Stata web services: Toward Stata-based healthcare informatics applications integrated in a service-oriented architecture (SOA)

Alexander Zlotnik  
Technical University of Madrid, ETSIT, DIE  
Ramon y Cajal University Hospital

Modesto Escobar  
Universidad de Salamanca

Ascensión Gallardo-Antolín  
UC3M, Department of Signals and Systems, Madrid

Juan Manuel Montero Martínez  
Technical University of Madrid, ETSIT, DIE

Stata is a registered trademark of StataCorp LP, College Station, TX, USA.
Why?
User-contributed programs

ssc install <program>

findit <program>
(runs both search and net search)

net from http://www.website.com/

manually copy program files to
C:\ado\plus\<subdir>\
Sometimes this is not enough
Sometimes your program...

... requires complex interactions with external software packages (ex: WinBUGS, MATLAB, Maxima, AnyLogic)

... uses proprietary data sources (ex: real-time currency exchange rates)

... uses proprietary source code
Sometimes your users...

... does not have the version of Stata your program requires (ex: it may require v14 and they may only have v12)

... does not have Stata at all (Stata is not very common in some fields)

... does not have a PC, but may have a smartphone with a web browser (ex: developing countries)
What if...?

Private data sources

External programs

Your program

Stata / Mata

Your server
What if...?

Private data sources

External programs

Your program

Stata / Mata

Web interface

Your server

Access from any device

- Web interface
  - Access from any device
What if...?

Private data sources

External programs

Your program

Stata / Mata

Web interface

Access from any device

Security: client isolation
Sometimes your program...

... requires **complex interactions** with **external software packages** (ex: WinBUGS, MATLAB, Maxima, AnyLogic)

... uses **proprietary data sources** (ex: real-time currency exchange rates)

... uses **proprietary source code**
What if…?

Private data sources

External programs

Your program

Stata / Mata

Web service

XML
What if...?

Web service

XML

WSDL
SOAP

Web applications

Desktop applications

Mobile applications

native iOS apps
native Android apps
Service-oriented architecture

Web service

XML

WSDL

SOAP

Enterprise Service Bus

Other web services
How?
Option 1:
Translate Stata / Mata program into a general-purpose programming language used in web applications. Ex: Java, C / C++, C#, ASP.net + VB.net, Python, Ruby, etc
How?

Your program

Option 1:
Translate Stata / Mata program into a general-purpose programming language used in web applications. Ex: Java, C / C++, C#, ASP.net + VB.net, Python, Ruby, etc

- Few numerical libraries
- May not have the same functions
- Functions may not be implemented in the same way
  - subtle errors
  - numerical precision issues
  - performance issues
How?

Your program

Option 2:
Translate Stata / Mata program into
R & RShiny or SAS Stored Process Web Application
Option 2: Translate Stata / Mata program into R & RShiny or SAS Stored Process Web Application

- Still requires a **laborious translation** in most cases

- Again, functions may **not** be implemented in the same way

- **RShiny** is a nice alternative but the free version only supports **one concurrent session**
How?

Option 3:
Use a slightly modified version of your existing Stata program in a web application.

Stata is a registered trademark of StataCorp LP, College Station, TX, USA, and the Stata logo is used with the permission of StataCorp.
How?

Option 3: Use a slightly modified version of your existing Stata program in a web application.

-- In this presentation, we will see how to build a web application/web service using your Stata program, with minimal modifications based on Stata/IC, Stata/SE or Stata/MP.

-- Very similar techniques can be used with Numerics for Stata.
Technologies

Program core: **Stata + Mata**

Web application language: **PHP**

Web server: **Apache**

Operating system: **Windows**
Technologies

Program core: **Stata + Mata**

Web application language: **PHP**

Web server: **Apache**

Operating system: **Windows**

Well-known

Easy to use
Technologies

Program core: **Stata + Mata**

Web application language: **PHP**

Web server: **Apache**

Operating system: **Windows**

- Well-known
- Easy to use
Web application language

**PHP** implementation example

*Other* languages may also be used:
- Java (servlets, JSPs)
- Python
- ASP / ASP.net + C# / VB.net
- C/C++, Perl (CGI interface)
- *et cetera*
Web server

**Apache** implementation example

**Other** web servers, application containers and application servers may also be used:
- Tomcat
- JBoss
- Oracle WebLogic
- IBM WebSphere
- Magic xpa
- *et cetera*
Operating system

It should be possible to do this on any operating system that supports Stata (i.e. Windows, Unix/Linux, Mac OS X).
General idea

Web interface
(HTML / JS)

Web application
(PHP / Java / ASP.net + C# / etc...)

Program written in
Stata / Mata

Web server / Application server

Stata IC / SE / MP

Operating system
Calling Stata

- Web interface (HTML / JS)
- Web application (PHP / Java / ASP.net + C# / etc...)
- Web server / Application server
- Program written in Stata / Mata
- Stata IC / SE / MP
- Operating system

Stata command(s)
Getting a response from Stata

Web interface (HTML / JS)

Web application (PHP / Java / ASP.net + C# / etc…)

Web server / Application server

Program written in Stata / Mata

images
data files
text files
log files

Stata IC / SE / MP

Operating system
Simplified example
Calling Stata

Web interface (HTML / JS)

Web application (PHP / Java / ASP.net + C# / etc…)

Web server / Application server

Program written in Stata / Mata

Stata IC / SE / MP

Operating system

Stata command(s)
Calling Stata

Stata command(s):

Send command(s) to Stata
Calling Stata

<html>
<head> Web interface </head>
<body>
<form action="call_stata.php" method="post">
  Stata command(s):
  <textarea name="stata_commands" ></textarea>
  <input type="submit" value="Send command(s) to Stata" >
</form>
</body>
</html>
Calling Stata

call_stata.php

<?php
...

$stata_commands = $_POST["stata_commands"];

write_stata_do_file($stata_commands);

execute_stata_do_file();

...
>

...
Calling Stata

Our web application will execute:

```
<path_to_Stata>/Stata.exe /q /e do "commands.do"
```

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>/q</td>
<td>suppress logo and initialization messages</td>
</tr>
<tr>
<td>/e</td>
<td>set background (batch) mode and log in ASCII text without prompting when Stata command has completed</td>
</tr>
</tbody>
</table>

(Stata User’s Guide, section [B.5])
Calling Stata

Our web application will execute:

<path_to_Stata>/Stata.exe /q /e do "commands.do"

We’ll previously write our commands here

$stata_commands
Calling Stata

Our web application will execute:

<path_to_Stata>/Stata.exe /q /e do "commands.do"

We’ll previously write our commands here

Example:
  cd <path_to_temp_folder>
  sysuse auto
  histogram price
Calling Stata

Web interface (HTML / JS)

Web application (PHP / Java / ASP.net + C# / etc…)

Program written in Stata / Mata

Web server / Application server

Stata IC / SE / MP

Operating system
Calling Stata

**Problem:** modern versions of Stata will **not** work if called directly from a web server (SYSTEM user).

```
stata.exe /e /q ...
```

Web application

PHP: `shell_exec()`
Calling Stata

**Problem:** modern versions of Stata will **not** work if called directly from a web server (SYSTEM user).

**Solution:** wrapper + user impersonation

Web application

`shell_exec()`

`stata.exe /e /q ...`

Wrapper library

Stata IC / SE / MP
Getting a response from Stata

Web interface (HTML / JS)

Web application (PHP / Java / ASP.net + C# / etc…)

Web server / Application server

Program written in Stata / Mata

Stata IC / SE / MP

Operating system
Getting a response from Stata

Our web application will execute:

```bash
<path_to_Stata>/Stata.exe /q /e do "commands.do"
```

We’ll previously write our commands here

Example:
```
  cd <path_to_web_folder>/img/
sysuse auto
  histogram price, normal saving(graph01, replace)
  graph export graph01.png, replace
```

Now our web application will be able to display

```bash
<path_to_web_folder>/img/graph01.png
```
Getting a response from Stata

call_stata.php

<?php
...

$stata_commands = $_POST["stata_commands"];
write_stata_do_file($stata_commands);
execute_stata_do_file();
display_results(); //display graph01.png

...?

?>
Getting a response from Stata

call_stata.php

<?php
...

function display_results() {
    echo "<html>";
    echo "  <head>Result</head>";
    echo "  <body>";
    echo "    <img src=img/graph01.png>";
    echo "  </body>";
    echo "</html>";
}
...
?>
Getting a response from Stata
Basic security

SQL injection attack:

'\; DROP TABLE users;
Basic security

Prevent “Stata injection” attacks:

- Limited, sanitized inputs,

Ideally, **no free text** fields on the web interface

- Avoid or restrict `shell()`, `xshell()`, `winexec()` in your Stata program
Basic security

Stata command(s):

Send command(s) to Stata

Bad practice

Dataset: auto

Command: histogram

☐ Add normal-density plot

☐ Add kernel-density plot

Send command(s) to Stata

Better practice
Basic security

Prevent "Stata injection" attacks:

- Limited, sanitized inputs,
  Ideally, **no free text** fields on the web interface

- Avoid or restrict `shell()`, `xshell()`, `winexec()` in your Stata program
Basic security

Bad practice

```plaintext
*! version 1.00.0
*authors:
program myshell
version 12
syntax [, ///
cmd(string)]
shell("\'command'")
end program
```

Better practice

```plaintext
*! version 1.00.0
*authors:
program myshell_better
version 12
syntax [, ///
params(string)]
//only pass parameters to a specific command
shell("externalprogram.exe ""\'params'""
) end program
```
Basic security

**Bad practice**

```plaintext
*! version 1.00.0
*authors:
program myshell
version 12
syntax [, //
cmd(string)]
shell("`command'")
end program
```

**Better practice**

```plaintext
*! version 1.00.0
*authors:
program myshell_better
version 12
syntax [, //
params(string)]
//only pass parameters to a specific command
shell("externalprogram.exe ""`params'""")
end program
```

It's even better to avoid dynamic `shell()` commands if Stata is executed through a web interface.
Implementation examples
Web interface for –coin–

Studying coincidences with network analysis and other multivariate tools

### Analysis of coincidences

#### Main

- **Variables:**
  - Vars. head
  - (Omni, Axiom, Affi, Exomel, OF)
  - Legend:

#### Outputs

- **Frequencies**
- **Column percentages**
- **Row percentages**
- **Odds ratios**

#### Residuals and significances

- **Residuals**
- **Stardarized residuals**
- **Normalized**
- **Significance or normalized residuals**

#### Other options

- **Distance matrix**
- **Adjacency matrix**
- **Similarity matrix**
- **p =** 0.5
- **Support**
  - 0

### Buttons

- **OK**
- **Cancel**
### Execution output:

#### Omni,Axiom,Affy,Exome/LOF

<table>
<thead>
<tr>
<th>Genotypes</th>
<th>Counts</th>
</tr>
</thead>
<tbody>
<tr>
<td>Has Omni Genotypes</td>
<td>2098</td>
</tr>
<tr>
<td>Has Axiom Genotypes</td>
<td>880</td>
</tr>
<tr>
<td>Has Affy 6.0 Genotypes</td>
<td>3</td>
</tr>
<tr>
<td>Has Exome/LOF Genotypes</td>
<td>2055</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>Genotypes</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Omni</td>
</tr>
<tr>
<td>Has Omni Genotypes</td>
<td>100.0</td>
</tr>
<tr>
<td>Has Axiom Genotypes</td>
<td>41.9</td>
</tr>
<tr>
<td>Has Affy 6.0 Genotypes</td>
<td>0.1</td>
</tr>
<tr>
<td>Has Exome/LOF Genotypes</td>
<td>98.0</td>
</tr>
</tbody>
</table>
female, male, Omni, Axiom, Affy, Exome/LOF
female, male, Has Omni Genotypes, Has Axiom Genotypes, Has Affy 6.0 Genotypes, Has Exome/LOF Genotypes

<table>
<thead>
<tr>
<th>Frequencies</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>female</td>
<td>1760</td>
</tr>
<tr>
<td>male</td>
<td>0</td>
</tr>
<tr>
<td>Has Omni Genotypes</td>
<td>1070 1028 2098</td>
</tr>
<tr>
<td>Has Axiom Genotypes</td>
<td>488 489 880 977</td>
</tr>
<tr>
<td>Has Affy 6.0 Genotypes</td>
<td>681 614 3 0 1195</td>
</tr>
<tr>
<td>Has Exome/LOF Genotypes</td>
<td>1062 1001 2055 881 3 2063</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Col. percentages</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>female</td>
<td>100.0 0.0 51.0 49.9 48.6 51.5</td>
</tr>
<tr>
<td>male</td>
<td>0.0 100.0 49.0 50.1 51.4 48.5</td>
</tr>
<tr>
<td>Has Omni Genotypes</td>
<td>60.3 59.1 100.0 90.1 0.3 99.6</td>
</tr>
<tr>
<td>Has Axiom Genotypes</td>
<td>27.7 28.1 41.9 100.0 0.0 41.3</td>
</tr>
<tr>
<td>Has Affy 6.0 Genotypes</td>
<td>33.0 35.3 0.1 0.0 100.0 0.1</td>
</tr>
<tr>
<td>Has Exome/LOF Genotypes</td>
<td>60.3 57.5 98.0 87.1 0.3 100.0</td>
</tr>
</tbody>
</table>
ACB, ASW, BEB, CDX, CEU, CHB, CHS, CLM, ESN, FIN, GBR, GIH, GWD, IBS, ITU, JPT, KHV, LWK, MSL, MXL, PEL, PJL, PUR, STU, ACB, ASW, BEB, CDX, CEU, CHB, CHS, CLM, ESN, FIN, GBR, GIH, GWD, IBS, ITU, JPT, KHV, LWK, MSL, M

<table>
<thead>
<tr>
<th>Frequencies</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>ACB</td>
<td>123</td>
</tr>
<tr>
<td>ASW</td>
<td></td>
</tr>
<tr>
<td>BEB</td>
<td></td>
</tr>
<tr>
<td>CDX</td>
<td></td>
</tr>
<tr>
<td>CEU</td>
<td></td>
</tr>
<tr>
<td>CHB</td>
<td></td>
</tr>
<tr>
<td>CHS</td>
<td></td>
</tr>
<tr>
<td>CLM</td>
<td></td>
</tr>
<tr>
<td>ESN</td>
<td></td>
</tr>
<tr>
<td>FIN</td>
<td></td>
</tr>
<tr>
<td>GBR</td>
<td></td>
</tr>
<tr>
<td>GIH</td>
<td></td>
</tr>
<tr>
<td>GWD</td>
<td></td>
</tr>
<tr>
<td>IBS</td>
<td></td>
</tr>
</tbody>
</table>

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0</td>
<td>112</td>
</tr>
<tr>
<td></td>
<td>144</td>
<td></td>
</tr>
<tr>
<td></td>
<td>109</td>
<td></td>
</tr>
<tr>
<td></td>
<td>183</td>
<td></td>
</tr>
<tr>
<td></td>
<td>108</td>
<td></td>
</tr>
<tr>
<td></td>
<td>171</td>
<td></td>
</tr>
</tbody>
</table>
Web interface for –nomolog–

A general-purpose nomogram generator for predictive logistic regression models

Zlotnik A, Abraira V. Stata Journal. 2015. Volume 15, Number 2

URL: http://www.zlotnik.net/stata/nomograms
### nomolog - Logistic regression nomogram generator

<table>
<thead>
<tr>
<th>Main</th>
<th>Variable ranges and decimals</th>
<th>Prob. values</th>
<th>cont # cont interactions</th>
<th>Regression command</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Graph title**
- Nomogram

- [ ] Use variable description as variable label (default: no)
- [ ] Show data values on dummy data value labels (default: no)
- [ ] Display table with variable divisions and corresponding scores (default: no)
- [x] Simplify interactions (default: yes)  
- [x] Negative values in red (default: yes)

**Size of variable name labels (default: 2.2)**
- 2.2

**Max N of chars to display in variable name labels (default: 240)**
- 240

**Size of data labels (default: 2)**
- 2

**Max N of chars to display in data labels (default: 100)**
- 100
In the web implementation, we must add a tab for loading the dataset and executing the logistic regression command.
Questions?
Credits

Special thanks to all the people who made and released these design resources for free:

◎ Presentation template by SlidesCarnival
◎ Photographs by Unsplash & Death to the Stock Photo (license)