, NSF now requires deposit of PDF/A files for all of their sponsored research [Nat]. Different repositories will have different requirements for deposit, but PDF/A and JATS are commonly supported.

It should be noted that the archival value of PDF/A has been questioned by some [Kli17].

7.2.3 LaTeX packages for XMP and PDF/A

Note that the `hyperref` package includes an option to generate PDF/A, but this only disables some things that would violate the PDF/A specification, and does not assist with XMP metadata. There are several LaTeX packages designed to inject XMP data into PDF files and/or produce PDF/A compliant output, including `hyperxmp`, `xmpincl`, and `pdfx`. These differ in their goals, their approach and the results. `hyperxmp` is incompatible with `pdfx`. Note also that `pdfx` requires the `luatex85` package to work with `lualatex`.

The simplest package for authors is hyperxmp. This modifies the behavior of the hyperref package to generate XMP metadata in the PDF. This is used by acmart to provide XMP, and the hyperxmp package works with acmart to intelligently infer some metadata. Unfortunately, hyperxmp has a few problems:

1. We found that some output fails validation as PDF/A under the verapdf package, even under the relatively lax PDF/A 1-b standard.

2. It is restricted to a predetermined set of schema, including the basic schemas but adding PRISM and others. It omits things like ORCID IDs, affiliations, ROR IDs, bibliography, and other elements that we consider desirable.

The pdfx LaTeX package [RTMS] was designed to provide support for generation of PDF/A compliant documents, and therefore also supports injecting XMP metadata into a PDF. It supports generation of embedded XMP through mappings from LaTeX macros to XML elements, such as \Title to generate <dc:title> and \Volume to generate <prism:volume>. Unlike hyperxmp, the pdfx package requires the author to specify metadata elements in an external file \jobname.xmpdata. By using these macros, it can ensure that the data in the Document Information Dictionary and the XMP data agree.

While hyperxmp and pdfx both support XMP by defining macros to target specific metadata elements, a more flexible approach is provided by the xmpincl package. xmpincl assumes that you will create the raw XML/RDF file, and just includes that. This places a considerable burden on the user, but provides more flexibility than hyperxmp. The pdfx package uses xmpincl to include XMP. There is no longer an active maintainer for xmpincl, but the package itself is fairly simple.

We should mention that the LaTeX project is working on a multi-year project for tagged PDF output that defines a new metadata interface [MF]. That will allow authors to specify a pdfstandard flag to include a color profile for PDF/A compliance. This is being included in LaTeX3 [l3p], but it still doesn't solve the problem of making sure that included graphics will comply with the declared color standard.

8 \author considered harmful

We now turn to the problem of how to embed metadata into the original LaTeX source. The original LaTeX definition of \author provides little help in capturing author metadata, and is also problematic for displaying large numbers of authors. In the standard article class, the author defines \author to include blocks of formatted text, separated by \and. Thus for example, there is no standard way to associate an ORCID ID with an author's name, or to associate affiliations or funding agencies with an author. Left to their own devices, authors might use various embedded macros or footnotes to link authors to their metadata, and this makes it very difficult to extract metadata from the LaTeX.

Part of the problem here is that the \author macro is intricately woven into the *display* of author information on the page. This is an example where the separation of concerns has been neglected, mixing structure with display. Because of this past history with the \author macro, we deliberately chose to break \author and use \addauthor instead. This means authors have to do some work to convert from other standard LaTeX classes to our class, but we judged that to be necessary because of the bad habits that LaTeX has encouraged.

The display of author information on an article can be very complex, particularly when each author may have multiple affiliations, funding agencies, and footnotes, or when there are a huge number of named authors[9]. The decision of how to *display* author information should be a separate decision from how to *capture* the author information. The role of

[9] e.g., https://arxiv.org/abs/2210.03375

the author is to correctly specify the metadata about authors, and the responsibility of the style is to decide how to lay it out on the page according to the style of the journal. Note that a journal may also impose special requirements on how the author metadata is encoded (e.g., whether to accept pseudonyms or ghost authorships, whether authors are grouped by roles, what kind of contact information is required, etc).

We are not the first to have recognized this problem, and some LaTeX styles have improved upon the basic use of `\author`, and have adopted metadata capture as part of their authoring process.

authblk This package provides several ways of displaying author information; either as footnotes or as blocks of text.

ltugboat The style used by TUGBoat redefines the `\author` macro to capture an individual author's name. Sequential calls to other macros like `\ORCID` allow linking of metadata to individual authors.

acmart The `acmart` [Vey] style also redefines the `\author` macro to capture individual authors, and by ordering of macros, allows the capture of affiliations and ORCID IDs for individual authors.

amsart The journal series of the American Mathematical Society [AMS19] uses physical mail address rather than affiliations, but they support multiple addresses per author and the option of specifying a different "current address" from where the research was done. They do not provide a field for ORCID IDs or ROR IDs. They also allow a second class of author called "contributor", where they are given credit for only a portion of the paper.

elsarticle The journal class by Elsevier also captures some metadata in a structured way. They use one `\author` command per author, and they allow multiple footnote marks to be associated with each author. Their construction is nearly ideal, since each author can have multiple affiliations, and affiliations need not be repeated since they each have their own reference number. Moreover, affiliations take different attributes, so they are extensible to specify things like ROR IDs in the future.

Each of these provides some advance in capturing the metadata about an article, but none of them rise to the level of expressiveness contained in something like JATS. Moreover, we are unaware of any that have attempted to provide functionality for a publishing workflow by extracting the metadata from the LaTeX.

9 The `iacrcc` LaTeX and BibTeX styles

Building on what we have learned from previous efforts, we have designed a new document class called `iacrcc`[10] that allows us to capture as much metadata as possible from a document. This may be used with either BibTeX with our own `iacrcc.bst` style, or may be used with the `biblatex` package. These files are designed to be used in a publishing workflow to produce metadata in several different formats. Not only do they produce metadata to go back into PDF, but they also produce a plain text version of metadata that can be easily processed for other purposes like DOI registration. We capture a broad range of metadata, including alternate titles, author names, surnames, ORCID IDs, affiliations with ROR IDs and addresses, and abstract. An example of author metadata for `iacrcc` is given in Figure 9. Some of this metadata can be expressed in the base schemas of XMP, but others require extension schemas from JATS.

[10] May be downloaded from publish.iacr.org/iacrcc

```
\title[running={Emojex documentation},
       onclick={https://example.com/emo},
       subtitle={Emojis in \LaTeX},
      ]{Emojex: use of emojis in \LaTeX}
\addauthor[orcid={0000-0002-0599-0192},
           inst={1,2},
           onclick={https://www.madmagazine.com/}
           email={fester@example.com},
          ]{Fester \surname{Bestertester}}
\addauthor[orcid={0000-0001-7890-5430},
           inst={2},
           footnote={Thanks mom!},
          ]{Kevin S. \surname{McCurley}}
\affiliation[ror=044t1p926,
             city={New York},
             country={United States}]{MAD}
\affiliation[country={United States}]{Self}
\addfunding[crossref=100011047,
            grantid={A-1234},
            country={Canada}
           ]{AGE-WELL}
```

Figure 4: Sample metadata entry in `iacrcc.cls`

9.1 How it works

The workflow for an author consists of the usual multiple rounds of running `lualatex`, `bibtex` or `biber`, followed by two more runs of `lualatex`. The output from this is a PDF file with XMP metadata, but also a file `\jobname.meta` file that contains all metadata in a structured format. The `.meta` file is written with macros using `\write` macros.

The structure of the `\jobname.meta` is similar to YAML. We thought about attempting to write YAML or JSON or XML format, but each output format has its own set of special characters and encoding requirements that are complicated to achieve in LaTeX. It was easier to write python code to parse our output format than it is to write LaTeX code to produce one of the more common formats. This python code is included in the repository for the `iacrcc`[11].

The basic metadata from the paper is written to the `.meta` file using macros from the `iacrcc.cls` file. The citation information is written into the `.meta` file in one of two different ways, depending on whether the author chooses to use BibTeX or `biblatex`. Both methods produce a `.bbl` file that can be supplied in a journal workflow as a substitute for the original BibTeX files, (which may be huge and contain references that are not used). Using either BibTeX or `biblatex`, the processing of the `.bbl` file results in the production of citation records to appear in the `.meta` file through the use of `\write` macros. The `\write` macros are implemented in the `iacrcc.cls` file for `biblatex`, and are implemented in the `iacrcc.bst` file for BibTeX. In both cases, the `.bbl` file ends up containing a structured form of the citations. This allows us to follow the standard practice of publishers to only require authors to submit their `.bbl` file rather than their entire BibTeX file.

[11] See the github repository at https:github.com/IACR/latex.

9.2 The submission pipeline

When the authors submit their paper, they need only submit their LaTeX source file(s), including the BibTeX file they used. The submission form is minimal, since all metadata is included in the LaTeX and BibTeX files themselves. We simply capture an authenticated `paperid` and require the submitting author to supply an email address for the contact author. We derive the DOI from the `paperid`, and inject it into the PDF during compilation along with the acceptance date and received date.

Once the authors upload their LaTeX sources, our server runs `latexmk` within a docker container containing an instance of texlive. We chose to use `lualatex` because we found it impossible to make `pdflatex` produce UTF-8 output files from `\write`. The server validates that the sources were compiled, and provides reports back to the author in case of any errors.

Once the document successfully compiles, then we run a python script to process the `.meta` file, creating metadata in XMP, JATS, JSON, and crossref formats. The JSON format is convenient for immediately publishing the article on the web. The crossref format may be used to register the paper with a DOI.

If the author is satisfied with the metadata and PDF produced from compiling, then the paper moves to the next step of copy editing. In our pipeline this step is fairly limited. We recognize that editors may perform additional tasks such as grammar checking, punctuation checking, or spell correction, but we believe that having to deal with the metadata should not consume the time of a human. While we have attempted to address the issue of metadata handling, we believe that copy editing remains as the biggest obstacle for lowering the cost of open access publishing.

Once the paper is given final approval by the copy editor, the paper may be published without need for a human to handle any of the metadata. At the time the paper is published, the DOI is registered.

We have described the workflow using `lualatex`, but we can also produce HTML output using TeX4ht or LaTeXML or another converter. We plan to investigate this once we have enough experience with the PDF pipeline.

10 Summary

We believe that LaTeX can be used to simplify the processing of metadata in the publishing process, and we have developed a document class that we hope will greatly improve our the quality of our metadata. By using this approach, we believe that it should be possible to streamline the publishing workflow of an open access journal running on a low budget.

Acknowledgements

The authors would like to thank Gaëtan Leurent and other contributors to the `iacrtrans` document class that was used as the starting point of this project. The authors would also like to thank Enrico Gregorio and David Carlisle for answering questions about the inner workings of LaTeX.

References

[AMS19] American Mathematical Society. *AMS Author Handbook Journal Classes*, 2019. URL: https://www.ams.org/arc/journals/packages/jams/jams_amslatex/Author_Handbook_Journals.pdf

[BFK+21] Jeroen Bosman, Jan Erik Frantsvåg, Bianca Kramer, Pierre-Carl Langlais, and Vanessa Proudman. OA diamond journals study. part 1: Findings, March 2021. This report was supported by Science Europe and cOAlition S. doi: 10.5281/zenodo.4558704 URL: https://doi.org/10.5281/zenodo.4558704

[Bre22] Rachel Brennesholtz. *Content Delivery Instructions for Third Party Content Partners*, March 2022. URL: https://elsevier.widen.net/s/hhldnnsq7r/cp-content-delivery-instructions-extrnl

[Cla21] Clarivate Web of Science. *Web of Science Core™ Collection Current Contents Connect®*, feb 2021. URL: https://clarivate.libguides.com/ld.php?content_id=27109663

[Croa] Crossref deposit schema. Accessed 2022-06-30. URL: https://data.crossref.org/reports/help/schema_doc/5.3.1/index.html

[Crob] Crossref funder registry. Described at https://www.crossref.org/services/funder-registry/. Accessed 2022-06-30. URL: https://www.crossref.org/services/funder-registry/

[DOA22] DOAJ. *DOAJ XML*, 2022. URL: https://doaj.org/docs/xml/

[GB21] Alexander Grossmann and Björn Brembs. Current market rates for scholarly publishing services. *F1000Research*, 2021. doi: 10.12688/f1000research.27468.2 URL: https://f1000research.com/articles/10-20/v2

[HFP+12] Laurel L Haak, Martin Fenner, Laura Paglione, Ed Pentz, and Howard Ratner. ORCID: a system to uniquely identify researchers. *Learned publishing*, 25(4):259–264, 2012.

[HRL21] Hanna Hottenrott, Michael E Rose, and Cornelia Lawson. The rise of multiple institutional affiliations in academia. *Journal of the Association for Information Science and Technology*, 72(8):1039–1058, 2021. doi: 10.1002/asi.24472 URL: https://arxiv.org/pdf/1912.05576.pdf

[Ken] Dianne Kennedy. PRISM basic metadata specification. URL: https://www.w3.org/Submission/prism/prism-basic.html

[Kli17] Marco Klindt. Pdf/a considered harmful for digital preservation. *iPRES*, 2017. URL: http://www-archive.cseas.kyoto-u.ac.jp/ipres2017.jp/wp-content/uploads/15.pdf

[l3p] The l3pdfmeta module. URL: https://ctan.math.washington.edu/tex-archive/macros/latex/contrib/pdfmanagement-testphase/l3pdfmeta.pdf

[Lam94] Leslie Lamport. *LATEX : A Document Preparation System, for LATEX 2ε*. Addison-Wesley Publishing Company, 2nd edition edition, 1994. ISBN 0201529831.

[Lam20] Rachael Lammey. Solutions for identification problems: a look at the research organization registry. *Science Editing*, 7(1):65–69, 2020.

[McK10] Patrick McKenzie. Falsehoods programmers believe about names, 2010. URL: https://www.kalzumeus.com/2010/06/17/falsehoods-programmers-believe-about-names/

[MF] Frank Mittelbach and Ulrike Fischer. URL: https://ctan.math.washington.edu/tex-archive/macros/latex-dev/required/latex-lab/documentmetadata-support-code.pdf

[MFL16] Frank Mittelbach, Robin Fairbairns, and Werner Lemberg. Latex font encodings, 2016. URL: https://www.latex-project.org/help/documentation/encguide.pdf

[MGB+04] Frank Mittelbach, Michel Goossens, Johannes Braams, David Carlisle, and Chris Rowley. *The LaTeX Companion*. Addison-Wesley, 2 edition, April 2004. ISBN-10: 0201362996 ISBN-13: 978-0201362992.

[Nat] Public access to results of nsf-funded research. URL: https://www.nsf.gov/news/special_reports/public_access/about_repository.jsp

[NIS] NISO. CRediT: Contributor roles taxonomy. URL: https://credit.niso.org/

[oH21] National Institute of Health. Communicating and acknowledging federal funding, 2021. URL: https://grants.nih.gov/policy/federal-funding.htm

[OJS] Open journal systems. URL: https://pkp.sfu.ca/ojs/

[Pak] Scott Pakin. *hyperxmp – Embed XMP metadata within a LaTeX document*. URL: https://ctan.org/pkg/hyperxmp?lang=en

[Pas10] Norman Paskin. Digital object identifier (DOI®) system. *Encyclopedia of library and information sciences*, 3:1586–1592, 2010.

[pdfa] *TechNote 0008: Predefined XMP Properties in PDF/A-1*. URL: https://www.pdfa.org/wp-content/uploads/2011/08/tn0008_predefined_xmp_properties_in_pdfa-1_2008-03-20.pdf

[pdfb] *Technical Note 0008 – Predefined XMP Properties in PDF/A-1*. URL: https://www.pdfa.org/wp-content/uploads/2011/08/tn0008_predefined_xmp_properties_in_pdfa-1_2008-03-20.pdf

[pdfc] *Technical Note 0009: XMP Extension Schemas in PDF/A-1*. URL: https://www.pdfa.org/resource/technical-note-tn-0009-xmp-extension-schemas-in-pdfa-1/

[RTMS] C. V. Radhakrishnan, Hàn Thế Thành, Ross Moore, and Peter Selinger. *Generation of PDF/X- and PDF/A-compliant PDFs with pdfTeX– pdfx.sty*. URL: https://ctan.org/tex-archive/macros/latex/contrib/pdfx?lang=en

[TXM+20] Dung Thai, Zhiyang Xu, Nicholas Monath, Boris Veytsman, and Andrew McCallum. Using bibtex to automatically generate labeled data for citation field extraction. *arXiv*, 2020. doi: 10.48550/ARXIV.2006.05563 URL: https://arxiv.org/abs/2006.05563

[Usd16] B. Tommie Usdin. So you want to adopt JATS. what decisions do you need to make? In *Journal Article Tag Suite Conference (JATS-Con) Proceedings 2016*, 2016. URL: https://www.ncbi.nlm.nih.gov/books/NBK350380/

[Vey] Boris Veytsman. *acmart – Class for typesetting publications of ACM*. URL: https://ctan.org/pkg/acmart?lang=en

[XMP] *ISO 16684-1:2019: Graphic technology — Extensible metadata platform (XMP) — Part 1: Data model, serialization and core properties*. URL: https://www.iso.org/standard/57421.html