

# Spring School on Integrated Operational Problems

May 14-16, 2018, Troyes, France

## PLAN

1rst session

Introduction (15-30 minutes)

C++ (1 hour)

The Clarke and Wright algorithm

Modifying the CAW and testing it

Adding your own strategy to the CAW

How to debug ?

2nd session

JSON (30 minutes)

Understanding the role of the JSON files

Adding two new algorithms to the CAW

PHP (30 minutes)

Adding a new problem to the web-site

Conclusion will be yours !



# Spring School on Integrated Operational Problems

May 14-16, 2018, Troyes, France

PHP or  
*"How to convert web data to useful data ?"*



# PHP customization

**But ... what else can we do ?**

Look, up to now, we only  
added a problem (WAC)  
and added two algorithms (WAC1 and WAC2)

And they all shared the same instances composed of  
a depot  
a list of customers with quantities of pizzas to deliver to them  
the capacity of the only delivery vehicle

The answer is  
**« we could add problems and algorithms with a different instance type ».**

# PHP customization

Data's instance

value of the courrant instance:

Merges(integer)

Capacity(integer)

Depot(point)

1

Lon Lat

clients(list)

load GPS points from a file:

current selection

Lon	Lat	Goods
2.3465	48.850	0
2.3355	48.856	1
2.3430	48.861	1
2.3535	48.861	1
2.3594	48.856	1
2.3640	48.851	1
2.3603	48.845	1
2.3484	48.842	1
2.3406	48.841	1
2.3317	48.845	1
2.3271	48.850	1

For the sake of clarity and efficiency we'll use again WAC to illustrate this.

But, you easily understand that you'll be able later to apply the following to any new type of problem and instance.

Imagine we want to customize the web-service for a real pizza shop. We may need to use a more adapted vocabulary.

Goods                   =>       Pizzas  
Capacity               =>       Giorgio's van capacity

And many other nice ideas (it is unlimited ...)

# PHP customization

There is a file describing the instance.  
This is the `WAC\instance.json` file,  
it contains a JSON array.

Data's instance

value of the courrant instance:

Merges(integer)

Capacity(integer)

Depot(point)

1

Lon	Lat
<input type="text" value="2.3465"/>	<input type="text" value="48.850"/>

clients(list)

load GPS points from a file:

current selection

Lon	Lat	Goods
<input type="text" value="2.3465"/>	<input type="text" value="48.850"/>	<input type="text" value="0"/>
<input type="text" value="2.3355"/>	<input type="text" value="48.856"/>	<input type="text" value="1"/>
<input type="text" value="2.3430"/>	<input type="text" value="48.861"/>	<input type="text" value="1"/>
<input type="text" value="2.3535"/>	<input type="text" value="48.861"/>	<input type="text" value="1"/>
<input type="text" value="2.3594"/>	<input type="text" value="48.856"/>	<input type="text" value="1"/>
<input type="text" value="2.3640"/>	<input type="text" value="48.851"/>	<input type="text" value="1"/>
<input type="text" value="2.3603"/>	<input type="text" value="48.845"/>	<input type="text" value="1"/>
<input type="text" value="2.3484"/>	<input type="text" value="48.842"/>	<input type="text" value="1"/>
<input type="text" value="2.3406"/>	<input type="text" value="48.841"/>	<input type="text" value="1"/>
<input type="text" value="2.3317"/>	<input type="text" value="48.845"/>	<input type="text" value="1"/>
<input type="text" value="2.3271"/>	<input type="text" value="48.850"/>	<input type="text" value="1"/>

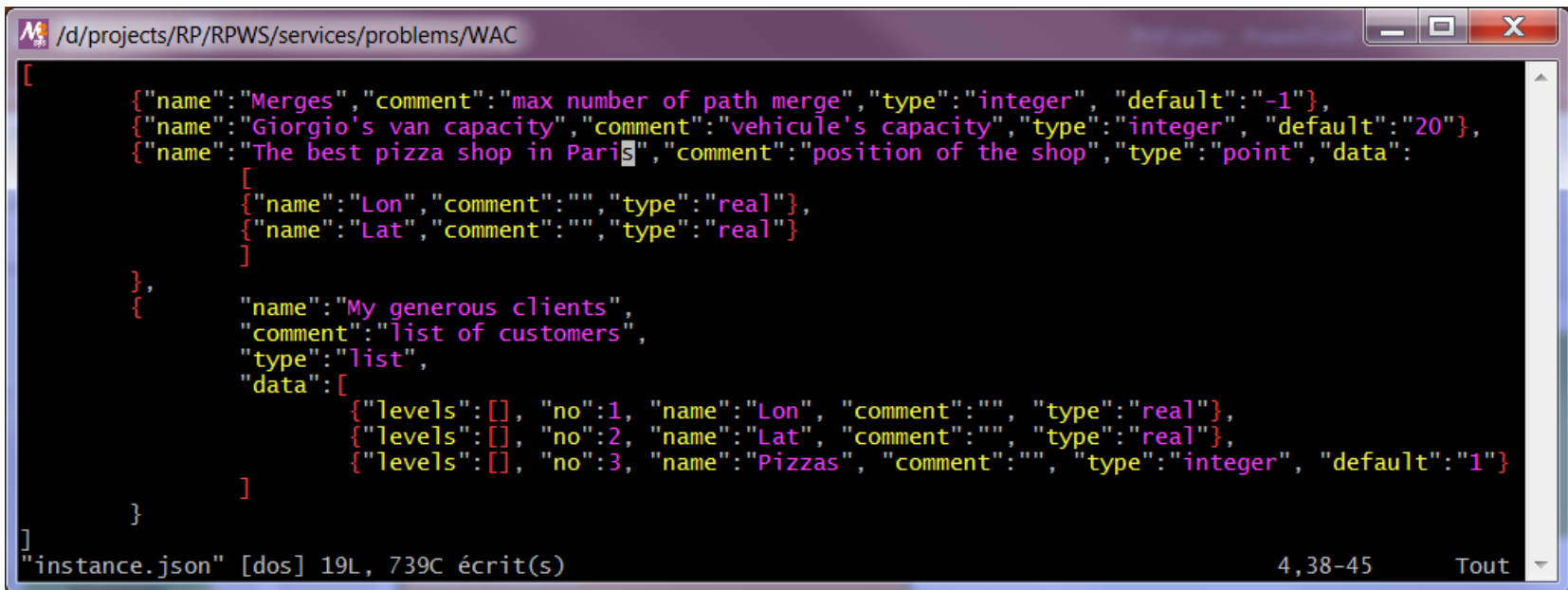
```
file:///d:/projects/RP/RPWS/services/problems/WAC
[
  {
    "name": "Merges", "comment": "max number of path merge", "type": "integer", "default": "-1"},
    {
    "name": "Capacity", "comment": "vehicule' capacity", "type": "integer", "default": "100"},
    {
    "name": "Depot", "comment": "position of the depot", "type": "point", "data":
      [
        {
        "name": "Lon", "comment": "", "type": "real"},
        {
        "name": "Lat", "comment": "", "type": "real"}
      ]
    },
    {
    "name": "clients",
    comment: "list of customers",
    "type": "list",
    "data": [
      {
      "levels": [], "no": 1, "name": "Lon", "comment": "", "type": "real"},
      {
      "levels": [], "no": 2, "name": "Lat", "comment": "", "type": "real"},
      {
      "levels": [], "no": 3, "name": "Goods", "comment": "", "type": "integer"}
    ]
  }
]
```

Each element of the array describes an item  
of the instance. Here we have four items.

- Merges is an integer
- Capacity is an integer
- Depot is a point ID (its position)
- Clients is a list of GPS points

# PHP customization

To apply our wishes, we only have to slightly modify this file, like this:



```
[
  {
    "name": "Merges", "comment": "max number of path merge", "type": "integer", "default": "-1"},
    {
      "name": "Giorgio's van capacity", "comment": "vehicule's capacity", "type": "integer", "default": "20"},
      {
        "name": "The best pizza shop in Paris", "comment": "position of the shop", "type": "point", "data":
          [
            {
              "name": "Lon", "comment": "", "type": "real"
            },
            {
              "name": "Lat", "comment": "", "type": "real"
            }
          ]
      },
      {
        "name": "My generous clients",
        "comment": "list of customers",
        "type": "list",
        "data": [
          {
            "levels": [], "no": 1, "name": "Lon", "comment": "", "type": "real"
          },
          {
            "levels": [], "no": 2, "name": "Lat", "comment": "", "type": "real"
          },
          {
            "levels": [], "no": 3, "name": "Pizzas", "comment": "", "type": "integer", "default": "1"
          }
        ]
      }
    ]
  }
]
"instance.json" [dos] 19L, 739C écrit(s) 4,38-45 Tout
```

Goods

=> Pizzas, with a default value of 1

Capacity

=> Giorgio's van capacity, with a default capacity of 20 pizzas

Clients

=> My generous clients

Depot

=> The best pizza shop in Paris

# PHP customization

Data's instance

value of the courrant instance:

Merges(integer)

Giorgio's van capacity(integer)

The best pizza shop in Paris(point)

1

Lon	Lat
2.3481:	48.850:

My generous clients(list)

load GPS points from a file:

current selection

Lon	Lat	Pizzas
2.3481:	48.850:	0
2.3477:	48.844:	1
2.3349:	48.850:	1
2.3365:	48.863:	1
2.3539:	48.862:	1
2.3629:	48.857:	1

Here is the new aspect of the instance.

Nicer isn't it ?

But that's only makeup,  
let's look at a more serious question !

*Imagine the shop bakes pizzas and burgers.  
Shouldn't we add a new item to the customers ?  
Could we call it « burgers » ?*

# PHP customization

OK, let's go for a new item called « burgers ».

```
/d/projects/RP/RPWS/services/problems/WAC
[
  {"name": "Merges", "comment": "max number of path merge", "type": "integer", "default": "-1"},
  {"name": "Giorgio's van capacity", "comment": "vehicule's capacity", "type": "integer", "default": "20"},
  {"name": "The best pizza shop in Paris", "comment": "position of the shop", "type": "point", "data":
    [
      {"name": "Lon", "comment": "", "type": "real"},
      {"name": "Lat", "comment": "", "type": "real"}
    ]
  },
  {"name": "My generous clients",
   "comment": "list of customers",
   "type": "list",
   "data": [
     {"levels": [], "no": 1, "name": "Lon", "comment": "", "type": "real"},
     {"levels": [], "no": 2, "name": "Lat", "comment": "", "type": "real"},
     {"levels": [], "no": 3, "name": "Pizzas", "comment": "", "type": "integer", "default": "1"},
     {"levels": [], "no": 4, "name": "Burgers", "comment": "", "type": "integer", "default": "1"}
   ]
  }
]
"instance.json" [dos] 20L, 831C écrit(s) 13,10-24 Haut
```

Don't forget  
the column  
here !



# PHP customization

Data's instance

value of the courrant instance:

Merges(integer)

Giorgio's van capacity(integer)

The best pizza shop in Paris(point)

Lon                  Lat

My generous clients(list)

load GPS points from a file:

current selection ▾

Lon	Lat	Pizzas	Burgers
2.3481	48.850	1	1
2.3477	48.844	1	1
2.3349	48.850	1	1
2.3365	48.863	1	1
2.3539	48.862	1	1
2.3629	48.857	1	1

Here is the new aspect of the instance.

Nicer again isn't it ?

But the real hard question is now

« **How shall we process this new item ?** »

The answer is « **with PHP** ».



# PHP customization

Like for JSON, we cannot teach you PHP in 20 minutes.

So we'll only show you very simple modifications to the current PHP files of WAC.

We suggest you to learn PHP if you want to add your own problems into any web site offering some web-services like ours.

There are thousands of good docs about PHP on the web.



# PHP customization

Each web-service is implemented at three software levels

1/ The HTTP server (or network communication level)

A « PHP-call » is received by the HTTP server with the JSON data  
It means a PHP file, for the selected (problem, algorithm) will be executed

2/ The PHP code (or data translation level)

converts the received JSON data into executable-compatible data  
calls the executable  
converts back the results into JSON data for the web-browser

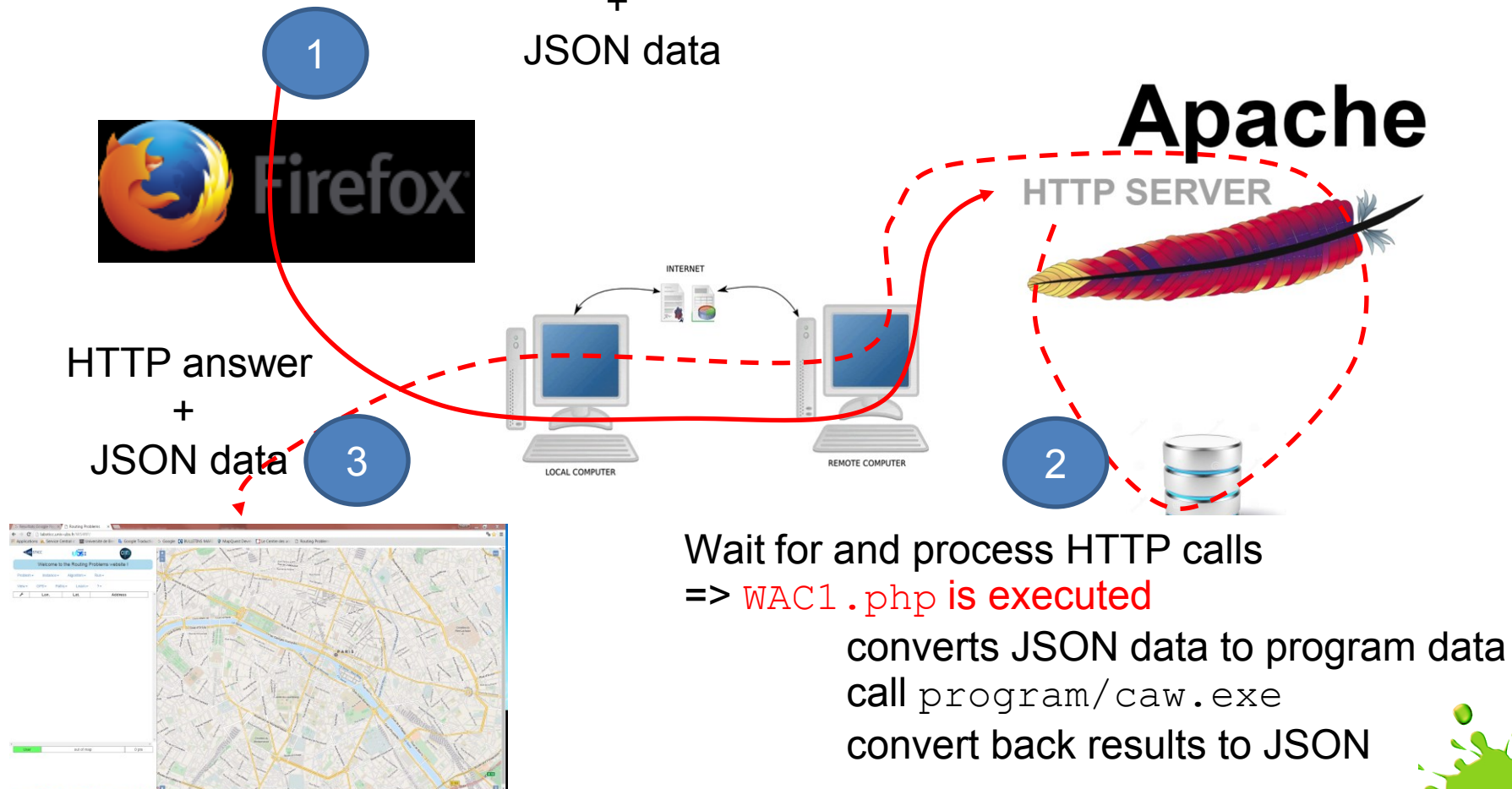
3/ The executable program (or implementation level)

The service is a program, written with any programming language  
and executed locally on the machine running the HTTP server.  
It is fed with data from the web-browser and produces data back for it.

# PHP customization

HTTP request for localhost://services/problems/WAC/WAC1.php

+  
JSON data



# PHP customization

A short example is better than a long story to explain how to decode JSON data.

The image shows two Notepad++ windows. The top window, titled 'WAC\instance.json', contains a JSON array of objects. The bottom window, titled 'WAC\WAC1.php', contains PHP code that decodes the JSON data and assigns values to variables. Blue arrows point from the JSON data in the top window to the corresponding PHP code in the bottom window.

```
WAC\instance.json
[
  {
    "name": "Merges", "comment": "max number of path merge", "type": "integer", "default": "-1"},
    {
    "name": "Giorgio's van capacity", "comment": "vehicule's capacity", "type": "integer", "default": "20"},
    {
    "name": "The best pizza shop in Paris", "comment": "position of the shop", "type": "point", "data":
      {
        "name": "Lon", "comment": "", "type": "real"},
        {
        "name": "Lat", "comment": ""
      }
    },
    {
    "name": "My generous client", "comment": "list of customer", "type": "list", "data": [
      { "levels": [], "name": "level 1" },
      { "levels": [], "name": "level 2" },
      { "levels": [], "name": "level 3" },
      { "levels": [], "name": "level 4" }
    ]
  }
]

WAC\WAC1.php
<?php
ini_set('display_errors', 'on');
error_reporting(E_ALL | E_STRICT);
// if there is an error go to catch
try
{
    if (isset($_REQUEST['json']))
    {
        // DATA from the json post request variable
        $json_data = json_decode($_REQUEST['json']);
        // json_data = {
        //     "instance": WS.instance,
        //     "instance[0]": merges,
        //     "instance[1]": capacity,
        //     "instance[2]": depot,
        //     "instance[3]": points
        //     "matrix": GUI.matrix,
        //     "parameters": WS.parameters,
        //     "runPos": run.pos
        // }
        //
        $merge_index = 0;
        $capacity_index = 1;
        $depot_index = 2;
        $points_index = 3;
        // uniqid() is a php function to obtain a name
    }
}
catch (Exception $e)
{
    echo $e->getMessage();
}

```

WAC\instance.json

WAC\WAC1.php

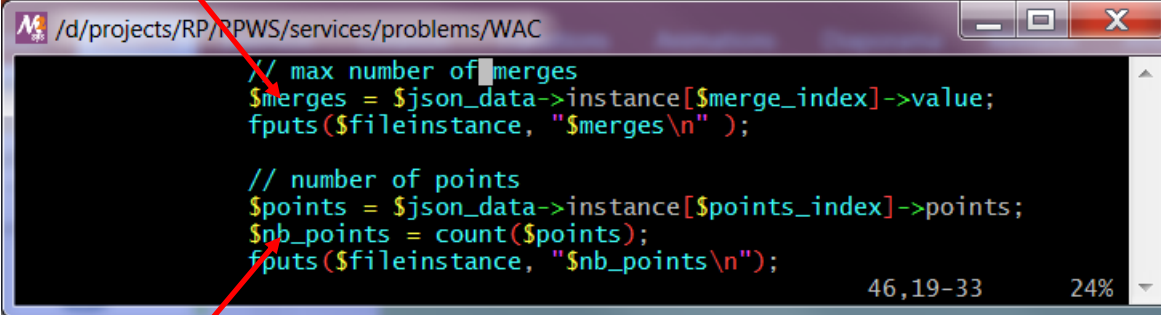
Automatically added  
distance matrix  
parameters  
run position

25,1-8 Haut

# PHP customization

Here is how to get specific values from the instance.

Obtain the value of « merges », which is an integer.



```
/d/projects/RP/PPWS/services/problems/WAC
// max number of merges
$merges = $json_data->instance[$merge_index]->value;
fputs($fileinstance, "$merges\n" );

// number of points
$points = $json_data->instance[$points_index]->points;
$nb_points = count($points);
fputs($fileinstance, "$nb_points\n");
```

The screenshot shows a terminal window with a dark background and light-colored text. A red arrow points from the text above to the line `$merges = $json_data->instance[$merge_index]->value;` in the code. The terminal title bar shows the path `/d/projects/RP/PPWS/services/problems/WAC`. The bottom right of the terminal shows `46,19-33` and `24%`.

Obtain the value of « clients », which is a list of GPS points + quantities.

# PHP customization

Here is how to get specific values from the instance.

```
WAC\instance.json

{
  "name": "My generous clients",
  "comment": "list of customers",
  "type": "list",
  "data": [
    {"levels": [], "no": 1, "name": "Lon", "comment": "", "type": "real"},
    {"levels": [], "no": 2, "name": "Lat", "comment": "", "type": "real"},
    {"levels": [], "no": 3, "name": "Pizzas", "comment": "", "type": "integer", "default": "1"},
    {"levels": [], "no": 4, "name": "Burgers", "comment": "", "type": "integer", "default": "1"}
  ]
}
```

```
WAC\WAC1.php

$pizzas = array();
$burgers = array();
for ($i = 0; $i < $nb_points; $i++) {
  $lon = $points[$i]->Lon;
  $lat = $points[$i]->Lat;
  $pizzas[$i] = $points[$i]->Pizzas;
  $burgers[$i] = $points[$i]->Burgers;
}
```

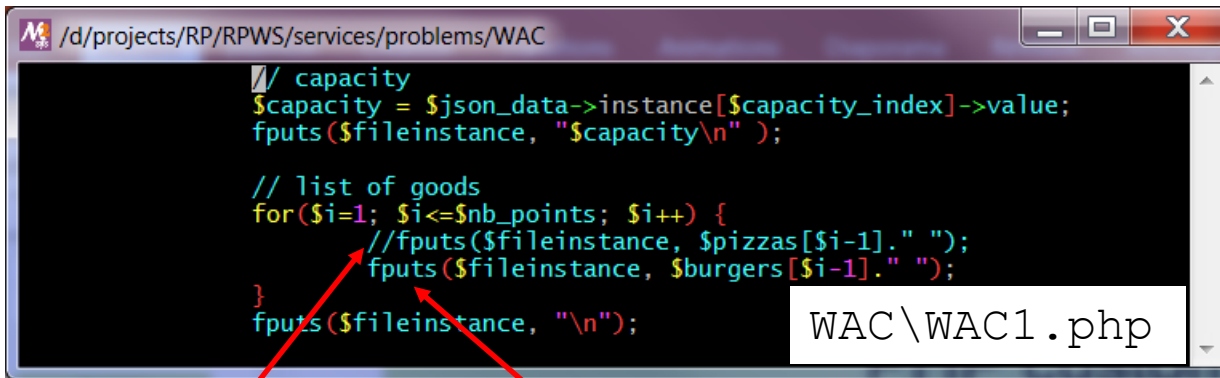
Obtain the pizzas to deliver.

Obtain the burgers to deliver.



# PHP customization

Here is how to choose between pizzas or burgers to deliver.



```
WAC\WAC1.php
// capacity
$capacity = $json_data->instance[$capacity_index]->value;
fputs($fileinstance, "$capacity\n" );

// list of goods
for($i=1; $i<=$nb_points; $i++) {
    //fputs($fileinstance, $pizzas[$i-1]." ");
    fputs($fileinstance, $burgers[$i-1]." ");
}
fputs($fileinstance, "\n");
```

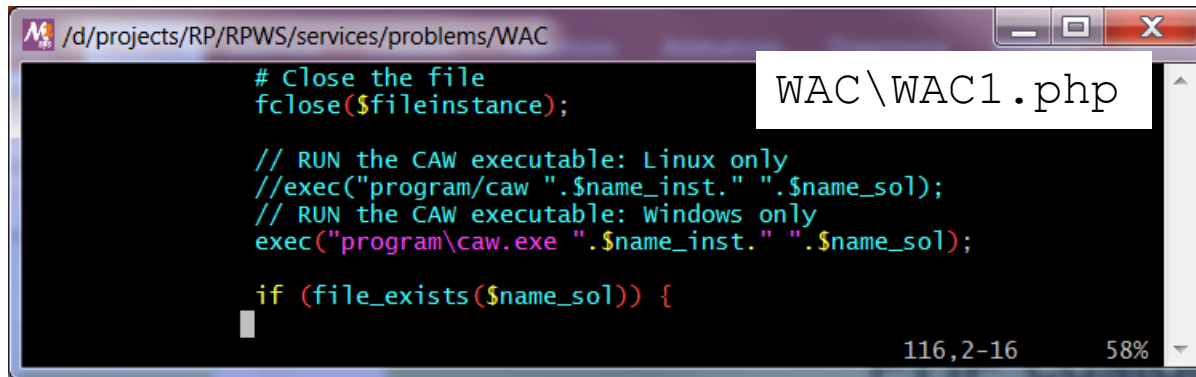
To apply CAW to the delivery of burgers keep that line.

To apply CAW to the delivery of pizzas uncomment that line and comment out the next.



# PHP customization

Here is how PHP calls the executable.  
Be careful, exact syntax depends on the OS !



```
/d/projects/RP/RPWS/services/problems/WAC  
# Close the file  
fclose($fileinstance);  
  
// RUN the CAW executable: Linux only  
//exec("program/caw ".$name_inst." ".$name_sol);  
// RUN the CAW executable: Windows only  
exec("program\caw.exe ".$name_inst." ".$name_sol);  
  
if (file_exists($name_sol)) {  
      
}
```

WAC\WAC1.php

116,2-16 58%

# PHP customization

Now you have several possibilities

- 1/ you read `WAC\WAC1.php` to better understand PHP  
and we can talk about it
- 2/ you modify `WAC\WAC1.php` to match  
what has been described in these slides  
to make it work properly on your PC
- 3/ you take some rest ...

(I suggest the second one 😊 )



# We are finished !!!!

## And the conclusion is YOURS !!!!!



Thank you for your kind attention.

