# 06 Towards a Hybrid Workflow Based on Markdown

Creating a workflow that is both structured and flexible enough to cater to different demands is a key step towards to an efficient electronic or hybrid publishing strategy. The hybrid workflow we propose here is based on the need for publishing across different mediums, while keeping the majority of the work process in-house instead of outsourcing.

Instead of developing a digital publication based on the printed book at the end of a production process, as is common practice by publishers, the regular workflow should be adjusted and made efficient and practical towards hybrid publishing at an earlier stage. From-scratch development of each publication format is thus replaced by single-source, multi-format publishing.

However, the description of the workflow which follows below starts at a point which in reality is not the beginning of the publishing trajectory, which starts in reality when an author hands in the definitive manuscript – so after the editing and rewriting process has passed through its final stages. Should the author already be working in Markdown, HTML or even XML format, this will change the workflow. However, in our experience manuscripts are mainly written in Microsoft Word and delivered in .doc or .docx.

Note: An important step preceding the publication trajectory lies in the formulation of the in-house style guide [![Bloglink](images/dpt\_blog\_verwijzing.png)](http://networkcultures.org/digitalpublishing/2014/10/21/style-guide-for-hybrid-publishing/ "Link to blog post: Style Guide for Hybrid Publishing")(Style Guide for Hybrid Publishing), where authors and editors can find the requirements for the manuscript. This style guide must be adapted according to the hybrid workflow before starting any work on the manuscript itself. For example: specification of the required file format, structuring of the text (headers, styles), image specifications, et cetera. See the Guide per genres <!--Miriam: internal link needed, comment copy edit (Amy or Andre)--> for pointers regarding adjusting the style guide in this sense.

We will now turn to implementing the new workflow for small edition and low budget publishing houses. [![Bloglink](images/dpt\_blog\_verwijzing.png)](http://networkcultures.org/digitalpublishing/2014/10/07/hybrid-workflow-how-to-introduction-editing-steps/ "Link to blog post: Hybrid Workflow How-To: Introduction & Editorial Steps")(Hybrid workflow how-to: introduction & editorial steps)

## E-publishing workflows: desktop publishing and Markdown

### Desktop publishing workflow (from Word to InDesign to digital)

Desktop publishing represents the following for a lot of publishers: a Microsoft Word file is imported into InDesign and, after designing and editing, exported to PDF, ready to be printed. After work on the printed edition has been completed, the book is translated into an electronic version, following the design of the 'original' as close as possible. The aforementioned traditional, print-oriented workflow can be seen as a standard for one-to-one publications.

![Traditional Workflow](images/07\_workflow\_traditional.png "Workflow Traditional.")

There are certain advantages to this workflow: it is simple, linear and there are no version branches. You end up with one consolidated manuscript, and What You See Is What You Get when it comes to design. To give an example: in the case of a print design where the page is fixed, changing hyphenation need to be done manually in the InDesign document. In digital publishing hyphenations are not fixed, but will be subject to change as they shift according to the aspect ratio and screen size of the device used. So when an editorial correction involves a hyphenation, this need not be adjusted in the digital file. This saves some work, but also means a limitation in electronic design possibilities.

The main disadvantage of the DTP workflow in going electronic is that you only target one medium and the steps to go from here to a digital edition are quite laborious, and do not make full use of the potential in electronic publishing. It is possible to go from the InDesign file to an EPUB <!-- internal link chapter 7-->, but to make use of the full possibilities hybrid publishing offers, such as modular publishing, immediate updating, optimal uses of audio and visual formats, this should be aimed for from the beginning. A workflow with the paper publication as its basis generally tries to translate print into the digital books, keeping the workflow one-dimensional instead of multi-dimensional.

Importantly, transferring an InDesign document to an electronic publication is not ideal, especially when working with older versions of the software. [![Bloglink](images/dpt\_blog\_verwijzing.png)](http://networkcultures.org/digitalpublishing/2013/05/21/epub-development-in-adobe-indesign-cs6/ "Link to blog post: Notes on EPUB Development in Adobe InDesign CS6")(Notes on EPUB Development in Adobe InDesign CS6) The results can be messy in terms of cleanliness of the code and may require extra steps in finalizing the publication, because unclean code can hamper the rendering of EPUBs generated by InDesign in various e-readers? InDesign, in its current state, is not optimized as a hybrid publishing tool, as it is (too) specifically developed for print design.[^EPUBStraighttothePoint] The latest version of the InDesign suite (InDesign CC) is geared more towards electronic publishing, but requires very careful structuring and preparation of documents in order to yield good results within manageable work times. It's best suited for projects where the content is generated from databases, imported as XML into InDesign and from there exported to EPUB. For a detailed look at these developments, see also paragraph from InDesign to EPUB. <!-- internal link chapter 7-->

It is possible to create 'interactive' publications in PDF, working with Microsoft Word and InDesign. In the end however these are also static InDesign documents, upgraded with some interactive layers. This is still a limited vision of what the possibilities are for digital publishing. The second workflow, which centers on the file format XML, does precisely that.

Another option is ICML import from pandoc to have structured text in InDesign which can be used by the designer. [![Bloglink](images/dpt\_blog\_verwijzing.png)](http://networkcultures.org/digitalpublishing/2014/10/08/markdown-to-indesign-with-pandoc-via-icml/ "Link to blog post: Markdown to InDesign with Pandoc (via ICML)")(Markdown to InDesign with Pandoc (via ICML))

###From Microsoft Word (.docx) to EPUB

Like InDesign, Microsoft Word and any other word processor or **\*\*text editor\*\*** that uses a similar approach (for instance OpenOffice) are not well suited for processing structured text. When working with structured text the author is expected to wrap elements (a heading to emphasize words) in tags, word processors generally apply a visual style to a text, without bothering with tags or any other form of structure. To a certain extent these text processing programs allow working in a structured manner, by using style sheets which determine for instance different types of headers. The problem is that they do not separate between formatting and structure, while in the world of digital publishing this is especially important.

There is a viable solution for generating EPUB from Microsoft Word files, using the command line tool pandoc (for more information visit the [pandoc](http://www.johnmacfarlane.net/pandoc/ 'pandoc website') [^Pandoc] website). The latest versions of pandoc (see below<!---insert cross-reference-->) support document conversion from '.doc' files generated by Microsoft Office 2007 or later, or by comparable programs like OpenOffice/LibreOffice. Since Word does not, as explained above, enforce good structure in a document, the EPUB generated by pandoc will never be perfect and ready for publishing. But it is still a clean enough, usable basis for a designer to produce a the final ebook. Other Word-to-EPUB programs, such as the built-in document converter of calibre [![Bloglink](images/dpt\_blog\_verwijzing.png)](http://networkcultures.org/digitalpublishing/2014/03/28/converting-a-docx-directly-to-epub-using-calibre/ "Link to blog post: Converting a DOCX directly to EPUB using Calibre")(Converting a DOCX directly to EPUB using Calibre), produce worse results.

In order to obtain the best possible EPUB file, the Word document should be formatted, solely if possible, with Word's standard paragraph styles such as 'Normal', 'Title', 'Subtitle', 'Quote' and most importantly 'Heading 1', 'Heading 2', 'Heading 3' for the headings according to their logical hierarchy. For example: 'Heading 1' for chapters, 'Heading 2' for sections, 'Heading 3' for subheadlines. The resulting EPUB document will contain a well-structured table of contents and document navigation menu based on the 'Heading' hierarchy, so well-structured headings are of paramount importance. Word footnotes will be appear as linked endnotes in the EPUB, elegantly simplifying an otherwise tedious document redesign task.

Word unfortunately lacks two features that would make it more suitable for hybrid publishing projects:

1). Word does not have a 'strict mode' that would force all writers and editors of a document to only use defined paragraph styles instead of manually formatting. This means, your document will likely contain headlines that haven't been defined as headlines, but are just bold-faced text, quotes that haven't been defined as quotes etc. Unfortunately, if the manual formatting looks the same as the predefined paragraph styles, it is hard to impossible to spot these parts of a text.

2). Word provides no automatic or semi-automatic tools to find manual formatting and replace it with predefined paragraph styles. The only way to achieve this is to manually control and adjust the whole document.

Often, such inconsistencies in a Word document will only become visible after the EPUB conversion, for example as a missing chapter headline in the table of contents of the electronic book. These are the inherent risks and limitations of using Word in the editorial workflow. Nevertheless, the Word + pandoc solution will likely be the easiest and least painful solution for publishers to adopt.

We recommend two ways of working with Word + pandoc, and discourage a third one:

1. Direct conversion from Word to EPUB using pandoc. This will require that the Word document is 100% consolidated and no further editorial changes will be applied to it. A graphic designer can take the converted EPUB document and quite painlessly transform it into the final electronic publication (among others, by changing the typographic design to make it suitable for e-readers, by scaling and optimizing images for screen reading, by adding bibliographic metadata etc.).

2. Conversion from Word to Markdown using pandoc. Since pandoc can also convert files *\*to\** the Markdown format, this often preferable, especially for complex publishing projects. The resulting Markdown file can then be used as the master file for conversions into all kinds of other file formats (such as HTML for websites). Converting to Markdown yields the advantage that any formatting glitch that existed in the Word document becomes clearly visible. For example, a headline erroneously formatted as bold standard text will show up as '\*\*\\**headline\*\*\\**' while a properly formatted headline will show up as '\#'. This makes it much easier to clean up the internal formatting of the document and have a clean master file for all subsequent document conversions. The EPUB file generated from this Markdown file would in most cases be better structured than the EPUB file directly generated from the Word file, and make the subsequent work of the designer easier. It is also possible to customize pandoc with conversion style templates. This even makes it possible to automatically generate complete and well-formatted EPUB files from the Markdown files without hiring a designer, depending on the type of publication.

3. Not advisable: Using pandoc to go back and forth between Word and EPUB. If the Word document is not consolidated, but subject to further editorial changes, conversion to EPUB (like in the first scenario) would have to be done again, and destroy all work of the designer on the previously exported EPUB file.

### Markdown workflow

Central in a hybrid publishing workflow, we recommend to use the markup language Markdown, as it is an easier language to use than XML. Markdown is not perfect, but is the most easy to work with and enables the creation of structured texts, thus allowing for hybrid publishing. [![Bloglink](images/dpt\_blog\_verwijzing.png)](http://digitalpublishingtoolkit.org/2014/04/mark-me-up-mark-me-down/ "Link to blog post: Mark me up, mark me down")(Mark me up, mark me down)

![Traditional XML](images/07\_workflowMarkdown.png "Workflow XML.")

####Introduction: advantages and limitations

John Gruber, developer of Markdown, describes Markdown on his website as follows: 'Markdown allows you to write using an easy-to-read, easy-to-write plain text format, then convert it to structurally valid XHTML (or HTML).'^[2] Markdown is a way to process plain, unformatted text with human-readable formatting symbols. That means that Markdown doesn't use HTML style tags to format, such as '<b>' for bold or '<author>' to markup the author name. For example, this is what the beginning of *\*Alice's Adventures in Wonderland\** would like in Markdown:

![Markdown](images/Markdown.png "Markdown.")

# Alice's Adventures in Wonderland

## Chapter I. Down the Rabbit-Hole

There was nothing so \_very\_ remarkable in that; nor did Alice think it so VERY much out of the way to hear the Rabbit say to itself,

> Oh dear! Oh dear!

> I shall be late!

(when she thought it over afterwards, it occurred to her that she ought to have wondered at this, but at the time it all seemed quite natural); but when the Rabbit actually \*\*took a watch out of its waistcoat-pocket\*\*, and looked at it, and then hurried on, Alice started to her feet, for it flashed across her mind that she had never before seen a rabbit with either a waistcoat-pocket, or a watch to take out of it, and burning with curiosity, she ran across the field after it, and fortunately was just in time to see it pop down a large rabbit-hole under the hedge.

A short breakdown: '#' signifies a top-level headline, '##' a second-level headline, '\_' italic text, ' **\*\*** ' bold text, '>' a block quote. Beyond that, Markdown provides conventions for marking up bold text, lists, embedded images and links. Its popular extension **\*\*MultiMarkdown\*\*** also supports footnotes, tables, mathematical formulas, cross-references, bibliographies and definition lists. With simple Open Source converter programs, Markdown text like the above can be automatically translated into well-structured HTML, EPUB, PDF, RTF (for importing into InDesign) and other document formats, with a single mouse click or keyboard command, requiring no manual adjustments.

Markdown is a child of Internet culture. It standardizes ad-hoc formatting signs used in e-mail and chats, and became popular in blogging software. There are similar plain text formatting languages to Markdown: the wiki **\*\*syntax\*\*** used in Wikipedia, the language *\*Textile\** that is used for a number of web content management systems and the language *\*reStructuredText\** used for writing technical manuals, and many others.

Markdown and its siblings are human-readable, human-friendly well-structured document formats, excellent for long-term storage and as the basis (or source code) for conversions into present and future document formats. While they are simple, they are also both strict and unambiguous enough in their formatting syntax that multiple writers and editors can work on a single document without introducing strangeness into its formatting. Another advantage of Markdown can be written and edited in any computer program capable of processing basic text. Unlike the file formats of Microsoft Word or other classical word processing programs, the file containing the Markdown flavored text is not bound to specific software, it can be opened using the simplest applications able to parse text.

Why do we recommend Markdown in particular? For particular publishing projects – for example, handbooks or books derived from wikis, it can be worth considering one the alternatives to Markdown such as *\*reStructuredText\**. There are, however, two reasons why we recommend Markdown as a practical tool for electronic and mixed media publishing:

1. Excellent software support. As the most popular human-readable plain text formatting language, there is a plethora of user-friendly, high quality software for writing and editing documents in Markdown and for converting it into other formats available. While Markdown can be written and edited in any computer program that allows to edit text, there are a number of very user-friendly text programs that make it easier to write and view, think of Mou, MacDown, Texts, MarkdownPad, UberWriter or MdCharm.

2. With MultiMarkdown (an extension of Markdown), it provides all the necessary formatting and document syntax needed in arts- and humanities-oriented text publishing. It is perfectly possible to write, for example, a cultural studies Ph.D. thesis in MultiMarkdown, or the essay part (complete with footnotes and bibliographical references) of an exhibition catalog.

Markdown/MultiMarkdown is not a magical one-size-fits-all solution, however. It is well suited for text-heavy editorial work, but limited when creating visual documents and not really usable for interactive publishing formats. It is also not a layout tool, but a pure manuscript format, excellent for keeping manuscripts in a well-structured, readable, durable, software-independent format.

Markdown and similar formatting/markup languages are meant for workflows in which there is a clear separation between editorial work – involving writers, translators and editors – on one hand and publication design on the other. For publications requiring lots of interaction between writers/editors and visual designers/artists from the very beginning of the authoring process, these document languages are not the right tool.

#### Markdown vs. XML

The most detailed structuring language developed is XML and it forms the foundation for many other languages. HTML (or XHTML), for example, is one XML-based document format, Microsoft Words' .docx is another.

XML is meant for structured documents that clearly separate logical structure from visual formatting. That doesn't mean it's actually used to that end, as mentioned before Microsoft Word isn't too strict when it comes to structuring, but .docx is still based on XML. Furthermore, as XML is in itself a toolkit (or actually: a syntax) for defining document formats, it's broad versatility also adds many layers of complexity. While XML theoretically presents the ideal way of working with single format files which deliver multiple output formats, we do not present it as the most advisable solution for small, independent publishing houses.

Markdown provides a good middle-of-the-road solution of a format that is easily usable for non-technicians yet much better structured, and a basis for easy document conversion into HTML, EPUB and many other formats, than Microsoft Word and similar classical word processing programs.

Technically speaking, Markdown provides some of the same features and advantages, namely separation of content structure from visual layout, painless translation into multiple output formats as XML does. The relative simplicity and human readability comes, however, at the price of extensibility and universality. Markdown does not provide the flexibility of building one's own extensions, but provide its own built-in, hard-wired syntax. In XML, there are standard methods of declaring and extending markup languages that can automatically be picked up by XML document converters.

##### Word Processing / editing programs

For Apple's Mac OS X and iOS, there are nice and very user friendly programs for editing in Markdown.

**\*\*Linux\*\***

- [UberWriter](http://uberwriter.wolfvollprecht.de) [^UberWriter], this editor also includes built-in support for pandoc

- [MdCharm](http://www.mdcharm.com) [^MdCharm], supports MultiMarkdown

**\*\*Mac\*\***

*\*Freeware:\**

- [Mou](http://www.mouapp.com) [^Mou], with features like live preview, sync scroll, auto save, powerful actions, auto pair, custom themes and CSS, HTML and PDF export, enhanced CJK support and more.

- [MacDown](http://macdown.uranusjr.com/) [^MacDown], released under the MIT License and influenced in design and setup by Mou.

*\*Paid\**

- [ByWord](http://bywordapp.com) [^ByWord], a user-friendly, distraction-free text writing program with built-in MultiMarkdown support and export to HTML, RTF, PDF and Microsoft Word. The program runs on Mac, iPhone and iPad.

- [iA Writer](http://www.iawriter.com/mac/) [^iAWriter], a program similar to ByWord. The program runs on Mac, iPhone and iPad.

- [Scrivener](http://www.literatureandlatte.com/scrivener.php) [^Scrivener], a word processing program popular among professional writers, for Mac OS X and Windows. Fully supports MultiMarkdown internally

**\*\*Windows\*\***

- [MarkdownPad](http://markdownpad.com/) [^MarkdownPad], free for personal use, with upgrade to MarkdownPad Pro to unlock additional features.

##### Document conversion programs

- [MultiMarkdown](http://fletcherpenney.net/multimarkdown/) [^MultiMarkdown], the original program converts MultiMarkdown files into HTML, PDF, OpenDocument (for later conversion into RTF or Microsoft Word). Open Source, runs on Linux, Mac OS X and Windows.

- [Pandoc](http://johnmacfarlane.net/pandoc/) [^Pandoc], similar in functionality to MultiMarkdown, but much more powerful. Pandoc reads more input formats (including HTML and reStructuredText) and can output HTML5, XHTML, LaTeX, RTF, Word, EPUB 2 and 3, PDF and many more. Typographic templates for the conversion can be easily customized. [![Bloglink](images/dpt\_blog\_verwijzing.png)](http://networkcultures.org/digitalpublishing/2013/08/30/docx-to-markdown-using-calibre-and-pandoc/ "Link to blog post: Docx to MarkDown using Calibre and Pandoc") (.Docx to MarkDown using Calibre and Pandoc)

- [Calibre](http://calibre-ebook.com/) [^Calibre], is an Open Source management and reading program for e-publications. It allows users to manage ebook collections as well as to create, edit, and read ebooks. It supports a variety of formats (including the common Amazon Kindle and EPUB formats), ebook syncing with a variety of ebook readers, and conversion (within DRM restrictions) from different ebook and non-ebook formats. Open Source, runs on Linux, Mac OS X and Windows.

Pandoc is the tool we recommend for working with Markdown, and has also been extensively used in creating this publication.

##### A note on limitations

A major downside of Markdown is that it exists in several variants, each with their own extensions of the basic Markdown syntax. In the context of this Toolkit, we recommend the widespread variant 'MultiMarkdown' that includes syntax for footnotes, tables, citations, cross-references, image captions and document meta data. It is also fully supported by pandoc, the recommended software tool.

Another downside is that Markdown allows some formatting to be marked up in different alternative ways (for example, underlines or asterisks for italic text) which can introduce inconsistency in a collaboratively edited document. The trick mentioned above, to use pandoc for converting from Markdown to Markdown, can be used to eliminate such inconsistencies in a master document.

At the time of this writing in late 2014, a controversial standardization effort of Markdown and its extensions is underway, under the name 'CommonMark'. [commonmark-controversy](http://blog.codinghorror.com/standard-markdown-is-now-common-markdown/) [^commonmark-controversy] We will update this Toolkit as soon as CommonMark has been standardized, released, and is supported by the software we recommend.

## Database publishing

What is gained by a shift to a hybrid publishing workflow? First of all, producing a publication with the possibility to have multiple output formats will be more efficient. However, when the workflow is put into use without restraint and when you really 'change your life', great possibilities will open up. The most important option is shortly discussed here, namely database publishing which uses a so-called content management system.

The ideal database is a collection of independent, but mutually related, objects. These objects can be everything from structured text to pictures of wildlife. The issue at stake is that we need clear grammatical and categorization systems that go beyond the immediate use of the database. Take as an example your mailing list. We can take the full address including the name of the person as one object. But if we want to select per postal code or any other sub category in a full address or want to add more information to the person’s name, such as age, email address and previous purchases, we have to make a strict scheme with so-called fields and sub-fields, including their interdependences (a house number demands a street name and vice versa), their indispensability (age might be not crucial), and so on.

What this means is that the holy grail of the all-encompassing database is not only almost impossible but also an extremely time consuming operation to build and maintain.

The best option is therefore to explicate the goals of the publication program as well as the practical limitations. The four genres we discuss in this publication all have their own demands in relation to the creation and the dissemination of publications. An art catalog demands clear descriptors about the artist, the materials used, the sizes, the provenance, copyright, keywords according to established list, etc. Other collections of pictures might only need a subset.

Also in pure text databases we have to be careful. On one hand we have obviously the data related to the author's name, affiliation, address, etc. But it becomes a different discussion if pictorial illustrations, graphs, etc. are shared between look-alike publications and for that reason we want to identify these objects individually. If such is the case, we might consider a special sector in our database that pertains to illustrations and their specific descriptive sub-field only. The same holds true for bibliographic references and the collection of hyperlinks used in the text.

In conclusion, we can only advise that various objects or in other words pictorial or textual entities are provided with as many consistent metadata (field descriptors) as possible.

[^iAWriter]: iA Writer, http://www.iawriter.com/mac/.

[^EPUBStraighttothePoint]: Elizabeth Castro provides a thorough guide for InDesign-to-EPUB publication in her book: EPUB Straight to the Point, San Francisco: Peachpit Press, 2010.

[^Pandoc]: Pandoc a universal document converter, http://www.johnmacfarlane.net/pandoc/.

[^UberWriter]: UberWriter, http://uberwriter.wolfvollprecht.de/.

[^MdCharm]: MdCharm, http://www.mdcharm.com/.

[^Mou]: Mou, http://25.io/mou/.

[^MacDown]: MacDown, The Open Source Markdown editor for OS X, http://macdown.uranusjr.com/.

[^ByWord]:ByWord 2, http://bywordapp.com/.

[^Scrivener]: Scrivener 2, http://www.literatureandlatte.com/scrivener.php.

[^MarkdownPad]: MarkdownPad, http://markdownpad.com/.

[^MultiMarkdown]: MultiMarkdown, http://fletcherpenney.net/multimarkdown/.

[^Calibre]: Calibre ebook management, http://calibre-ebook.com/.

[^commonmark-controversy]: Jeff Atwood, 'Standard Markdown is now Common Markdown', Coding Horror, 05 Sep 2014, http://blog.codinghorror.com/standard-markdown-is-now-common-markdown/.

[^EpubCheck]: EpubCheck is a tool to validate IDPF EPUB files, version 2.0 and later. https://github.com/IDPF/epubcheck