Supplemental Material 2

MAKERGAUL: An innovative MAK-based model and software for real-time PCR quantification

Christoph André Bultmann and Ralf Weiskirchen

1. Security remarks

While using MAKERGAUL is very easy, the installing and configuring is more challenging. If you are not sure how to work with and administrate a server, please ask advice from a competent person or institution.
The authors explicitly point out that the use of MAKERGAUL should be performed on an internal, non-Internet-connected server. Because eligible entries already cause very high server loads, the program is an easy target for DOS attacks (denial of service). The security against script injections has not been tested for all variables in all modules. If you use makergaulxx, the server needs execution-rights which can be abused by an attacker. Unauthorized persons could gain access through undiscovered server vulnerabilities. Use of the software is at your own risk.

2. Structure

2.1. Design decision

To effectively apply the MAKERGAUL model on real-time PCR data, a variety of similar software based on PHP [1-3] HTML [4], CSS [5], JavaScript [6] and C++ [7-12] including the libraries GMP v. 5.1.2 [13] and MPFR v. 3.1.2 [14] were developed using further given tools and references [15, 16]. The decision to use a server-based program brings some advantages that we think are helpful for use in a laboratory:

Table 1: Advantages of MAKERGAUL

Power concentration	Computationally intensive processes take place only on the server; the performance of user's computer is irrelevant.
Maintenance	New program modules must only be integrated into one system, but are available directly for all users.
Data security	Generated data are not only stored in the user's client program but also as a central copy. Data loss is prevented, and data manipulation can be uncovered.
Platform independence	All users can use the software on their preferred computer.

2.2. Program structure

Starting point of the software is the real-time PCR file. It contains the fluorescence data and status information generated by the respective PCR cycler presented by the panel

object. Each function should be represented by a separate object. All function objects should act independently. All functions of the program aim to manipulate data of the disk object. The output of the program is as an HTML page. It is understood that each generated and displayed page corresponds to a program call (i.e., the request).

2.3. Data handling

The storage of data across multiple program invocations is done in global, cross-program variables (session variables).
In MAKERGAUL, only the central session variables (definition see below) should be addressed by all disk objects. Session variables, which are additionally required by a functional object, should be initialized in first start()- and unset in the unload()-member function of the object.
In preparing these variables, the object should always form an array, which bears the name of the module and contains all other session variables of the object. Cross-talk between modules over self-initialized session-variables is to avoid. Together, these rules prevent the occurrence of unwanted overlaps and thus errors when loading new modules.
Each module must have standardized methods for initialization, input and output, which are called by the main program:

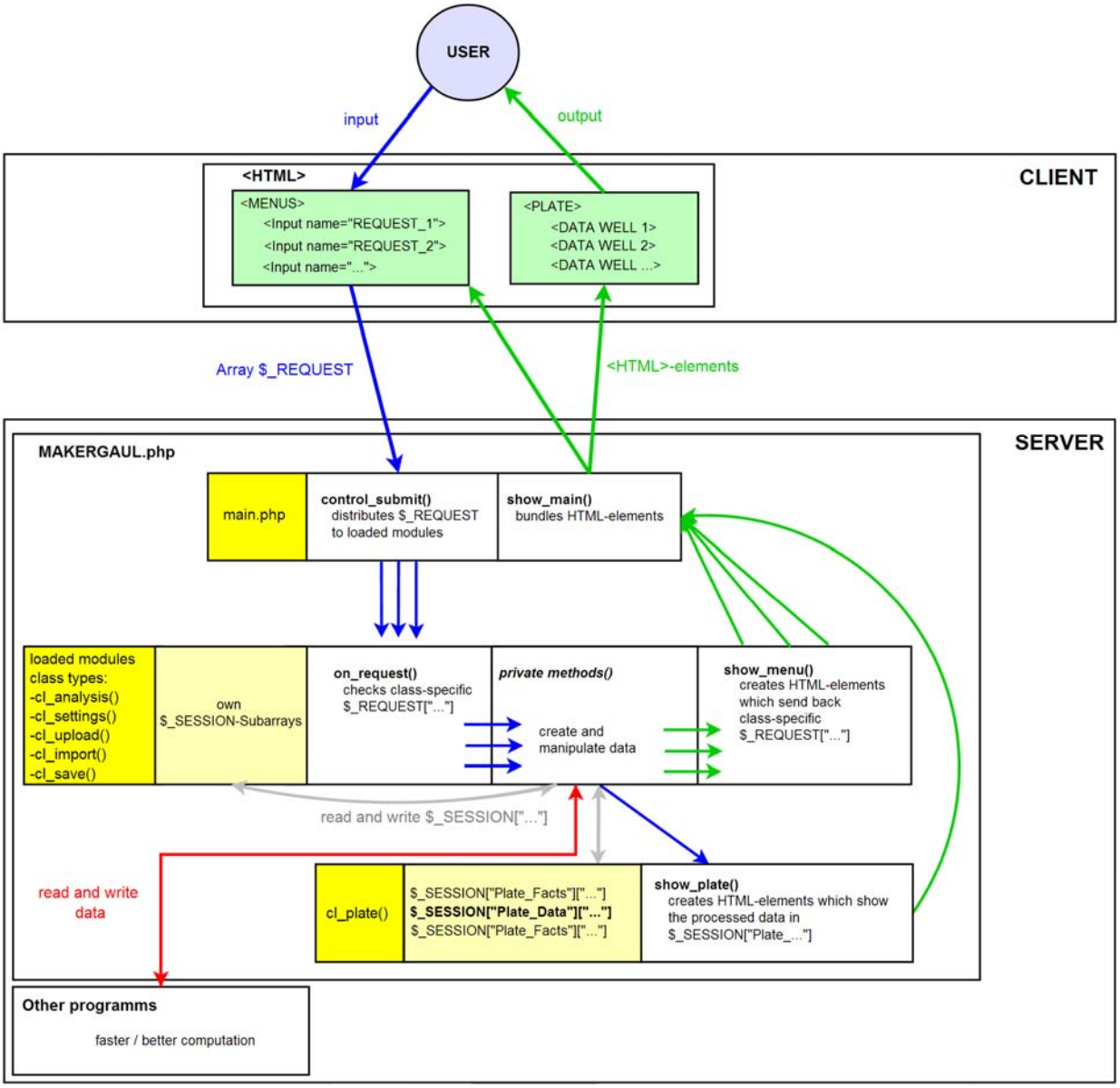
Table 2: Standard member-functions of MAKERGAUL-objects

Name of function	Function
static firststart()	Is active only when program is first started; the object's class initializes the object required by the session variables before first use.
on_request()	Processes the transmitted HTML requests that the object will accept (e.g., button pressed, changed fields, ...)
show_menu()	Generates the HTML code that provides the object (such as menu items)
unload()	Unsetting of the session variables formerly initialized in first start()

By use of this program strategy, it should be very easy to develop new analysis modules and quickly integrate them into the existing program structure.

The data flow that occurs during processing of MAKERGAUL is shown in Figure 1.

Figure 1: Data flow in MAKERGAUL



3. Components

3.1.1 index.php

This php-document is the starting point of the program. It manages including of the php-files (Modules of MAKERGAUL) in the subfolders “modules/main”, “modules/menus” and “modules/analysis”.

3.1.2 main. php

This file is included in "index.php" and provides the module interface functions.
Control of the modules is realized by calling and following to the standard member functions. Each available makergaul-module-class is described above. Incoming user-requests are allocated by control_submit() to the loaded modules (= objects). The

HTML-output from the objects to the main page is controlled and bundled *via* show_main().
The Main.php also provides the first bunch of main session variables:

Table 3: $_SESSION-Variables initialized by main.php

Name	Depth 1	Depth 2	Function
$_SESSION [„Module“]	[module]	[„Type“]	class name of the loaded module
		[„Active“]	The active object to address
$_SESSION [„Main“]	[„Show“]	Stores display-directives which are processed from show_main()	
	[„Error“]	Stores abortive errors to display	
$_SESSION ["Plate_Facts"]	[„Area“]	Number of wells (= data storage space) on the current plate	
	[„Rows“]	Number of rows (display-option)	
	[„Name“]	Name of the plate, e.g for creating files	

3.2. Plate-Class (cl_plate)
Contains all RTPCR-data in the main session array $_SESSION [“Plate_Data”]. This array is highly branched. Modules which read and write into must handle with care to prevent data loss and errors!

3.3 Modulchanger-Class (cl_modulechange_)
That menu communicates with the main.php to shift active modules.

3.4 Upload-Class (cl_upload_)
The upload module currently supports importing of *.csv files that were created with the *ABI 7300 Real Time PCR System*® and the re-import of MAKERGAUL-*.csv files. Under the setting "own" all *csv files are accepted that have the following structure:

Table 4: Structure of files that can be read by setting the "Own" in MAKERGAUL

File structure "own" *.csv file. First line: name line; line two to n: Data lines.					
Well	**Name**	**Detector**	**Cycle 1**	**Cycle 2**	**…**
1	e.g. Test-DNA	e.g. GAPDH	e.g. 1.2345	e.g. 1.5643	e.g. 2.9102
2	e.g. Std-DNA	e.g. Collagen	e.g. 0.1456	e.g. 0.9876	e.g. 1.2456
…	…	…	…	…	…
Delimiter: semicolon (;), meaning that all values in a row must be separated by a semicolon (;)					

3.5 Import-Class (cl_import_)

The button below each well allows loading data on a text entry window by "copy & paste". All consecutive numbers will be accepted as entries, irrespectively how they are embedded in the text. However, it is important to specify the decimal separator that is used within a number (point or comma).

3.6. Export-Class (cl_export_)

The export function is generated from the *.csv file from each plate that includes the analysis results (module dependent), the original measured values and the calculated curve values in a directory on the server and provides a link to download for the user.
Note: In the moment when there is no implementation of a garbage collection, all user generated files will be rested by the default in the “exported”-folder until the server-administrator removes them manually.

3.7 Settings-Class (cl_settings-)

Provides the menu to control the analysis-object. This Class-type needs some additional member functions:

Table 5: Additional member functions required in cl_analysis -objects

Name of function	Function
run_analysis()	Creates an analysis-object and starts the analysis.
static presets (par1, par2)	Can be called by analysis-modules to change the default presets.

3.8 Analysis-Class (cl_analysis_)

As the central part of the Makergaul software, these modules have to communicate with many other objects and bring along some specials:

Table 6: Additional member functions required in cl_analysis -objects

Name of function	Function	Returs	
static firststart()	An analysis-module must call requirement(par_type, par_module) to define the settings-class it will use.		
static result_pars()	Called by Plate. Has to return name and numbers of the result-parameters produced by this function in a structured array. This is needed to built plate on loading properly.	Array[0,1,…]	[“ID”] = name [“Value”] = preset
run_analysis(setting1, setting2, …)	Called by settings-object. Contains the user parameter and starts the analysis.		

The two standard modules for DNA analysis in MAKERGAUL are of the same analysis model (cf. Supplementary Material 1) and an implementation of the MAK2 model [17]. In the "Settings" menu the user can choose between the two models.
Analysis of the Well-Data is done in a three step process: Check which values to include, choose implementation (PHP or C++) and run analysis.

3.8.1 Communication-Class (cl_com_)
Objects of this type are used to run and control processes outside PHP. They only communicate with the creating analysis-module.

3.8.2 makergaulxx
By default, the MAKERGAUL-package is distributed with a C++ implementation of the basic algorithms MAKERGAUL and MAK2. MAKERGAUL_C is also available together with this additional component. All computations run much faster than the php variants. The compiled files are addressed *via* cl_com objects and stored in “modules/analysis/binarys”.
When binaries for Windows and Linux systems (both compiled for 32-Bit x86-CPUs) are available, also the C++ files can be found in the developer-folder to modify and compile the code for other systems.
makergaulxx can also be used as a stand alone application in which input and output of the program are processed through the command-line interface of the operating system. Thereby it is easy to fill the program by own applications. However, for proper functionality of the MAKERGAUL-package deactivation of this program should be avoided.
Please note that the server needs execution rights in the binary folder.

3.9 Checkbox-Class (cl_checkbox_)
Provides the menu to automatically check and uncheck group of wells.

3.10 Names Class (cl_names_)
That menu enables the user to change the names and detectors of a well.

3.11 expansion.php
This file contains improved functions for arbitrary precision math (less casting from string to float result in precision loss), colorize-functions for HTML-colours and other functions without clear categories.

4. Testing Recommendation
If you are not familiar in Server-administration and want to test the program on Windows, we suggest downloading and installing EasyPHP [2].
First you have to change the configuration of PHP. Start EasyPHP, the program will appear as a Symbol in the taskbar. Right click and open EasyPHP → Configuration → PHP. A Text file (php.ini) will appear.

Find and Replace entries you also find in the MAKERGAUL-File “development/php_ini_config_adds.txt” (searching dialog in most text editors is Ctrl+F). If there are entries you only find in php_ini_config_adds.txt, just copy these lines at the end of the php.ini. Save and quit the file.

After that you copy the contents of the folder “user” from the MAKERGAUL-Package into the www-folder of EasyPHP (for example “C:\Programs\Easy-PHP-12.1\www).
Right click EasyPHP →Local Web. Your Browser will open. If not, open your Browser manually and enter “127.0.0.1” in your browser line. The www-folder will then open in the Browser.
Choose index.php – and Makergaul will run.

References cited:

[1] PHP-Reference, http://www.php.net/ (retrieved May 15, 2013)

[2] EasyPHP 12.1, http://www.easyphp.org/ (retrieved May 15, 2013)

[3] Weaverslave 3.9.18, http://www.weaverslave.ws/ (retrieved May 15, 2013)

[4] HTML-Reference, http://de.selfhtml.org/ (retrieved May 10, 2013)

[5] CSS-Reference, http://www.w3schools.com/css/default.asp (retrieved May 2, 2013)

[6] JavaScript-Reference, http://www.w3schools.com/js/default.asp (retrieved May 15, 2013)

[7] C++-Reference, http://www.cplusplus.com/reference/ (retrieved June 23, 2013)

[8] J. Wolf. Grundkurs C++. 2. Edition, Galileo Computing, Bonn 2013. ISBN 978-3-8362-2294-5.

[9] Ryan J. Russell: Netbeans and MinGW-w64, http://stackoverflow.com/questions/8478317/netbeans-and-mingw-w64 (retrieved June 23, 2013)

[10] NetBeans IDE 7.3, http://netbeans.org/downloads/ (retrieved June 23, 2013)

[11] MinGW-w64 GCC in different versions, download at http://mingw-w64.sourceforge.net/download.php (retrieved June 26, 2013)

[12] MSYS 15.05.2013, rev13., http://sourceforge.net/projects/mingwbuilds/files/external-binary-packages/

[13] GMP-Reference, http://gmplib.org/manual/version 5.2.1 (retrieved June 23, 2013)

[14] MPFR-Reference,http://www.mpfr.org/mpfr-current/mpfr.html version 3.1.2 (retrieved June 23, 2013)

[15] Fedora 19, http://fedoraproject.org/de/ (retrieved July 10, 2013)

[16] GCC version 4.8.1, http://gcc.gnu.org/ (retrieved July 10, 2013)

[17] G.J. Boggy, P.J. Woolf, A mechanistic model of PCR for accurate quantification of quantitative PCR data, PLoS ONE 5(8) (2010) e12355.