

Ajax and Ruby on Rails

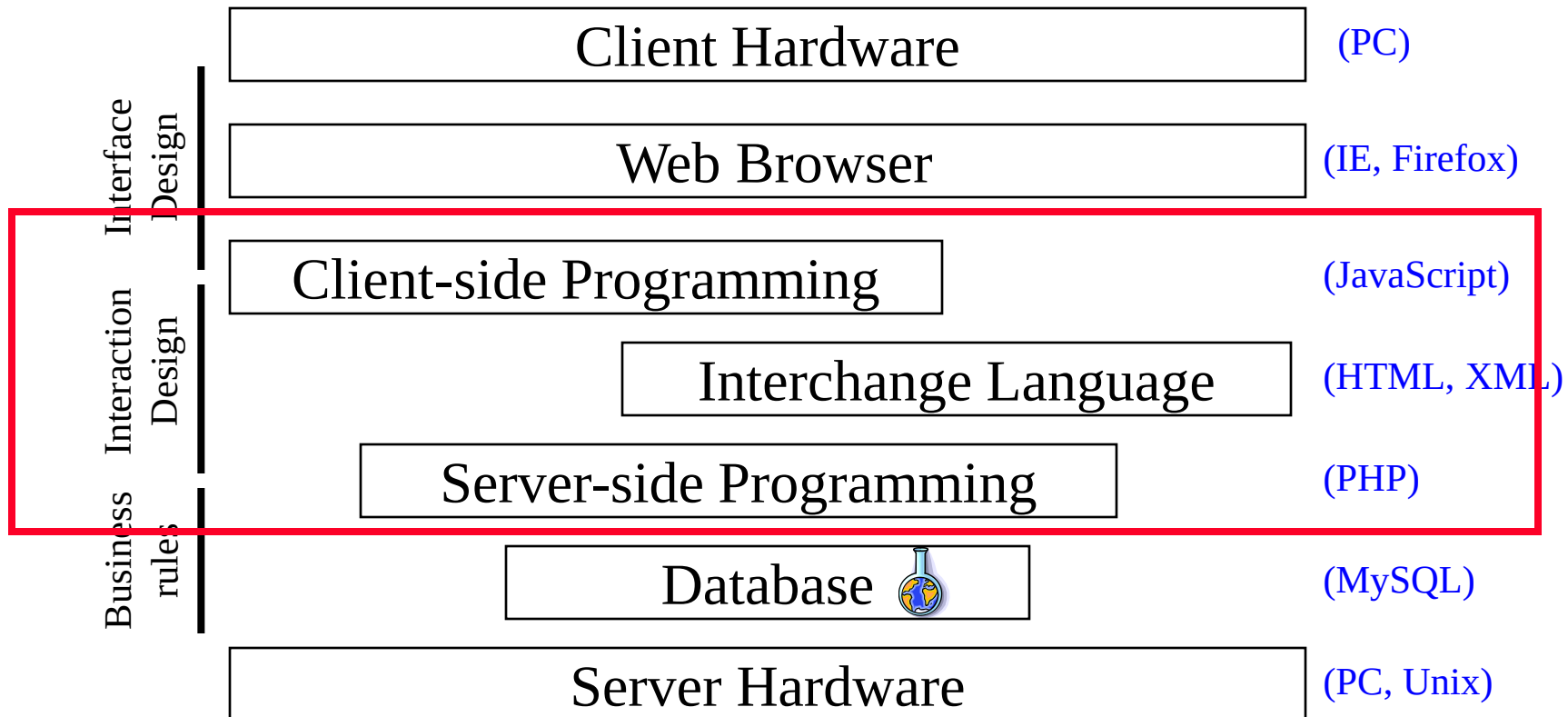
Session 9

INFM 603

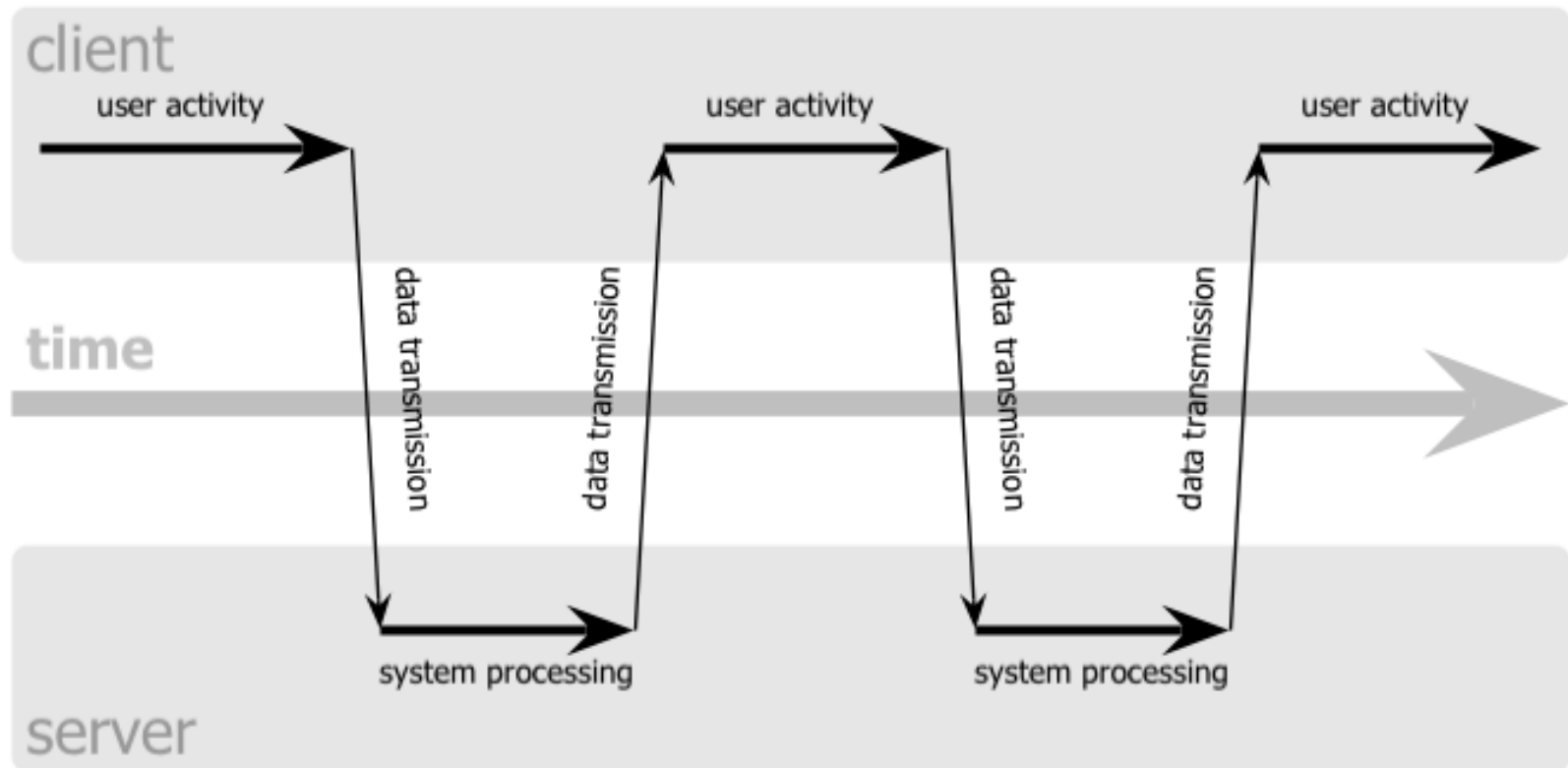
Agenda

- Ajax
- Ruby
- Rails (Model-View-Controller)

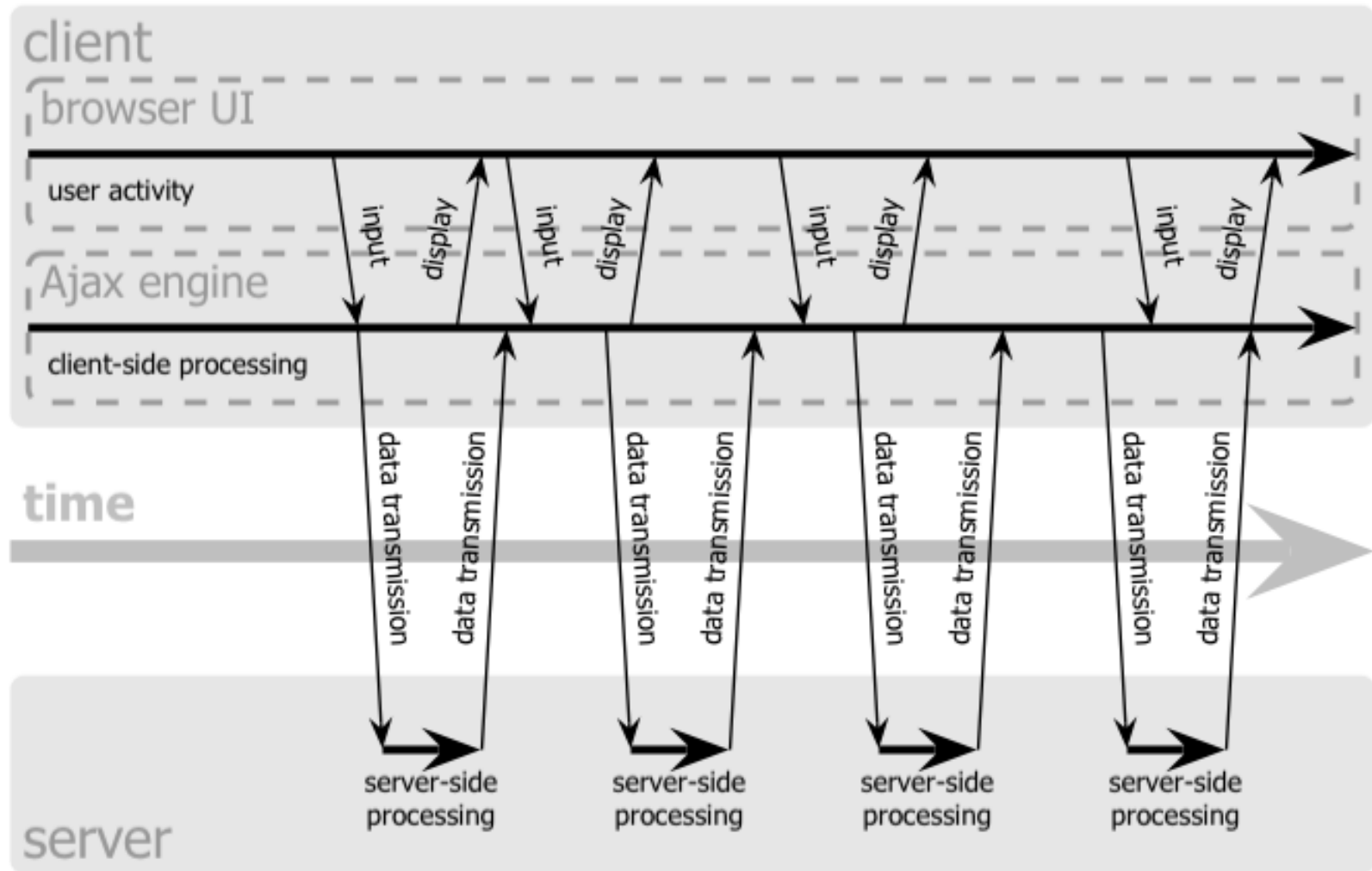
- Relational normalization
- Structured programming
- Software patterns
- Object-oriented design
- Functional decomposition



classic web application model (synchronous)



Ajax web application model (asynchronous)



Sajax: Simple Ajax for PHP

<http://www.modernmethod.com/sajax/>

<?

```
require("Sajax.php");
```

```
function multiply($x, $y) {  
    return $x * $y;  
}
```

```
sajax_init();
```

```
// $sajax_debug_mode = 1;
```

```
sajax_export("multiply");  
sajax_handle_client_request();
```

?>

```
<html><head>
  <title>Multiplier</title>
  <script>
    <?
    sajax_show_javascript();
    ?>
    function do_multiply_cb(z) {
      document.getElementById("z").value = z;
    }
    function do_multiply() {
      // get the folder name
      var x, y;
      x = document.getElementById("x").value;
      y = document.getElementById("y").value;
      x_multiply(x, y, do_multiply_cb);
    }
  </script>
</head>
<body>
  <input type="text" name="x" id="x" value="2" size="3">
  *
  <input type="text" name="y" id="y" value="3" size="3">
  =
  <input type="text" name="z" id="z" value="" size="3">
  <input type="button" name="check" value="Calculate"
    onclick="do_multiply(); return false;">
</body></html>
```

Communication methods

- GET/POST (client to server) – simple data
- XML (the “X” in AJAX)
 - Structured data between client and server.
 - Client responsible for updating page
- JSON – a light-weight alternative to XML
- HTML (server to client) – server sends rendered result
- Widget – GUI library hiding (some) AJAX details (e.g. DoJo)

Ruby

<http://tryruby.org>

- `2+2`
- `puts "Hello world!"`
- `backward = "Hello world".reverse`
- `backward.capitalize`
- `numbers = [12, 3, 10]`
- `numbers.sort!`
- `numbers`

Variables and Operators

- Ruby variables are not specially marked

`size = 12`

- Variables can hold any value type
- Similar operators to PHP, Javascript

`+ / == & &`

String handling

- String concatenation is “+” (like Javascript, **unlike PHP!**):

```
puts "My name is " + name
```

- Strings can also be interpolated:

```
puts "My name is #{ name }"
```

```
puts "Two plus two is #{ 2 + 2 }"
```

- Single quoting turns off interpolation

Objects

- Ruby has objects

```
user = User.new("John", "Doe")
```

- Object method reference is using “.”

```
user.set_given_name("Jane")
```

- No-argument methods don't need ()

```
user.get_given_name
```

Arrays and Hashes

- Ruby has numerically-indexed arrays

```
fruits = ["mango", "orange", "apple" ]  
fruits[1]    # mango
```

- Ruby also has string-indexed hashes

```
food = {"mango" => "fruit", "pear" =>  
"fruit", "onion" => "vegie"}  
food["onion"]    # "vegie"
```

- The “=>” operator links keys with values
(also used outside of hashes)

Symbols

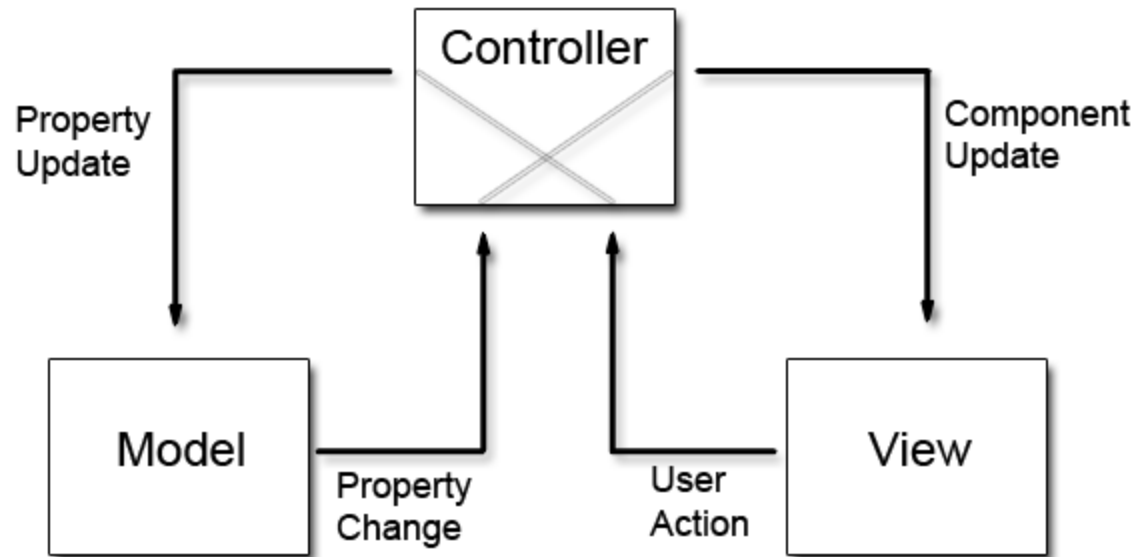
- Ruby has a special type called symbols

`:author => "Jane"`

- A symbols means ... itself!
- Think of them as a kind of keyword

```
create_table :users do |t|  
  t.column :name, :string, :limit=>11  
  t.column :password, :string  
end
```

Model-View-Controller



Simple MVC in WebApps

- Model: database and code to handle basic CRUD (Create, Read, Update, Delete) functions
- View: dynamic web interface, populated with server-side variables
- Controller: code in between: extracts information for views, executes updates from user

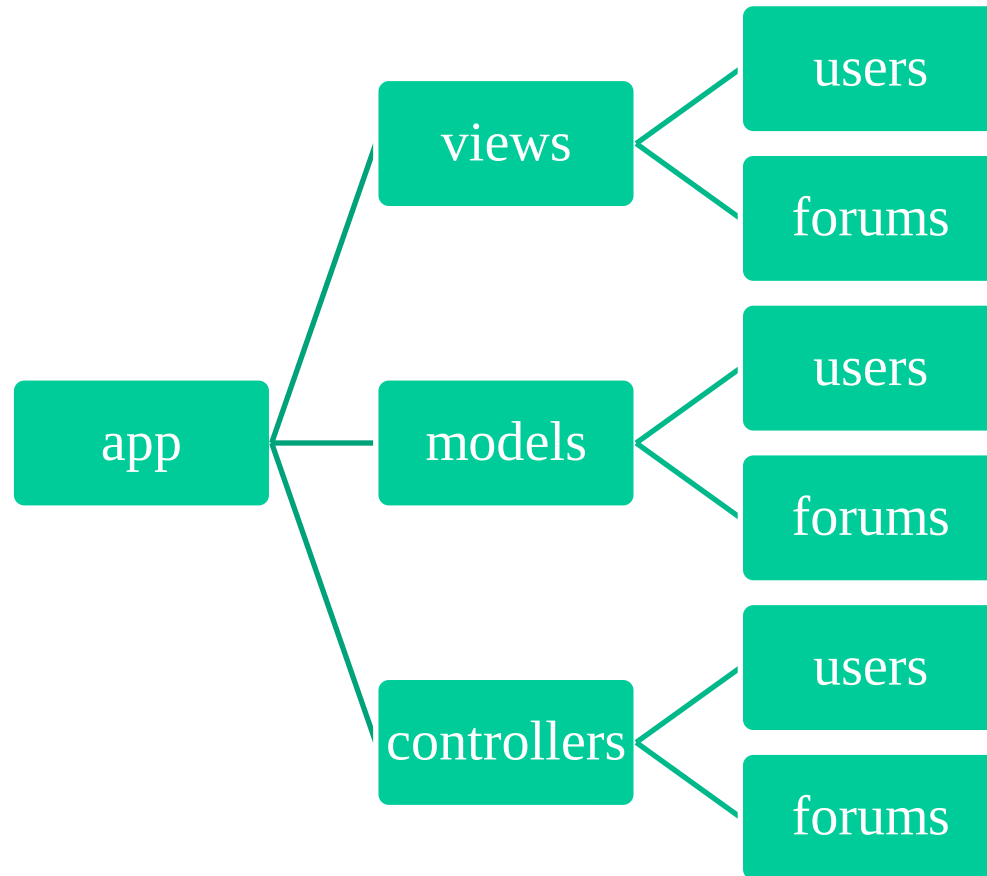
Without MVC separation

```
$result = mysql_query("SELECT given, family FROM User
WHERE login='" . mysql_real_escape_string($_GET["uid"])
. "'");
# NOTE: no error handling!
$fields = mysql_fetch_assoc($result);
$given = $fields["given"];
# Populate other information
...
<form method="post">
    Given name: <input name="given" value="<?php print
$given ?>"> <!-- what about unclean characters? -->
    <input type="submit" name="Update" value="update">
</form>
...
```

Without MVC separation (cont.)

```
# This code would actually go at the _start_ of the
# php code file
if (isset($_POST["Update"])) {
    # Error checking and handling
    mysql_query("UPDATE User SET given = " .
        mysql_real_escape_string($_POST["given"]) . "'");
    # More error checking
    # Possible redirect?
}
```

MVC separation in Rails



Wrapper, scaffold, generation

MVC discipline allows:

- Wrapper code: re-usable code for performing common tasks
- Scaffold code: basic implementations (e.g. of editing a user) that can be replaced when ready
- Generation: auto-generation of template code to be extended

Generating Templates

- `ruby script/generate model employee`
 - `self.up` method creates a database table
 - `self.down` method drops that table
- `ruby script/generate controller employees`
 - Creates generic methods
 - `new`, `index`, `show`, `edit`
 - Creates matching views

Object-relational model (ORM)

- In naïve webapp, data has three lives:
 1. Row in database
 2. Variable in program (field of object)
 3. Parameter in request (HTTP POST)
- ORM wraps database in object-oriented code (table rows look like objects)
- MVC allows near-transparent extension to Web forms, parameters.

Rails and AJAX

- MVC AJAX: a view within a view
- Request minimally rewritten
 - `link_to` becomes `link_to_remote`
 - identify target for result
- Response sends HTML code for updated division within full HTML page