Web Service Oriented Generation of LaTeX Plots

Katarína Žáková and Zoltán Janík

***Abstract*—**Preparation of graphics for publication of scientific results can be time-consuming procedure and therefore a lot of authors is looking for the way how do it in simpler and faster manner.

The paper is devoted to the facilitation of work with LaTeX environment. It shows how to automate the preparation of plots for easy use in LaTeX documents. It is realized using JSON-RPC based web services that can be available to all users. The service and the complementary web application enable to take into account user's preferences regarding layout and appearance of resulting pictures and facilitate the picture production.

The implementation was realized using two LaTeX packages: Pgfplots and Gnuplottex based on Gnuplot. Both are widely distributed, their syntax is not so difficult to learn and commands for graphics generation can be written directly in the LaTeX source code.

***Keywords*—** computer-aided design, web service, JSON-RPC, Latex.

I. Introduction

ALL scientists are expected to share their research work with others and in this way to increase the knowledge of all interested humans. Probably the most used form is the conference or journal paper or some other form of technical documentation. Writing high-quality scientific papers takes time and in spite of the fact that this time is well invested, many persons try to reduce the time period devoted to the preparation of a document. Good selection of a word processor can help the author a lot with the paper finalizing. Although these days it is possible to produce nice documents using WYSIWYG word processors such as Microsoft Word or Open Office, LaTeX provides a plethora of professional possibilities that are indispensable for an optimal readability of documents.

It is said that "it's always better to see once rather than to hear twice". Another Chinese proverb says that "a picture's meaning can express ten thousand words" [6]. The meaning of both sayings is more or less the same and it can be related not only to common everyday life but also to technical education and scientific representation of achieved results. Measured experimental data or results from simulations are very often documented by means of plots. They enable to evaluate the quality of achieved outputs much faster than e.g. tables or verbal description. Pictures with charts can be generated by various software applications, e.g. Matlab, Maple, Mathematica, Excel, Maxima, etc. We could list many of them. However, it is usually not so common that such pictures are generated directly in the word processor. LaTeX can be considerd as one of such examples. This paper aims to show how this procedure can be done automatically and more user-friendly. In order to achieve this goal, we have designed a new web service for this purpose.

Different authors give different definitions to Web Services. For example J.P. Morgenthal, chief services architect at Software AG introduced that Web services can be "any application that exposes information processing or serves up information in a reusable data format using common Web technologies" [13].

Leidago Noabeb [9] introduces that "A Web service basically is a collection of open protocols that is used to exchange data between applications. The use of open protocols enables Web services to be platform independent. Software applications that are written in different programming languages and that run on different platforms can use Web services to exchange data over computer networks such as the Internet."

The advance of Web services technologies promises to have far-reaching effects on the Internet and enterprise networks [11]. Web services based on the eXtensible Markup Language (XML), Simple Object Access Protocol (SOAP), and related open standards, and deployed in Service Oriented Architectures (SOA) allow data and applications to interact without human intervention through dynamic and ad hoc connections. Web services technology can be implemented in a wide variety of architectures, can co-exist with other technologies and software design approaches, and can be adopted in an evolutionary manner without requiring major transformations to legacy applications and databases.

II. LaTeX and Graphics

The LaTeX home site (http://latex-project.org/intro.html) states that LaTeX is a document preparation system for high-quality typesetting. It is most commonly used for medium-to-large technical or scientific documents but it can be used for almost any form of publishing. However, it is to say that it is not a word processor, but it is a markup-based typesetting language.

LaTeX contains features mainly for typesetting journal

This work was supported by the grant "Program for support of young researchers" of Slovak University of Technology. It has also been supported by the Slovak Grant Agency, Grant KEGA No. 032STU-4/2013 and APVV-0343-12.

K. Žáková and Z. Janík are with the Faculty of Electrical Engineering and Information Technology, Slovak University of Technology, Bratislava, Slovakia (e-mail: katarina.zakova @ stuba.sk, zoltan.janik @ stuba.sk).

articles, technical reports, books, and slide presentations. It is able to take control over large documents containing sectioning, cross-references, tables and figures. It enables to typeset complex mathematical formulas, generate bibliographies and indexes, etc.

What makes LaTeX better in comparison with WYSIWYG (What You See Is What You Get) word processors is:

- one can focus on content over form,
- it is easier to assemble large documents from multiple files,
- it is possible to use text-based version control (CVS/SVN/etc.),
- it is much more stable even on slower low-end machines,
- output looks the same no matter which platform is used for the document compilation. It is possible to achieve high typographical quality of the documents.

LaTeX can be a good choice for everybody who writes the complex scientific document. The time invested to its learning will be certainly returned back in spite of the fact that writing the first LaTeX document will take much longer than the first Word document. Many people spend most of their time working with documents of complexity to the left of the cross over. Here the LaTeX can bring a big help.

One of the most time-consuming parts of writing publishable research material is making figures. Although it may seem rather simple to produce figures in e.g. Matlab, spreadsheet program or some other environment, the systematic generation of high-quality figures is not so easy. There is a lot of manual work involved in spite of the fact that we aim to draw only plots of changing variables. The figures need to be created at the right size, text fonts, with various width of lines, etc. However, the monotonous work that has to be done with each picture can be automatized.

LaTeX can be also used to produce graphical objects. Of course, there can arise a question if is it really necessary to graph the function within LaTeX. For somebody could be easier to generate figures in Matlab, R, Excel or something else and to use the command \includegraphics in the LaTex source code to include graphs in the text. This procedure can even speed up compilation. So, why to complicate the way of document preparation? There are several good reasons for doing this within LaTeX:

- this approach brings greater ease if one makes edits,
- he author has possibility to conditionally change parameters in the plot,
- the text in the graphics is handled by the same typesetting engine that is handling the text in the documents which makes its look more cohesive and professional,
- it usually brings less files to handle.

LaTeX can be used together with PostScript (on which PDF is based). LaTeX's mathematical capability, its paragraph building, its hyphenation, and its programmable extensibility can cooperate with the graphical flexibility and font-handling capabilities of PostScript and PDF [2].

PostScript is object-oriented computer language enabling to describe the graphical appearance of a printed page. It was developed by Adobe Systems Incorporated. Its main advantage is that it is independent on device on which the document is printed. PostScript can be considered as a standard for desktop publishing. It is supported by the most of high-resolution printers to produce camera-ready copy. There exist several packages for generation LaTeX graphics that are based on PostScript:

- PSTricks, where the necessary PostScript code can be generated by TeX macros defined in the package. It is often used as a base for other LaTeX libraries.
- Pst-plot utilizes PSTricks macros for creation of vector graphics.
- Bardiag, Bchart – both of these packages are already deprecated. They can be used only for producing bar graphs.

Another possibility to generate LaTeX Graphics is offered by command line based Gnuplot program that is a portable command-line driven graphing utility for various platforms.

The set of programs PGF/TikZ also enables the creation of high quality drawings for LaTex. PGF is the basic layer, providing a set of basic commands for producing graphics, and TikZ is the frontend layer with a special syntax, making the use of PGF easier.

PGFPlots is a package that is based on the combination of programs PGF/TikZ. It was directly developed to generate graphs for LaTeX.

Package Matlab2Tikz converts graphs from Matlab to LaTeX using PGFPlots library.

Metapost programming language enables to create vector graphics. It generates figures for technical documents where some aspects of a picture may be controlled by mathematical or geometrical constraints that are best expressed symbolically [3].

Similarly Asymptote is also language for generating vector graphics. However, it requires an external program for compilation of a resulting output document.

Finally, R language can be mentioned. It is a free software environment for statistical computing and graphics.

A brief comparison of all mentioned projects is listed in Table 1. Since we usually use MikTeX distribution of LaTeX, we were interested in the fact, whether the considered package is included in MikTex repository or not. Subsequently, it is interesting to know whether the package includes functions for graphics generation directly. Since the final documentation is very often prepared in PDF format, it is important to know how demanding the preparation of PDF document is - whether it can be compiled directly from the LaTeX source code or whether the source code needs to be compiled to a postscript file first and only then it can be converted from the postscript file to the PDF format.

Title	MikTex repository	Integrated graph functions	pdf export	PostScript based	URL address
PSTricks	✔			✔	http://www.ctan.org/pkg/pstricks-base
Pst-plot	✔	✔		✔	http://www.ctan.org/pkg/pst-plot
Bardiag	✔			✔	http://www.ctan.org/pkg/bardiag
Bchart	✔			✔	http://www.ctan.org/pkg/bchart
Barkom	✔			✔	http://www.ctan.org/pkg/barkom
Gnuplot	package only	✔	✔		http://www.gnuplot.info/download.html http://www.ctan.org/pkg/gnuplottex
PGF/TikZ	✔		✔		http://sourceforge.net/projects/pgf/
TikZ	✔		✔		http://sourceforge.net/projects/pgf/
PGF Plots	✔	✔	✔		http://pgfplots.sourceforge.net/
Matlab2Tikz	✔	✔	✔		http://www.mathworks.com/matlabcentral/fileexchange/22022-matlab2tikz
Metapost	✔	✔			
Asymptote		✔			http://asymptote.sourceforge.net/
R		✔	✔		http://cran.r-project.org/

Tab. 1 Packages for LaTeX graphics generation

Considering all properties of mentioned packages, we decided to focus on two solutions: Pgfplots and Gnuplottex based on Gnuplot. Both of them are wide-spread, their syntax is easy to learn and commands for graphics generation can be written directly in the LaTeX source code. Pgfplots enables to produce PDF format of document without a need to use external programs (using pdflatex). Gnuplottex needs to prepare Gnuplot code that can be compiled to PDF format directly. Both of these packages were implemented into the presented solution.

III. Remote Procedure Call

Remote Procedure Call (RPC) is a protocol for requesting a service from a program located in a remote computer through network without having to understand the under layer network technologies [11].

RPC uses the client/server model. The requesting program is a client and the service-providing program is the server. First, the client sends a call message that includes the procedure parameters to the server process (Fig.1). After that, the caller process waits for a reply message. Next, a process on the server side, which is idle until the arrival of the call message, extracts the procedure parameters, computes the results, and sends a reply message. The server waits for the next call message. Finally, the caller's process receives the reply message, extracts the results of the procedure, and the caller resumes execution [11]. The server and the client application logic are kept independently, thus multiple clients may be written using the same server API.

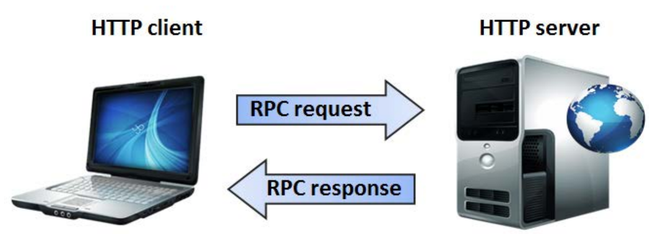

Fig. 1 Remote Procedure Call communication

According to the format of the sent messages, it is possible to consider XML-RPC or JSON-RPC. We decided to use JSON-RPC because JSON is not as verbose as XML and for humans, it is faster to write JSON. Despite that XML is used to describe structured data, it doesn't include arrays, while JSON does.

IV. Realization

As it was already mentioned, the developed service allows to process requests in JSON-RPC. The server provides responses according to the used RPC request protocol.

A. *Web Service Request*

For API implementation JSON-RPC 2.0 specification was used [8]. In such a case, the request object can contain three properties:

- method - a string containing the name of the method to be invoked;
- params - an array of objects to pass as arguments to the method. This property can be omitted;
- id - the request ID that can be of any type. It is used to match the response with the request that it is replying to.

The created web service implements only single method – generate. Its task is to produce LaTeX code that is able to generate the picture presenting plots with values specified in the input file.
Afterwards the generated code can be included to the LaTeX document and compiled to the final PS or PDF file. The created service enables to produce not only LaTeX code with the required graphics but also its PNG or PDF equivalent.

Fig.2 illustrates an example of the JSON-RPC request. As one can see, the request contains several parameters.


```
{"jsonrpc":"2.0",
 "method":"generate",
  "params":[{"APIkey":"xxxxx",
             "graph_type":0,
             "user_template":"pgfplots.tex",
             "user_input_data":"input.dat",
             "font_size":2,
             "pdf":1,
             "show_grid":1,"legend_show":1,"x_tics":1,
             "x_min":-10,"x_max":10,"y_min":-1.2,"y_max":1.2,
             "width":"15cm","legend_position":"outer north east",
             "x_label_pos_x":1.05,"x_label_pos_y":0.5,
             "y_label_pos_x":0.5,"y_label_pos_y":1.1,
             "lines":[{"file_index_x":0,"file_index_y":1,
                       "legend":"sinus"},
                      {"file_index_x":2,"file_index_y":3,
                       "legend":"kosinus",
                       "line_color":"green"}]}],
  "id":1}
```


Fig. 2 Example of JSON-RPC request

Parameter user_input_data enables to specify input file with data that should be visualised by means of graphical dependence. The data file is a simple plain text with 2 or more data columns that are separated by a space (Fig.3). The first column presents a variable that is visualized along the horizontal axis. All other columns define dependent variables that are plotted along the vertical axis. In the picture, one or more graphical dependences can be presented.

```
-10 0.54402111088937 -10 -0.83907152907645
-9.9 0.45753589377532 -9.9 -0.88919115262536
-9.8 0.36647912925193 -9.8 -0.93042627210475
-9.7 0.27176062641094 -9.7 -0.96236487983131
-9.6 0.17432678122298 -9.6 -0.98468785579413
-9.5 0.075151120461809 -9.5 -0.99717215619638
-9.4 -0.024775425453358 -9.4 -0.99969304203521
-9.3 -0.12445442350706 -9.3 -0.9922253254526
-9.2 -0.22288991410025 -9.2 -0.97484362140416
-9.1 -0.31909836234935 -9.1 -0.94772160213111
-9 -0.41211848524176 -9 -0.91113026188468
-8.9 -0.50102085645788 -8.9 -0.86543520924111
-8.8 -0.58491719289176 -8.8 -0.81109301406166
-8.7 -0.66296923008218 -8.7 -0.7486466455974
-8.6 -0.73439709787411 -8.6 -0.67872004732001
-8.5 -0.79848711262349 -8.5 -0.60201190268482
```

Fig. 3 Example of input data file

Parameter user_template allows to choose a LaTeX template that forms the environment for graphics included to the document source code. APIkey parameter specifies the alphanumerical code of the person that is allowed to use the presented service. Parameter pdf defines the format of the desired output. In similar way, one can define PNG or LaTeX format. The rest of parameters describe the graphical presentation and graphical layout of the resulting figure.

In Fig.4, one can see an example of LaTeX template that was used in the previous JSON-RPC request. As it is possible to see, the template is a standard LaTeX document that starts with the import of necessary packages and continues with setting of page style. The body of the document is created by the graphics that was completed using the presented service.

```
\documentclass{article}

\usepackage{pgfplots}
\usepackage{fullpage}

\pagestyle{empty}
\begin{document}

%EXPORT_BEGIN%
\begin{tikzpicture}
  \begin{axis}[axis x line=center, axis y line=center,
    xlabel={$x$}, ylabel={$y$},
    xlabel style={at={(1.05, 0.5)}},
    ylabel style={at={(0.5, 1.1)}},
    xmin=-10, xmax=10, ymin=-1.2, ymax=1.2,
    width=15cm, height=5cm,
    every axis/.append style={font=\small},
    xtick={-10,-9,...,10},
    grid, legend pos=outer north east]

  \addplot[smooth, line width=1pt, color=red, style=solid]
          table[x index=0, y index=1] {input.dat};
  \addplot[smooth, line width=1pt, color=green, style=solid]
          table[x index=2, y index=3] {input.dat};
  \legend{$sinus$, $kosinus$}

  \end{axis}
\end{tikzpicture}
%EXPORT_END%

\end{document}
```

Fig. 4 LaTex template specified in JSON-RPC request (pgfplots.tex)

B. Web Service Response

The response object also has three properties:

- result - the object that was returned by the invoked method;
- error - an error object if there was an error invoking the method;
- id - the same id as the corresponding request id.

It is to say that in the JSON-RPC 2.0 specification, each response object can contain only one of the first two properties (either "result" or "error").

The successfully realized response to the request from the previous example is shown in Fig.5.

The response returns the URL address of the generated PDF file. In the case when the LaTeX code was requested, the result property contains the code that can be seen in Fig.3 between strings EXPORT_BEGIN and EXPORT_END.

If the request fails, the response will contain the error property as it is demonstrated e.g. in Fig.6.


```
{"jsonrpc":"2.0",
 "result":"http://www.example.sk/latex/tmp/139357/pgfplots.pdf",
 "id":1}
```


Fig. 5 Example of successful JSON-RPC response


```
{"jsonrpc":"2.0",
 "error":{"code":1,"message":"Wrong API key."},
 "id":1}
```


Fig. 6 Example of failure JSON-RPC response

To extend the presented web service, we also developed the online web application (Fig.7). It enables to set all graph parameters via online form. Then, after uploading the file containing numerical data that should be visualized as plot(s) in the picture, choosing the preferred library and the output format, the generation of the required result can start. The created graphical user interface offers a comfortable way of the picture production.

V. Conclusion

The paper presents a service that can facilitate preparation of scientific documents written in LaTeX - the documents that include plots of measured or calculated values. The required LaTeX code that describes the picture is generated automatically using one of two available LaTeX packages: Pgfplots and Gnuplottex. The introduced tool enables to take into account user's preferences regarding the layout and the appearance of produced pictures. In this way, it offers extension to the LaTeX usability for unskilled users, too. It can also be used by scientists, teachers and students during writing their final thesis.

Acknowledgment

Authors thank to Pavol Buzák for his help with problem solution and problem algorithmization.

References

[1] H. M. El-Bakry and N. Mastorakis," Performance Evaluation of XML Web Services for Real-Time Applications", *International Journal of Communications*, Issue 2, Vol. 3, 2009.

[2] M. Goossens, F. Mittelbach, S. Rahtz, D. Roegel, H. Voβ , "*The LaTeX Graphics Companion*", Second Edition, Addison-Wesley, 2007.

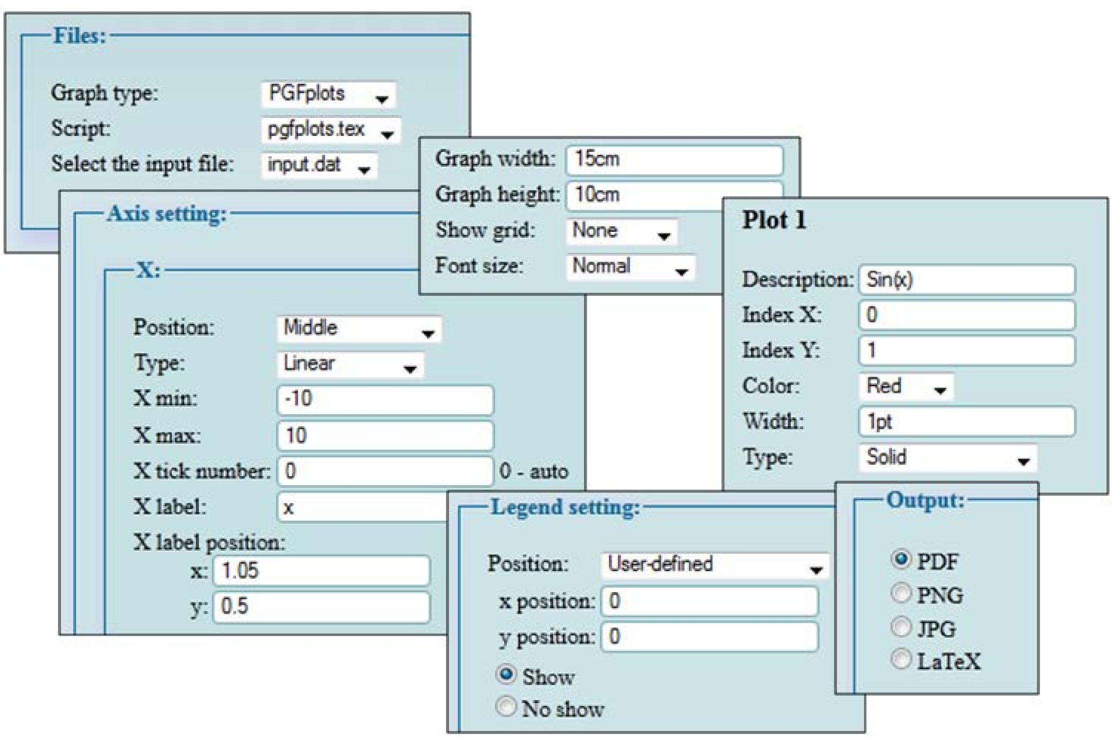

Fig. 7 Web interface for the LaTeX plots generation

[3] J. D. Hobby et al., "*METAPOST a user's manual* v.: 1.503", 2013. [online] Available at http://ftp.cvut.cz/tex-archive/graphics/metapost/base/manual/mpman.pdf [Accessed 02 January 2014].

[4] R. Koebler, "Simple is better: RPC / JSON-RPC", 2008. [online] Available at http://www.simple-is-better.org/rpc/ [Accessed 27 February 2014].

[5] E. Králiková, M. Kamenský, J. Červeňová, "E-Learning Support in Engineering Education of Robotics Specialists", *Proc. of Int. Conf. Distance Learning, Simulation and Communication* "DLSC", Brno, Czech Republic, IDET 2013, pp. 121-127, 2013.

[6] P. M. Lester, "A Picture's Worth a Thousand Words?", 2014. [online] Available at http://commfaculty.fullerton.edu/lester/writings/letters.html [Accessed 27 February 2014].

[7] Z.Mahmood, "The Promise and Limitations of Service Oriented Architecture", *International Journal of Computers*, Issue 3, Vol. 1, 2007.

[8] M. Morley, " JSON-RPC", 2013. [online] Available at http://json-rpc.org/ [Accessed 10 May 2013].

[9] L. Noabeb, "Anatomy of a Web Service: XML, SOAP and WSDL for Platform-independent Data Exchange", 2013. [online] Available at http://www.webreference.com/authoring/web_service/index.html [Accessed 09 May 2013].

[10] A. Ploscar, "XML - RPC vs. SOAP vs. REST web services in Java – uniform using WSWrapper", *International Journal of Computers*, Issue 4, Vol.6, 2012.

[11] RPC protocol. [online] Available at http://www.javvin.com/protocolRPC.html [Accessed 09 May 2013].

[12] A. Singhal, M. Gunestas, D. Wijesekera, "Forensics Web Services (FWS)", NIST Interagency Report 7559, Computer Security Division, Information Technology Laboratory, National Institute of Standards and Technology, Gaithersburg, MD 20899-8930, June 2010.

[13] C. Sliwa, "Defining Web Services Is No Easy Task", Computerworld, September 2002. [Online]. Available: http://www.computerworld.com/article/2577306

[14] P. Slivka, "Design of API for CAS service", Diploma thesis, FEI STU Bratislava, in Slovak, 2012.

[15] Ľ. Stuchlíková, J. Benkovská, M. Donoval, "An Easy and Effective Creation of E-Learning Courses", *Proc. of Int. Conf. Distance Learning, Simulation and Communication* "DLSC", Brno, Czech Republic, IDET 2013, pp. 159 – 164, 2013.

[16] A. J. Turgeon. Implications of Web-Based Technology for Engaging Students in a Learning Society", *Journal of Public Service and Outreach* 2(2): 32-37.

[17] I. Zolotová, R. Mihaľ, R. Hošák, "Objects for Visualization of Process Data in Supervisory Control", *Topics in Intelligent Engineering and Informatics 2: Aspects of Computational Intelligence: Theory and Applications*. - Berlin Heidelberg: Springer-Verlag, pp. 51-61, 2013.