PARCAT — Applying \TeX in industry

Witkoł Dzubiński, Marcin Woliński and Grzegorz Murzynowski

Abstract
PARCAT is highly sophisticated software for managing a company’s product database and producing a printed catalogue alongside an online shop web site in an automated way. To produce the printed catalogues PARCAT employs \LaTeXX.

In this article we present PARCAT in general and its \TeX{} back-end, describing some of its features in more detail:
• fitting tables to the column width,
• catalogue layouts,
• modularisation of the code,
• system of the layout parameters.

1 Introduction
The PARCAT system is a complex database application whose primary purpose is to comprehensively manage descriptions of products. Descriptions can be given in any language or in several languages. The main distinctive feature of the program is the ability to typeset a catalogue fully automatically. The generated catalogue can be passed to the printing houses in PDF format without any additional treatment. In addition to generating PDF files, PARCAT also prepares files ready for use on web sites. Due to this feature the system can be used as the base tool for managing product descriptions.

Another distinctive and, we dare say, revolutionary feature of PARCAT is its ability to generate several language versions of a catalogue into a single PDF with each language version on a separate layer, which in the printing house can correspond to one black plate, with no need to prepare separate full CMYK plates for each language.

2 History
PARCAT was developed for a large trade company operating for 20 years on the market of electronic components. For over a dozen years Transfer Multisort Elektronik (TME), the company in question, has been publishing the catalogue of its products, in 2011 reaching eight language versions containing about 1800 pages each. The catalogue was made by using CorelDraw software which took approximately five weeks for one language version, excluding the preparation of images. On its web site, the company also presented the same data from the catalogue now in the form of HTML files prepared without any automation. Thus it became obvious that the dynamic development of the company would be limited without tools to manage product descriptions.

PARCAT has been developed continuously since early 2008. After its successful deployment in TME it was decided to continue the development of the software and to try to prepare a universal version which could be useful for other companies as well.

2.1 The structure of a catalogue

In connection with the need to store very large amounts of information, the idea of creating descriptions of individual products was taken from the previous manual composition. It is based on a catalogue divided into parts (e.g., semiconductors, tools, electrical and installation equipment, etc.), which are divided into chapters, and chapters are divided into sections. Frames, assigned to sections, are the smallest (and indivisible) part of the catalogue. A list of products featuring such information as manufacturer, summary description and prices is imported from the sales system. There is no need to enter symbols manually, which could pose a serious problem for such large quantities of products (tens of thousands and more).

2.2 Frames
The concept of a frame is the foundation of the system. It systematises the management of large amounts of products, facilitates the editing of data, making of amendments and also simplifies the data management for a large number of users. A frame is assigned to a section and can describe any number of products (in particular, one product). It is good if the products are a homogeneous group which

This article is a combination of articles from the Bacho\TeX 2010 and the Euro\TeX 2011 proceedings.

PARCAT — Applying \TeX in industry
Frame editor in PARCAT can be described by means of common characteristics. Thus, for example, it is possible to assign to one frame a whole series of resistors with a power of 0.25W in the SMD0608 housing (housing, power, manufacturer, maximum voltage are common features for all these resistors, and resistance — different for each resistor — is the distinguishing feature), or three universal hammers made by one manufacturer which vary in weight and the length of the shaft, but other characteristics (e.g., material, properties, applications) are the same. In addition, a frame features several permanent elements, such as frame title, overall picture, text description.

2.3 Parameters

The description of products is mainly based on user-defined parameters. This solution primarily ensures the consistency of generated descriptions and significantly reduces the costs of translation since the fields associated with a given parameter are translated only once. There are various types of parameters: text, single or multiple choice, several numeric options, as well as those allowing assignment of graphical objects.

2.4 Construction of tables

Parameters in a frame are automatically divided into two groups. The first group contains those parameters whose values are the same for all products in the frame, called common parameters. The parameters which have different values for individual products belong to the second group, called distinguishing parameters.

Clearly the method of data presentation for both groups should be different. The common parameters (Fig. 3) can be associated with their values and listed one under another. They can also be presented as a list with bullets. In general, since the common parameters relate to all products in the frame by definition, a list of respective products does not need to be specified.

The distinguishing parameters (Fig. 4) must be correlated with a list of products, thus showing precisely the products to which they apply.

Witkø Dziubiński, Marcin Woliński and Grzegorz Murzynowski
By default, PARCAT shows a simple text form for the common parameters and a table for the distinguishing parameters. However, the user has a number of tools to change the look of these tables, their order, the order of parameters inside the table, the font size in the table, the position of pictures around the table and many other aspects of the created description. By modifying the construction of tables, the user can adjust the appearance of the frames to their needs and present the data in the most readable way.

2.5 Translation

The list of languages used for product descriptions is not fixed and can be dynamically changed during work on a particular catalogue. All texts present in the system will be automatically marked for translation into a newly added language.

PARCAT features a complex panel for translation management. It indicates the exact number of phrases which require translation into a given language, and each phrase will be translated only once. It allows you to select those sections of a catalogue to be translated. Generated orders are sent to translators for whom a special application is prepared to facilitate their work. It is very important that the translators always see the full context of translation, even if they need to translate only one phrase. This allows for translation quality at the highest level. After the translator finishes work, their order is imported into the database, after verification steps to eliminate mistakes and improve the quality of translation further.

2.6 Graphics

All graphical elements in the program are stored as a pair of PDF files (ready for printing, CMYK with a cut out background) as well as JPG, GIF, etc. (for Internet use). The system features a mechanism allowing users to order the preparation or editing of graphic element. In this way, users without the ability to use graphics programs can still manage graphics. Orders along with the comments are sent to other users with permission to edit graphics (graphic designers). After the image processing is done, the user who sent the order is able to accept changes or request new ones.

2.7 Preview

The possibility of previewing a frame at any stage of its creation (in particular, with the graphics still not accepted!), even without saving changes, is a very important feature of the system (Fig. 5). What is more, the preview in a freely chosen language or languages is created immediately and shows the frame exactly as it will look in print (with accurate pagination) or when connected to a web site, along the lines of WYSIWYG.

A true and fast preview is vital in extensive publications of over 1000-page catalogues. It makes
it possible for all users of the program to assist in the preparation of material, eliminating the complicated, time consuming and discouraging intermediate stage which is usually “the preparation of the preview” in traditional composition systems. Furthermore, the instantaneously generated preview allows users to quickly find both factual and graphical errors.

In addition to the frame preview it is possible to prepare a preview of a section, chapter or part of a catalogue. This allows you to check portions of a catalogue without the need to process all the material. It also enables easy monitoring of the number of pages in the forthcoming catalogue, which is important for ordering at the printing houses. The composition of a single chapter is no longer instantaneous, but it takes only a few minutes to compose tens of generated pages. Therefore these previews can also be generated as necessary.

All these features translate into a significant, if not revolutionary, way to shorten the time users must spend to prepare the catalogue material.

2.8 Typesetting of a catalogue

After preparing and checking all the material you can proceed to a catalogue composition. A generated PDF file features cropmarks and trimboxes, which makes it suitable for submission to the printing houses without additional modifications.

Multi-lingual typesetting with replacement of only the black colour plate is a strategic function of PARCAT. The system provides the ability to compose a catalogue where the data in all languages chosen in the process of composition can be found in one output file. This means that the texts in each of the languages are inserted into the same spaces maintaining the common position of illustrations and the same page breaks. Every language is placed on a separate layer, which allows you to have a preview or trial print for the given language. In addition, each language layer is composed by means of its own additional colour (the spot colour), which facilitates the work of the CTP studio before the preparation of printing plates. This method of printing, with replacement of the black plate, allows us to achieve enormous financial savings. The more language versions of a catalogue and the more catalogue pages, the more savings.

3 Under the hood: \TeX in PARCAT

The typesetting in PARCAT is done using \LaTeX with the \Xe\TeX engine and a highly customised document class. Since we have to cope with text in several languages written in Latin and Cyrillic scripts, we eagerly switched to using Unicode. In this context \Xe\TeX with its UTF-8 input and its ability to use multilingual OpenType fonts provided a comfortable working environment.

\TeX, being a batch processor, plays its role as a typesetting back-end for PARCAT very well. It is fast enough to provide almost instantaneous preview of selected frames. The user can also be practically certain that the result shown for a separate frame will be identical in a complete chapter. We think that in some aspects we have reached the edge of \TeX’s abilities, e.g., with respect to rearranging the language variants. Some of these manipulations would perhaps be easier in Lua\TeX. We will probably investigate this possibility in the future.

Typesetting product catalogues is a rather atypical use of \LaTeX, so we had to solve quite a few \TeXnical problems, the most important being the handling of language variants. This comprises combining several streams of text in one source file; overlaying language variants in such a way that page-breaking and picture positions are the same in all variants, and finally outputting all the variants of the text to the same PDF file using the “optional content groups” feature of the PDF format, with a separate spot colour for each language.

3.1 Tables

One major issue was handling the tables presenting parameters of the products being offered, an important part of the catalogues. Some of these tables are long, spanning up to about 20 pages. They contain headers repeating at the top of each column and should be typeset in a two-column arrangement. Unfortunately, standard \LaTeX packages (\multicols and longtable, supertabular, etc.) do not handle such a combination. We developed a specialised solution in which page breaking is done by the \multicols package. The package was slightly modified to carry the headers for the table in \TeX marks (we used the \TeX ability to create new mark classes).

The column widths are automatically set to fit a table to the column width. A typical table in question consists of a body containing some parameter values, usually decimal numbers, and a header naming those parameters, sometimes with very long contents.

The original class hacks the tabular environment so that it measures its columns and performs a trial setting of a table. There are at most three trials:

- without line breaking,
- with line breaking, at every allowed blank (not at ‘s),

Witkot Dziubiński, Marcin Woliński and Grzegorz Murzynowski
• with line breaking, at any allowed point (including word hyphenations).

In some not-so-extreme cases the result appears as in Fig. 6, left. The table consists of narrow columns and large “column-glue” at the right filling the width to \columnwidth.

Let’s underline that making this work fully automatically is quite an achievement, as anyone who knows something of TEX would agree. But it doesn’t look too good compared with other tables, especially if you know nothing of TEX, does it?

So, the next step taken is to make such a table look as in Fig. 6, right, still fully automatically and without changes to the source of that table (so that no changes are necessary to the front-end software generating it).

The desired effect is achieved by repeating trial settings with an increasing \looseness in a sort of \raggedright scope (turning respective cells into p{(dimen)}), until a minimum value of \looseness is found (or a limit of iterations reached). Then widths of table columns are measured and applied to the final leading.

3.2 New layouts

As the project developed and the PARCAT system is offered to different clients, the need for new page layouts, or rather, graphical concepts, is natural. Samples of the layouts designed so far are shown in Fig. 7.

They are intended not only to present different shapes of graphical elements or placement of headings, but also to illustrate the fact that all those samples are typeset from the same product data and, moreover, the same “intermediate” TEX code.

To be more specific, PARCAT’s front-end software (non-TEX) produces TEX code such as

\begin{wykaz}{@{}lll}
\wyknaglowek{\textbf{PL}{Symbol}{\war{PL}{Symbol}}
\war{EN}{Symbol}{\war{DE}{Symbol}}\war{CZ}{Symbol}\war{HU}{Jelölés}} & 
\najweziej{\war{PL}{Klasa wykonania}{\war{EN}{Manufacture class}{\war{CZ}{Třída provedení}\war{DE}{Ausführungsklasse}{\war{HU}{Érintkező bevonata}}} \} \}

\end{wykaz}

\keysfornextframe{2x3=0:1/0}
\namedinput{frame_24_1/frame_24_1.tex}

\keysfornextframe{2x3=1:2-1}
\namedinput{frame_3_1/frame_3_1.tex}

\end{multicols}

in so-called intermediate files. \namedinput is an input wrapped with stacking the file name so that it can be referred to in messages, which we’ll discuss later (section 3.4). So, as you see, the intermediate file inputs the frame files.

This file in turn is input by the main LATEX document file, alongside files containing settings and configuration data generated by the front-end of the system.

You get all the different outputs (and more) depending on which main file you use — on the same intermediate and frame files!

Our intention is to keep all the templates compatible with one another. For example, notice the unconditional invocation of the multicol environment in the intermediate file (the code sample above) while only one of the examples shown in Fig. 7 is actually two-column.

Turning the multicol environment off was relatively easy. (Relatively, since it’s off only at the main level, where \currentgrouplevel = 0.)

A bit more difficult was to reach a reasonably simple solution for the layout introduced in the IL template (lower left corner of Fig. 7). As you see, the pictures are typeset on the right side of text (tables) and the table(s) break in pages. The complication...
The “classic” STE template

The WZ template

The IL template, mini-toc on left page

The MK template

Figure 7: Samples of the PARCAT templates’ output

Wiktor Dziubiński, Marcin Woliński and Grzegorz Murzynowski
is that the picture is not boxed with adjacent text (which would kill the flexibility of \vskip) but put in a “smashed” box preceding the text, and proper page breaking (i.e., forbidding breaks until subsequent text at least reaches the height of the picture) is ensured by a local change to \pagegoal.

The template presented in the lower right corner of Fig. 7 (Mod), is quite distinct from all the others. It positions a logical frame in a geometric frame consisting of some number of modules sensu graphico, i.e., rectangles of a grid. The grid on the illustration is 2 × 3.

This template is under construction. Our goal is to make it typeset a frame in a proper shape according to free space left on page. So far it typesets frames in a “greedy multicolumn” shape, either “vertical-first” or “horizontal-first” by default or in a shape specified by a key. For instance, the middle frame of the example has (in the optional argument of the \ramka environment) a key 2x3=1:1, which means that on the 2 × 3 grid it should be put in two horizontally adjacent rectangles.

The keys are handled by the pgfkeys package by Till Tantau, which I find much easier to learn than xkeyval, thanks to the path-like structure of the keys.

Although in this article we present illustrations in grayscale, real templates allow colours, of course. Moreover, all the colours are adjustable by the end user. But that belongs to the next story.

3.3 Changing the layout parameters

3.3.1 ParcatColours

The PARCAT templates provide a straightforward mechanism for setting colours. The end user doesn’t have to define the colour of each graphical element separately but is given a set of colour variables that are initialised hierarchically.

This mechanism is realised with the commands \NewParcatColor and \SetParcatColor, whose names are self-explanatory. These commands use the mechanisms of colour definition of the xcolor package, so a very wide range of assignments, reassignments, mixing and shading is available.

In most of the templates the basic colour is nadroz, a legacy name derived from “nadrozdział”, ‘super-chapter’ in Polish. Other colours are tints of it by default. There are also colours (colour names) for the distinctive frame of inserted advertisements, for the backgrounds of pictures and so on.

A nearly parallel mechanism handles other parameters such as dimensions of graphical elements, values of Boolean switches, etc.

3.3.2 ParcatParameters

The main commands are:

- \NewParcatParameter,
- \SetParcatParameter,
- \RenewParcatParameter and
- \OldParcatParameter.

These commands not only declare or set PARCAT parameters but also put them on checklists to issue an error message if a parameter isn’t set at the point of \AtBeginDocument (which we discuss in section 3.4).

The first three names are self-explanatory. The last serves to include standard ($\LaTeX$) parameters in PARCAT’s checklists (with the information about their types). E.g., parcat.cls has a declaration specifying \parindent as a value of type \texttt{dimen}:

\OldParcatParameter\parindent \dimen[0pt]

This allows the standard \parindent to be specified with \SetParcatParameter in a settings\_\_\_\_.tex file. It doesn’t have to be set because the optional argument sets it to \texttt{0pt} by default.

Among the available parameter types are \texttt{dimen}, \texttt{count}, \texttt{skip}, and the corresponding \texttt{dimexpr}, \texttt{\nu}; \texttt{mexpr}, \texttt{\glueexpr} to denote a proper text for the respective $\LaTeX$ primitives, and \texttt{\dim}.

This last item stands for ‘evaluated \texttt{dimexpr}’, where evaluation (\texttt{\the}\texttt{-}\texttt{-}\texttt{-}\texttt{expansion}) of the expression is performed \AtBeginDocument.

As with ParcatColors, \texttt{\edim}s are organised in a sort of inheritance hierarchy.

3.4 PARCAT messages

As mentioned earlier, PARCAT’s $\LaTeX$ back-end generates information, warning and error messages in an XML format, which is intended for parsing by the front-end PARCAT software to provide information to the end user, who most probably is not acquainted with $\LaTeX$.

In particular, such messages are issued if some parameters are not set while they should be, or if an attempt is made to assign a parameter value not appropriate for it. As an example, the parcat.cls class declares

\NewParcatParameter\TitleOnBg \boolean[true]

which means the only allowed values for the parameter are (case-insensitive) true and false. But suppose the settings\_\_\_.tex file contains a typo:

\SetParcatParameter\TitleOnBg{false}

Then $\LaTeX$ outputs an error message:
The only values suitable for \TitleOnBg are \texttt{true} and \texttt{false} while you gave \texttt{false}.

For that purpose we created a mechanism based on these three commands:

- \DeclareParcatModules
- \ProvideParcatModule and
- \LoadParcatModule.

The last two names are self-explanatory, given that a module (\textit{sensu programmatico}) is much like a \LaTeX macro package or a document class options file. But why is the first in plural? It is because from the very beginning of the operation (Operation \textit{Divide et Impera ;-)}) it was clear to us that there would be more than one module (variant) for each logical part of a template.

And indeed, so far there are about 10 modules of page layout, 6 modules of headings, 3 modules of index, 5 modules of frames, 3 modules of tables, &c.

We intend to keep the modules compatible with one another, e.g., any of the frame modules can work with any of the table modules (as far as logically possible). The tests performed so far seem to confirm that such a condition is preserved.

Not the least in preserving that compatibility is the manner of writing the code: all in one source file (using the \texttt{gmdoc} package, with an abundance of commentary) and generating the working files with \texttt{docstrip} directives.

But what about that “industry” in the title? — one may ask.

Our clients are companies with large product bases. To mention just one of them, its printed catalogue for the year 2011 is over 1700 A4 pages in thousands of copies in each of eight languages. That \textit{is} industry and being able to handle it with \TeX makes us proud.

Another and probably not the least aspect of the “industrial strength” of PARCAT’s \TeX back-end is its flexibility for the different needs of different clients, achieved with modularisation and variation of the parameters.

The PARCAT project is developing dynamically and dedicated to treating every client individually so hopefully there’ll be many new features to present in the future.

More information about the PARCAT project can be found at \texttt{www.parcat.eu}.

\begin{itemize}
\item Wiktor Dziubiński \texttt{v.dziubinski at parcat (dot) eu}
\item Marcin Wolński \texttt{wolinski at gust (dot) org (dot) pl}
\item Grzegorz Murzynowski \texttt{g.murzynowski at parcat (dot) eu}
\end{itemize}