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

Web interface

Your server

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

Web applications

Desktop applications

Mobile applications

native iOS apps native Android apps

WSDL

SOAP
Service-oriented architecture

Web service

XML

WSDL
SOAP

Enterprise Service Bus

Other web services
How?
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?

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?

Your program

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

- Open source
- 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…)

Program written in Stata / Mata

Web server / Application server

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

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

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

Stata command(s)
**Calling Stata**

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

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

Diagram:
- Web application
  - PHP: `shell_exec()`
  - `stata.exe /e /q ...`
- Stata IC / SE / MP
Calling Stata

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

**Solution:** wrapper + user impersonation
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

images
data files
text files
log files

Operating system
Getting a response from 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_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

```
<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>
    <head>Result</head>
    <body>
    <img src=img/graph01.png>
    </body>
    </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
1  *! version 1.00.0
2  *authors:
3  program myshell
4  version 12
5  syntax [, ///
6  cmd(string)]
7  shell("\`command\``)
8
9  end program
```

**Better practice**

```plaintext
1  *! version 1.00.0
2  *authors:
3  program myshell_better
4  version 12
5  syntax [, ///
6  params(string)]
7
8  //only pass parameters to a specific command
9  shell("externalprogram.exe ""\`params\``"")
10
11  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

coin - Analysis of coincidences

Main
- Variables:
  - Vars. head
  - (Omn, Axiom, Affi, ExomELOF)
Legend: [ ]

Plots

Output options
- Outputs
  - Frequencies
  - Column percentages
  - Row percentages
  - Odds ratios
- Residuals and significances
  - Residuals
  - Standardized residuals
  - Normalized
  - Significance or normalized residuals

- Distance matrix: [ ] haberman [ ]
- Adjacency matrix: [ ]
- Centrality measures: [ ]
- Bonferroni-adjusted: [ ]

[OK] [Cancel]
Execution output:

| Omni, Axiom, Affy, Exome/LOF |
|------------------------------|--------------------------------|
| Has Omni Genotypes           | 2095                           |
| Has Axiom Genotypes          | 880                            |
| Has Affy 6.0 Genotypes       | 3                              |
| Has Exome/LOF Genotypes      | 2055                           |

**Frequencies**

<table>
<thead>
<tr>
<th>Has Omni Genotypes</th>
<th>2095</th>
</tr>
</thead>
<tbody>
<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>

**Col. percentages**

<table>
<thead>
<tr>
<th>Has Omni Genotypes</th>
<th>100.0</th>
<th>90.1</th>
<th>0.3</th>
<th>99.6</th>
</tr>
</thead>
<tbody>
<tr>
<td>Has Axiom Genotypes</td>
<td>41.9</td>
<td>100.0</td>
<td>0.0</td>
<td>41.3</td>
</tr>
<tr>
<td>Has Affy 6.0 Genotypes</td>
<td>0.1</td>
<td>0.0</td>
<td>100.0</td>
<td>0.1</td>
</tr>
<tr>
<td>Has Exome/LOF Genotypes</td>
<td>98.0</td>
<td>97.1</td>
<td>0.3</td>
<td>100.0</td>
</tr>
</tbody>
</table>
coin - Analysis of coincidences

Variables:
- Gender
- vars, nead
- (Omni, Adom, Affi, ExomeLOD)

Legend:
- ticks/boxes

Outputs:
- Frequencies
- Column percentages
- Row percentages
- Odds ratios

Residuals and significances:
- Residuals
- Standardized residuals
- Normalized
- Significance or normalized residuals

Distance matrix: haberman
- Adjacency matrix
- Similarity matrix
- Centrality measures
- Bonferroni-adjusted

p <= .50
Support 0
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></th>
<th>female</th>
<th>1760</th>
<th>1740</th>
</tr>
</thead>
<tbody>
<tr>
<td>male</td>
<td>0</td>
<td>1070</td>
<td>1028</td>
</tr>
<tr>
<td>Has Omni Genotypes</td>
<td>488</td>
<td>489</td>
<td>880</td>
</tr>
<tr>
<td>Has Axiom Genotypes</td>
<td>581</td>
<td>614</td>
<td>3</td>
</tr>
<tr>
<td>Has Affy 6.0 Genotypes</td>
<td>1062</td>
<td>1001</td>
<td>2055</td>
</tr>
<tr>
<td>Has Exome/LOF Genotypes</td>
<td>881</td>
<td>3</td>
<td>2063</td>
</tr>
</tbody>
</table>

Col. percentages

<table>
<thead>
<tr>
<th></th>
<th>female</th>
<th>100.0</th>
<th>0.0</th>
<th>51.0</th>
<th>49.9</th>
<th>48.6</th>
<th>51.5</th>
</tr>
</thead>
<tbody>
<tr>
<td>male</td>
<td>0.0</td>
<td>100.0</td>
<td>49.0</td>
<td>50.1</td>
<td>51.4</td>
<td>48.5</td>
<td></td>
</tr>
<tr>
<td>Has Omni Genotypes</td>
<td>60.3</td>
<td>59.1</td>
<td>100.0</td>
<td>90.1</td>
<td>0.3</td>
<td>99.6</td>
<td></td>
</tr>
<tr>
<td>Has Axiom Genotypes</td>
<td>27.7</td>
<td>28.1</td>
<td>41.9</td>
<td>100.0</td>
<td>0.0</td>
<td>41.3</td>
<td></td>
</tr>
<tr>
<td>Has Affy 6.0 Genotypes</td>
<td>33.0</td>
<td>35.3</td>
<td>0.1</td>
<td>0.0</td>
<td>100.0</td>
<td>0.1</td>
<td></td>
</tr>
<tr>
<td>Has Exome/LOF Genotypes</td>
<td>60.3</td>
<td>57.5</td>
<td>98.0</td>
<td>87.1</td>
<td>0.3</td>
<td>100.0</td>
<td></td>
</tr>
</tbody>
</table>
Variables: Continents

Legend: [Selected]

Outputs
- Frequencies [selected]
- Column percentages [selected]
- Row percentages
- Odds ratios

Residuals and significances
- Residuals
- Standardized residuals
- Normalized
- Significance or normalized residuals

Distance matrix: haberman
Adjacency matrix
Similarity matrix
Centrality measures
Bonferroni-adjusted

p = 0.50
Support = 0

OK  Cancel
Web interface for nomolog

A general-purpose nomogram generator for predictive logistic regression models

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

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

Main

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)

Simplify interactions (default: yes)  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

OK  Cancel
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
(llicense)