Yxilon – A Client/Server Based Statistical Environment

Wolfgang Härdle*
Sigbert Klinke*
Uwe Ziegenhagen*

* Humboldt-Universität zu Berlin, Germany

This research was supported by the Deutsche Forschungsgemeinschaft through the SFB 649 "Economic Risk".

http://sfb649.wiwi.hu-berlin.de
ISSN 1860-5664

SFB 649, Humboldt-Universität zu Berlin
Spandauer Straße 1, D-10178 Berlin
Yxilon – A Client/Server Based Statistical Environment*

May 30, 2007

Wolfgang Härdle, Sigbert Klinke, Uwe Ziegenhagen
Humboldt-Universität zu Berlin
School of Business and Economics
Unter den Linden 6
10099 Berlin, Germany
haerdle/sigbert/ziegenhagen@wiwi.hu-berlin.de

Keywords  e-learning, statistical software

JEL classification  C88

Along with many others, we agree that a modern education in statistics needs to incorporate the practical analysis of real datasets, which are usually more complex than the common examples found in standard textbooks.

The software used in the teaching of statistics includes standard spreadsheet environments such as OpenOffice and Excel and dedicated commercial and non-commercial packages such as R, Minitab or SPSS. With the freely available Yxilon environment we add another package and proliferate the statistical programming language XploRe, using a modern client/server based architecture. This architecture has the capabilities of serving

*This research was supported by the Deutsche Forschungsgemeinschaft through the SFB 649 “Economic Risk”.

1
statistical results in a variety of flavors for different groups of users. In this paper we describe the general setup of the Yxilon environment and present selected technical details.

1 From XploRe to Yxilon

Following the requirements we gave in Härdle, Klinke & Ziegenhagen (2004), the value of a statistical software environment does not only depend on general features such as stability of the software and the set of available methods, but also on the way it allows an efficient interaction with the user. Nowadays the user operates statistical software either via character input in a terminal or mouse. Well-known packages for the former method are R and Matlab, for the latter Minitab and SPSS, which also allows the extension of its methods via VisualBasic.

XploRe, the predecessor of Yxilon and its scientific fundament, belonged to the category of language-operated statistical packages. The codes, the so-called quantlets, were written in a C-style programming language, which was then evaluated by XploRe. Details of the XploRe syntax can be found in Härdle, Klinke & Müller (2000). XploRe included more than 1500 quantlets, covering especially non- and semiparametric methods and the statistics of financial markets. The internal design of XploRe was monolithic, which means that a large proportion of components was tied closely together in one executable. This made the maintenance of the code complicated and led us to the decision to reimplement the XploRe language in the Yxilon project, using a simplified and modular client/server approach with two tiers. On the one hand this allows the easier adding of new modules, on the other hand existing modules can easily be maintained or exchanged.

2 The Client

The Yxilon environment is operated by the user through the client. In contrast to the server which was written in C++, it was implemented in Sun Microsystems’ Java. For our purpose Java offers various advantages if compared with other programming languages. Compiled Java code can be run on any platform for which a Java Runtime Environment (JRE) is available, it offers a large set of methods for internet-related techniques (TCP/IP, XML, SOAP), it can be embedded into HTML pages and is supported by a large user community.

The client satisfies different needs: It hides the technical implementation of the communication and offers an test platform for the implementation of new functions. The current version does not only offer basic editing and the exchange of code and results
with the server but provides advanced editing features such as the highlighting and autocompletion of code and database connectivity.

When the user presses the connect button, the client reads the communication settings from a text files and exchanges information with the server, e.g., login and password. After this handshake process the user edits his sourcefiles and sends them to the server.

Figure 1: Yxilon Java client

For the display of graphics under Java there are different solutions. We have chosen the Jaspplot library (Yamamoto, Nakano, Fujiwara & Kobayashi 2004), a library explicitly designed for the display of statistical graphics. This library can easily be embedded into Yxilon and offers a wide range of statistical plots. It furthermore allows the linking of different plots, so selected (brushed) observations in one plot are automatically highlighted in another.

Figure 1 depicts a screenshot of the Yxilon client, showing relevant windows of the interface. In the topleft corner the editor window with the Yxilon code needed to produce the scatterplot, in the bottom left the 'Debug' window, which shows selected information.
about the communication packages sent and received, and the output window used for
the display of numerical results. Furthermore, the client also contains a database wizard,
which allows to connect to different databases such as MySQL, MS Access or Oracle
via the Java Database Connector (JDBC).

3 The Server and the Communication Protocol

For the communication between a server and its client, there are numerous ways to
exchange information. We evaluated common techniques such as Remote Procedure Calls
(RPC) and Remote Method Interaction (RMI) but finally decided to develop our own
binary protocol. This approach had the disadvantage that a significant amount of time
had to be spent on low-level network programming but provided the advantage that we
kept the complete control of the communication, which resulted in a very small overhead,
therefore in a very fast communication.

Table 1 gives an overview about different types of objects currently used for the commu-
nication. There are ten types of objects: different types of lists, numeric and character
matrices as well as a different types of exceptions. These exceptions are extremely useful
when the code submitted by the user is incomplete or erroneous since they contain
detailed information about the location and type of the error. A runtime exception is
usually caused by errors in the evaluation of syntactically correct programs, e.g. when a
dll could not be loaded properly. User exceptions are defined in the Yxilon code. Their
task is to give feedback in form of warnings, if for example conditions concerning the size
and values of matrices are not met. The parser exception is the third type, capturing
incomplete input. Here the user receives a message that the input, he or she submitted,
had syntactical errors.

When the server executable is started, it reads the necessary settings from a textfile, e.g.
the network port it binds to, the paths for the dlls and quantlets and the login/password
needed for clients to connect. After the successful loading of these settings a TCP/IP
server port is opened. From now on clients can connect.
<table>
<thead>
<tr>
<th>ID</th>
<th>Object Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>empty object</td>
<td>to denote null objects</td>
</tr>
<tr>
<td>1</td>
<td>numeric matrix</td>
<td>stores numeric matrices (up to 3 dimensions)</td>
</tr>
<tr>
<td>2</td>
<td>string matrix</td>
<td>stores string objects (up to 3 dimensions)</td>
</tr>
<tr>
<td>3</td>
<td>object list</td>
<td>may contain matrices or lists</td>
</tr>
<tr>
<td>4</td>
<td>named list</td>
<td>a named list object</td>
</tr>
<tr>
<td>5</td>
<td>parameter list</td>
<td>stores named key/value pairs</td>
</tr>
<tr>
<td>9</td>
<td>runtime exception</td>
<td>captures errors in the server</td>
</tr>
<tr>
<td>10</td>
<td>user exception</td>
<td>allows user to define own errors/warnings</td>
</tr>
<tr>
<td>13</td>
<td>parser exception</td>
<td>captures incomplete input</td>
</tr>
</tbody>
</table>

**Table 1: Overview of Communication Objects**

When Yxilon objects are sent from the server to the client as a serialized data stream, the first information transmitted is the object type. Based on this type identification the client reads the remaining bytes and generates the necessary objects as matrices and lists which are then used e.g. to generate a statistical graphics or character output.

The server component of Yxilon is currently available for Microsoft Windows only, however we will provide versions for Linux and Solaris in the future. Due to the concept of modularity and exchangeable parts the server itself has only very limited functionality, such as the management of objects, basic calculations and the communication stack. All computational functionality above the level of simple algebra and basic matrix manipulations has been sourced out to dynamic link libraries (DLLs). Listing 1 illustrates the simple structure of one of these libraries.

Following the initial import statements we define in the `LibMain` function a set of variables such as the author and the version of the DLL. These information, accessible also from Yxilon code, provide valuable information in the development process, e.g. to ensure that a required dynamic link library has a certain version level. The current list of available libraries for Yxilon contains the handling of matrices, mathematical functions and the generation of various random numbers, further libraries are planned.
#include <LibInclude.hpp>
#define CExtObj CExpCompObject

EXPORT LibMain(CExtLibrary *pLib){
    pLib->setAuthor(L"Sheldon Kelly");
    pLib->setVersion(L"1.0");
    pLib->setName(L"Mathematical Functions");
    pLib->addFunction(L"abs", L"Yabs");
    pLib->addFunction(L"ceil", L"Yceil");
    pLib->addFunction(L"floor", L"Yfloor");
    pLib->addFunction(L"cos", L"Ycos");
};

Listing 1: Example of C++ code for a dynamic link library

4 Conclusion

The Yxilon project is our attempt to proliferate the ideas and concepts of the XploRe language in a modern, modular environment. By using innovative user features together with a fast and reliable communication protocol we want to provide an easily learnable environment especially suitable for the education of students. The next steps on the agenda include the further integration of statistical graphics into the client and the conversion of existing XploRe code to Yxilon. We highly appreciate feedback from interested students and scientists, more information and downloads can be found online at [http://www.quantlet.org](http://www.quantlet.org).

REFERENCES


SFB 649 Discussion Paper Series 2007

For a complete list of Discussion Papers published by the SFB 649, please visit http://sfb649.wiwi.hu-berlin.de.

002 "Robust Risk Management. Accounting for Nonstationarity and Heavy Tails" by Ying Chen and Vladimir Spokoiny, January 2007.
007 "Rules, Discretion or Reputation? Monetary Policies and the Efficiency of Financial Markets in Germany, 14th to 16th Centuries" by Oliver Volckart, February 2007.
009 "Union Wage Compression in a Right-to-Manage Model" by Thorsten Vogel, February 2007.
010 "On σ-additive robust representation of convex risk measures for unbounded financial positions in the presence of uncertainty about the market model" by Volker Krätschmer, March 2007.
017 "Empirical Pricing Kernels and Investor Preferences" by Kai Detlefsen, Wolfgang Härdle and Rouslan Moro, April 2007.
019 "Regional and Outward Economic Integration in South-East Asia" by Enzo Weber, April 2007.
020 "Computational Statistics and Data Visualization" by Antony Unwin, Chun-houh Chen and Wolfgang Härdle, April 2007.
021 "Ideology Without Ideologists" by Lydia Mechtenberg, April 2007.

This research was supported by the Deutsche Forschungsgemeinschaft through the SFB 649 "Economic Risk".


027 "Long Memory Persistence in the Factor of Implied Volatility Dynamics" by Wolfgang Härdle and Julius Mungo, May 2007.

028 "Macroeconomic Policy in a Heterogeneous Monetary Union" by Oliver Grimm and Stefan Ried, May 2007.


036 "Yxilon – A Client/Server Based Statistical Environment" by Wolfgang Härdle, Sigbert Klinke and Uwe Ziegenhagen, June 2007.