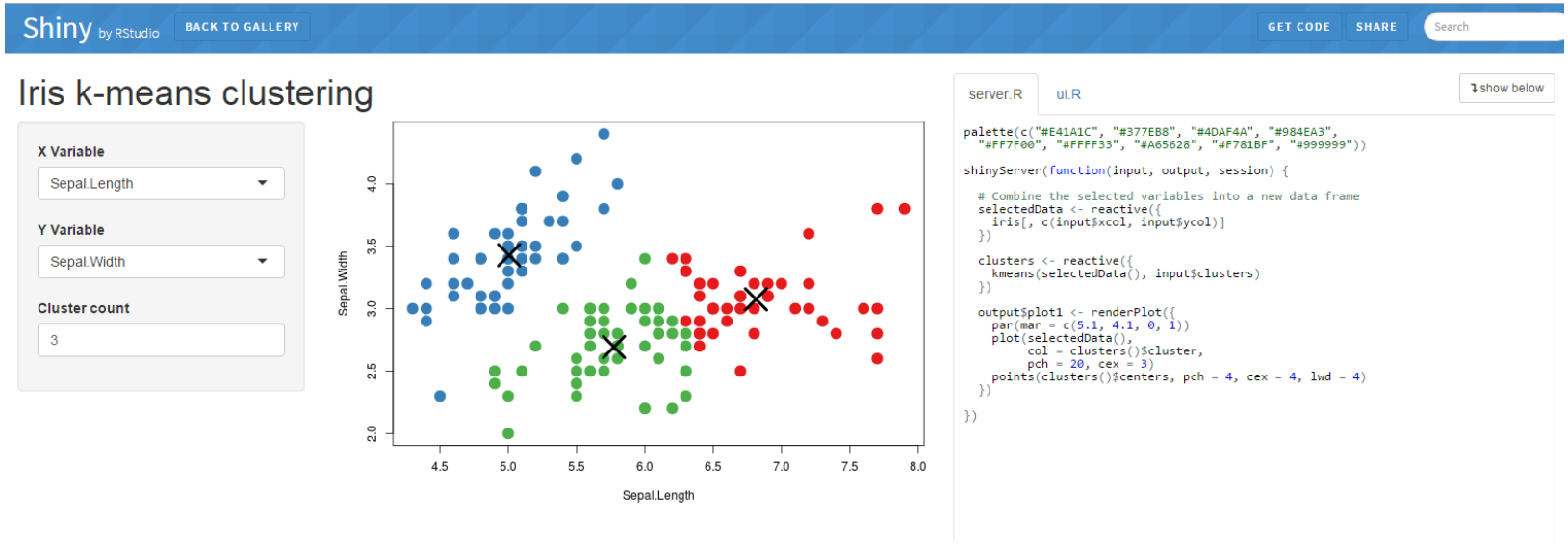
Elementos personalizados en Shiny

Borja Varela

¿Qué es Shiny?

- Paquete para crear interfaces gráficas HTML
- Desarrollado por Rstudio en 2012



Motivaciones

- Permitir una interacción dinámica
- Facilitar el uso de nuestros paquetes
- Ofrecer nuestros paquetes online

Características

- Programación Reactiva
- HTML5/CSS3 y Javascript + node.js y R
- Basado en tecnología websocket
- Disponible en CRAN

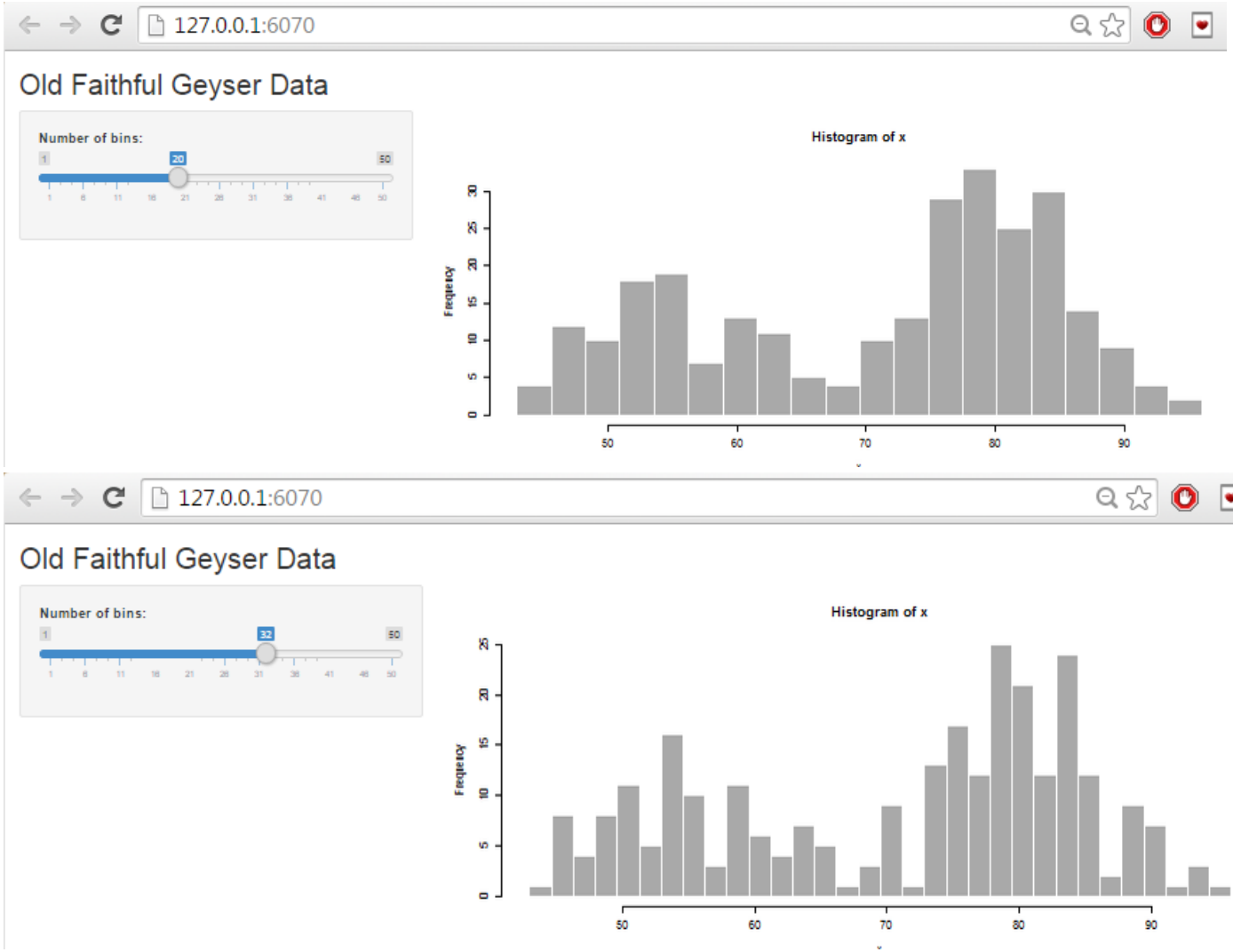
Hello Shiny (ui.R)

```
1
2  # This is the user-interface definition of a Shiny web application.
3  # You can find out more about building applications with Shiny here:
4  #
5  # http://shiny.rstudio.com
6  #
7
8  library(shiny)
9
10 shinyUI(fluidPage(
11
12   # Application title
13   titlePanel("Old Faithful Geyser Data"),
14
15   # Sidebar with a slider input for number of bins
16   sidebarLayout(
17     sidebarPanel(
18       sliderInput("bins",
19                   "Number of bins:",
20                   min = 1,
21                   max = 50,
22                   value = 30)
23     ),
24
25     # Show a plot of the generated distribution
26     mainPanel(
27       plotOutput("distPlot")
28     )
29   )
30 ))
31
```

Hello Shiny (server.R)

```r

# This is the server logic for a Shiny web application.
# You can find out more about building applications with Shiny here:
#
# http://shiny.rstudio.com
#

library(shiny)

shinyServer(function(input, output) {

  output$distPlot <- renderPlot({

    # generate bins based on input$bins from ui.R
    x    <- faithful[, 2]
    bins <- seq(min(x), max(x), length.out = input$bins + 1)

    # draw the histogram with the specified number of bins
    hist(x, breaks = bins, col = 'darkgray', border = 'white')

  })

})

```

Hello Shiny


127.0.0.1:6070
Old Faithful Geyser Data
Number of bins:
1
20
50
Histogram of x
Frequency
0
5
10
15
20
25
30
50
60
70
80
90
127.0.0.1:6070
Old Faithful Geyser Data
Number of bins:
1
32
50
Histogram of x
Frequency
0
5
10
15
20
25
50
60
70
80
90

Restricciones

- Shiny implementa los principales INPUTs en HTML (Text, RadioButtons, Sliders, SelectButtons, etc…).
- Pero a veces nuestros paquetes reciben o muestran datos que no son fácilmente representables con dichos elementos.
- La definición de elementos de entrada/salida es extensible por medio de HTML, CSS , Javascript y R.

Crear un nuevo input

1. Crear un archivo .js
2. Extender el objeto *Shiny.InputBinding()*
3. Sobrescribir los métodos
   - find
   - getId
   - getValue
   - subscribe
   - receiveMessage
   - initialize
4. Registrar el input mediante la función *Shiny.inputBindings.register()*

Checkbox Dinamico (1 /5)

```
var checkboxVectorDistrInputBinding =
        new Shiny.InputBinding();

$.extend(checkboxVectorDistrInputBinding, {
…
});
```

Checkbox Dinamico (2/5)

```
find: function(scope) {
    return $(scope).find('.shiny-checkboxvdistr-input');
  }

getId: function(el) {
    return Shiny.InputBinding.prototype.getId.call(this, el)
              || el.name;
 }
initialize: function(el) {
    this.distributions = new Array();
  }
```

Checkbox Dinamico (3/5)

```
getValue: function(el) {
    var id = $(el).attr("id");
    var res = new Array();
    $("#"+id).find(":checked").each(function(i, v) {
      res.push($(v).attr("value"));
    });
    return(res);
 },
```

Checkbox Dinamico (4/5)

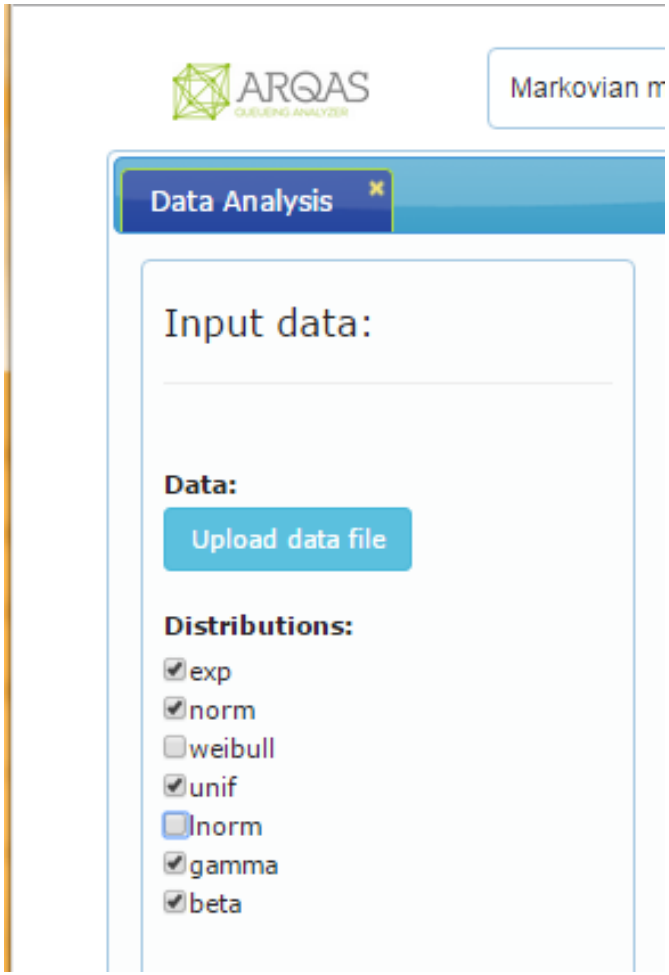
```
subscribe: function(el, callback) {
      var id = $(el).attr("id");
      $("#"+id).on("click", function(event) {
                        callback(true);
                  });
  }
```

Checkbox Dinamico (5/5)

```
receiveMessage: function(el, data) {
      var id = $(el).attr("id");
      if (data.hasOwnProperty('distributions')) {
        this.distributions[id] = data.distributions;
          for (var i=0; i<data.distributions.length; i++) {
                    $("#"+id).append(
                              "<input type='checkbox'
                                        value='" + data.distributions[i] + "'
                               checked>" +
                                data.distributions[i] +
                              "</input><br>");
                    }
                    $("#"+id).trigger('click');
          }
    }
```

Checkbox Dinamico

Cada vez que cambian los valores de los checkbox, R recibe un array con los valores que están marcados



Crear un nuevo output

1. Crear un archivo .js
2. Extender el objeto *Shiny.OutputBinding()*
3. Sobrescribir los métodos
   - find
   - renderValue
   - renderError
4. Registrar el output mediante la función *Shiny.outputBindings.register()*

Visualización de grafos (1/3)

```
var networkOutputBinding = new Shiny.OutputBinding();
$.extend(networkOutputBinding, {
…
}
Shiny.outputBindings.register(networkOutputBinding,
   'shiny.networkOutput');
```

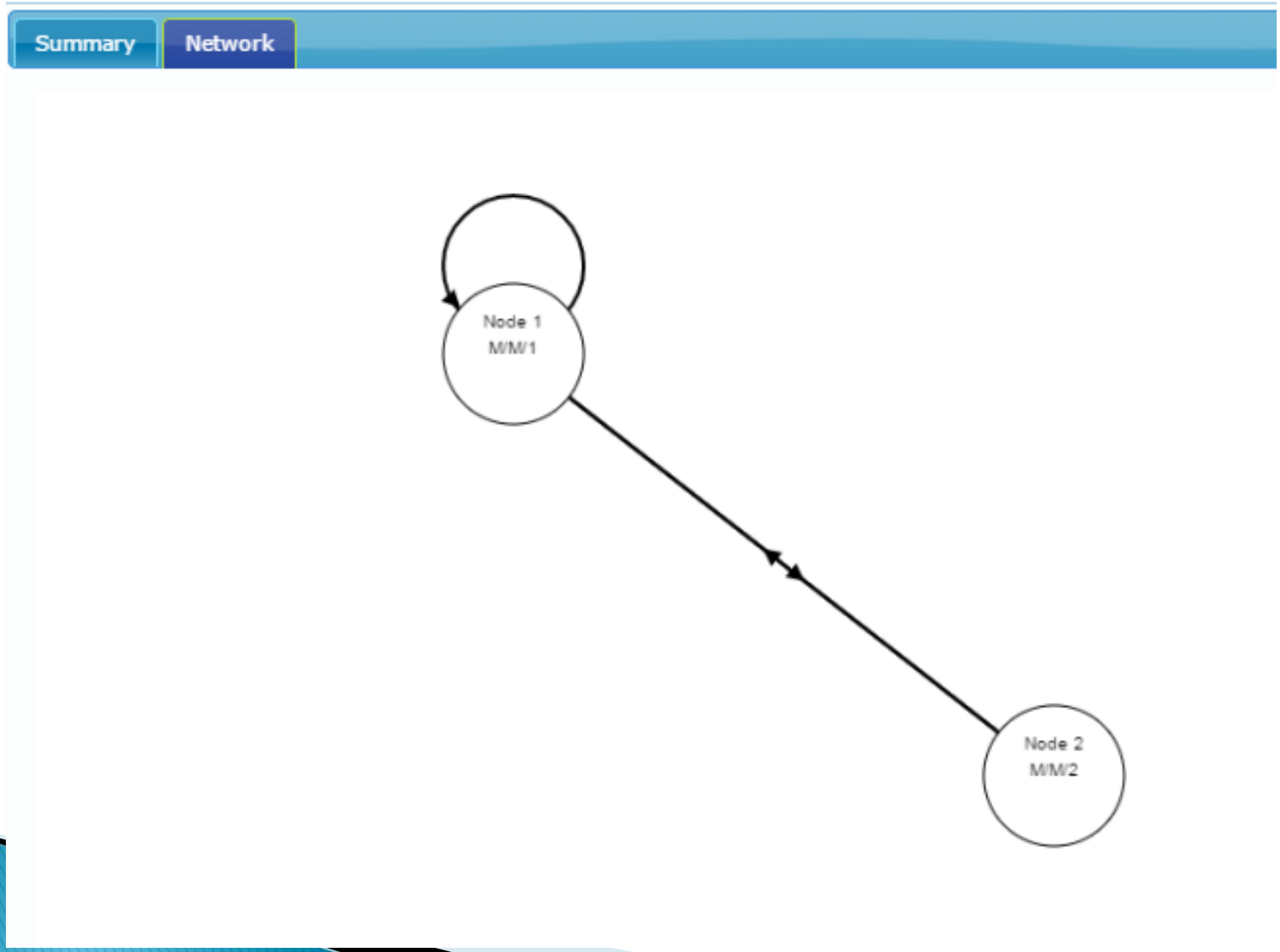
Visualización de grafos (2/3)

```
find: function(scope) {
  return $(scope).find('.shiny-network-output');
}
renderError : function(el, err) {
    $(el).empty();
    $(el).append(
      "<p class='shiny-output-error'>"+err.message+"</p>");
 },
```

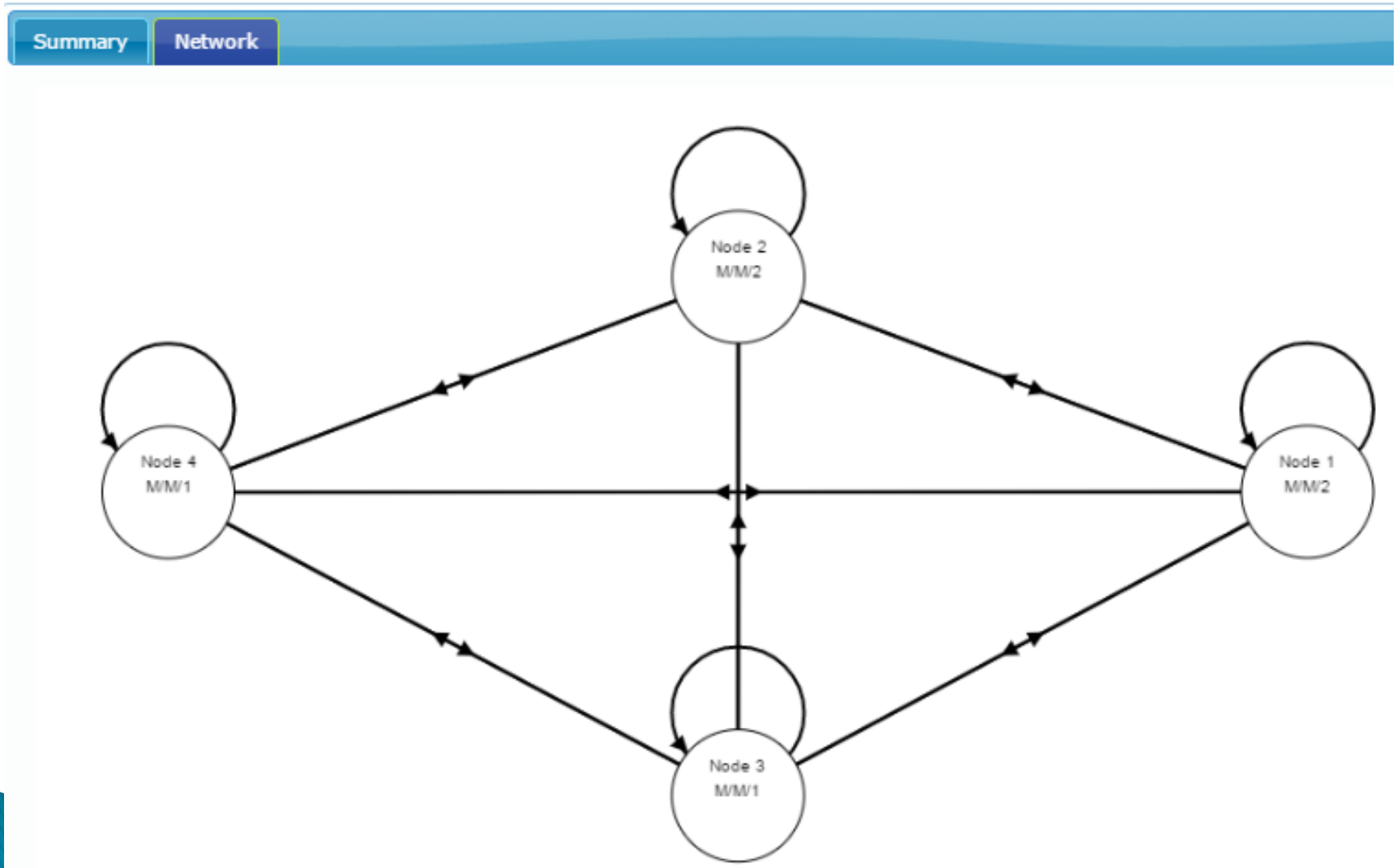
Visualización de grafos (3/3)

```
renderValue: function(el, data) {
        var id = $(el).attr("id");
        if(!this.hasOwnProperty("particleSystem"))
                this.particleSystem = new Array();
        if (data != null) {
            $(el).empty();
            $(el).append("<canvas id='" + id + "-graphcanvas' width=850 height=540></canvas>");
            this.particleSystem[id] = arbor.ParticleSystem(1000, 50, 0.5, true);
            this.particleSystem[id].renderer = new this._Renderer("#" + id + "-graphcanvas");

            for(i=0;  i<data.s.length; i++)
                        this.particleSystem[id].addNode(i+1, {s: data.s[i], nodeText: "M/M/"});
            for(i=0; i<data.s.length; i++) {
                for(j=0; j<data.s.length; j++) {
                    if(data.p[i][j] != 0)
                        this.particleSystem[id].addEdge((i+1), (j+1), {p: parseFloat(data.p[i][j])});
                }
            }}}
```

Visualización de grafos



Visualización de grafos



Conclusiones

- Existen numerosas librerías en javascript que podemos añadir a nuestros paquetes en R
- Gracias a que se pueden añadir a nuestras interfaces HTML podemos mejorar la interacción con los usuarios.