# CREACIÓN INTELIGENTE DE GRUPOS

Arquitecturas Orientadas a Servicios

11 DE ENERO DE 2018

Iván Carrasco Alonso

Rafael Borrego Vidal

Raquel Fernández Peláez

Jonathan Mateos Marcos

# Índice

# Enunciado

Realizar un conjunto de servicios que permita la ordenación de una serie de alumnos/as dado en equipos base para su distribución en grupos cooperativos. Toda la teoría de grupos cooperativos será facilitada por el profesor. Los alumnos se ordenarán en función de una serie de características y un determinado peso en cada característica. Los servicios implementados correrán un algoritmo que será el encargado de implementar la solución. Se valorará la solución aportada a nivel de servicios, a nivel de ESB así como la interfaz creada para el cliente.

# Teoría de Grupos Corporativa

## Tabla de variables

| **Variables** | **1** | **2** | **3** | **4** | **5** |
|---|---|---|---|---|---|
| Académico | Muy Bajo | Bajo | Medio | Bueno | Muy Bueno |
| Social | Asocial | Poco Social | Montón | Sociable | Muy Sociable |
| Es capaz de ayudar | No | Poco | A veces | Si | Siempre |
| Necesita ayuda | No | Poco | A veces | Si | Siempre |
| Organización | Muy Mala | Mala | Media | Buena | Muy Buena |
| Liderazgo | Muy Malo | Malo | Medio | Bueno | Muy Bueno |
| Decisión-Participación | Muy Baja | Baja | Media | Alta | Muy Alta |
| Orden en clase | Muy Malo | Malo | Medio | Bueno | Muy Bueno |

*Tabla 1: Roles y Perfiles*

Donde cada número corresponde con:

1. 20%
2. 40%
3. 60%
4. 80%
5. 100%

| **Sexo** | **Valor** |
|---|---|
| Mujer | 0 |
| Hombre | 1 |

*Tabla 2:Sexo*

| **Alumno** | **No quiero sentarme** |
|---|---|
| Alumno 1 | Alumno 3, Alumno 5 |
| Alumno 2 | |
| Alumno 3 | Alumno2, Alumno 4 |
| …. | ….. |

*Tabla 3:No quiero Sentarme*

## Perfil

Hay tres tipos de perfiles:

- **Perfil cooperativo AB**
  - académico 50%
  - social 25%,
  - capaz de ayudar 25%

- **Perfil cooperativo C**
  - académico 50%
  - social 25%
  - capaz de ayudar 12.5%
  - necesita ayuda 12.5%

- **Perfil cooperativo DE**
  - académico 50%
  - social 25%
  - necesita ayuda 25

## Roles

Existen cuatro tipos de roles:

- **Coordinator**
  - organización >=4
  - liderazgo >=4
  - decisión>=3
  - orden en clase>=3

- **Environment**
  - organización >=3
  - liderazgo >=3
  - decisión>=3
  - orden en clase>=4

- **Speaker**
  - liderazgo >=3
  - decisión>=4

- **Supervisor**
  - organización >=4
  - liderazgo >=3

## Requisitos de los grupos

Para poder formar un grupo hay que reunir una serie de características:

- Que los **perfiles** se compenetren, es decir que los que necesiten ayuda estén con los que dan ayuda. Los posibles grupos serán:
    - AB AB AB AB -> 4AB
    - AB AB AB DE -> 3AB + 1DE
    - AB AB DE DE -> 2AB + 2DE
    - AB AB AB C -> 3AB + 1C
    - AB AB C C -> 2AB + 2C
    - AB C C C -> 1AB + 3C
    - AB AB C DE -> 2AB + 1C + 1DE
    - AB C C DE -> 1AB + 2C + 1DE
    - C C C C -> 4C
- Que haya una persona de cada **rol**.
    - Coordinator + Environment + Speaker + Supervisor
- Que importe el **sexo**, para que si es posible la mayoría de los del grupo sean mujeres.
- Que no haya ningún alumno sentado con alguien que **no quiere sentarse**.

## Cosas a tener en cuenta

Para poder hacer un grupo cuyos perfiles sean correctos se ha asignado a cada tipo de perfil una puntuación, a saberse, A=10, C=5 y DE=1.

Por lo tanto, los posibles grupos quedarían con está puntuación:

- AB AB AB AB -> 40
- AB AB AB DE -> 31
- AB AB DE DE -> 22
- AB AB AB C -> 35
- AB AB C C -> 30
- AB C C C -> 25

- AB AB C  DE -> 26
- AB C  C  DE -> 21
- C  C  C  C  -> 20

Como se puede observar la puntuación más baja que se puede obtener es 20, por lo tanto, cualquier grupo que carezca de como mínimo esa puntuación será invalido.

# Algoritmo

## algoritmo.php

Fichero que contiene la llamada a funciones que realiza la organización completa de los grupos.

Si no existiera el archivo json llevaría una página de error que informaría del problema.

```php
<?php
    //Incluimos todas las funciones que utiliza el algoritmo
    include_once ("funciones.php");

   /*Se recogen los datos de los alumnos a través de un fichero json
   que se obtiene como un array asociativo con las características de los alumnos*/
    $alumnos = leerFichero();
    if(empty($alumnos)){  //Si no existe el archivo json, llevar a la página de error
        header('Location: esb.php?r=error');
    }else {
        //Se asignan los perfiles correspondientes a cada alumno
        asignacionPerfil($alumnos);

        //Se asignan los roles correspondientes a cada alumno
        asignacionRoles($alumnos);

        $listadoAlumnos = $alumnos;
        /**
         * Se lleva a cabo la cración de los grupos teniendo en cuenta las siguientes características en el siguiente orden:
         * 1º Cada grupo debe poseer un rol de los distintos tipos
         * 2º La suma de los puntos de los perfiles de los integrantes del grupo debe ser 20 o superior
         * 3º Los grupos deben posibilitar que el alumno se siente acorde a sus preferencias
         * 4º Los grupos deben poseer mas mujeres que hombres en la medida de lo posible
         **/

        $alumnosTotales = creacionGrupos($alumnos);

        $alumnosAgrupados = $alumnosTotales[0];
        $alumnosSobrantes = $alumnosTotales[1];

        //Introducimos en la tabla los alumnos sobrantes
        introducirAlumnosSobrantes($alumnosSobrantes, $alumnosAgrupados);

        //Recalcular puntos del grupo
        for ($i = 0; $i < sizeof($alumnosAgrupados); $i++)
            $alumnosAgrupados[$i]["ValorGrupo"] = contarPuntosGrupo($alumnosAgrupados[$i]["Alumnos"]);
    }
?>
```

## funciones.php

- LeerFichero()

  El primer módulo llamado en funciones.php es el que recoge los datos de los alumnos almacenados en un json y lo guarda en un array asociativo.

```php
function leerFichero()
{

    $data = file_get_contents("AlumnosIntro.json");
    $alumnos = json_decode($data, true);

    return $alumnos;
}
```

- MostrarJson():

Función que muestra el array de alumnos.

```php
/**
 * Función que muestra un array asociativo de forma visible
 * @function mostrarJson
 * @param $alumnos
 */
function mostrarJson(&$alumnos){
        echo "<pre>";
        print_r($alumnos);
        echo "</pre>";
}
```

- MonstrarAlumnosGrupos():

Función que muestra los alumnos con cada una de sus características, además de su rol y su perfil. También se muestra la puntuación personal de cada alumno.

```php
/**
 *  Función que muestra los datos referentes a los alumnos
 * @function mostrarAlumnosGrupos
 * @param $grupos
 */
function mostrarAlumnosGrupos($grupos)
{
    print_r("<br><b>GRUPOS</b><br>");
    for ($i = 0; $i < sizeof($grupos); $i++) {
        print_r("<b>grupos:$i</b><br>");
        for ($j = 0; $j < sizeof($grupos[$i]["Alumnos"]); $j++) {

            print_r($grupos[$i]["Alumnos"][$j]["Nombre"]);
            print_r(" ");
            print_r($grupos[$i]["Alumnos"][$j]["Rol"]);
            print_r("Perfil: ");
            print_r($grupos[$i]["Alumnos"][$j]["Perfil"]);
            print_r(" No se quiere sentar: ");
            print_r($grupos[$i]["Alumnos"][$j]["No quiero Sentarme"]);
            print_r("Sexo: ");
            if($grupos[$i]["Alumnos"][$j]["Sexo"]) print_r("Hombre");
        else print_r("Mujer");
            print_r("<br>");
        }
        echo "Valor grupos: " . $grupos[$i]["ValorGrupo"] . "<br>";
    }

}
```

- AsignacionPerfil():
  Función que asigna un perfil a cada alumno dependiendo de las características de ese alumno.

```php
/**
 *    Función que asigna los perfiles de cada alumno atendiendo
      a sus características
 * @function asignacionPerfil
 * @param $alumnos
 */
function asignacionPerfil(&$alumnos)
{
        //Recorremos el array de los alumnos
        for ($i = 0; $i < sizeof($alumnos); $i++) {
            //Obtenemos los valores de las variables
            $capacidad_de_ayudar = $alumnos[$i]['Es capaz de ayudar'];
            $necesidad_de_ayuda = $alumnos[$i]['Necesito ayuda'];

            /*Asignación de un único perfil a cada alumno*/
            /*Asignación del perfil AB*/
            if ($capacidad_de_ayudar > 2 && $necesidad_de_ayuda < 1) {
                $alumnos[$i]["Perfil"] = "AB";
            }
            /*Asignación del perfil C*/
            else if ($capacidad_de_ayudar >= 1 && $necesidad_de_ayuda >= 1) {
                $alumnos[$i]["Perfil"] = "C";
            }
            /*Asignación del perfil DE*/
            else {
                $alumnos[$i]["Perfil"] = "DE";
            }

        }

}
```

- AsignacionRoles():

Función que asigna cada rol a cada alumno en función de sus características.

```php
/**Función que lleva acabo la asignación de los roles a cada alumno
    atendiendo a sus habilidades
 * @function asignacionRoles
 * @param $alumnos*/
function asignacionRoles(&$alumnos){
    //Recorremos todos los alumnos
    for ($i = 0; $i < sizeof($alumnos); $i++) {

        //Obtenemos el valor de cada habilidad
        $organizacion = $alumnos[$i]['Organización'];
        $liderazgo = $alumnos[$i]['Liderazgo'];
        $decision = $alumnos[$i]['Decision-Participacion'];
        $orden = $alumnos[$i]['Orden clase'];

        /*Incializamos la caracteristica del ROL a un array ya que un
        alumno puede tener varios roles*/
        $alumnos[$i]["Rol"] = array();
        /*Asignamos los roles de cada alumno aplicando filtros
        a sus habilidades*/
        /*Para el ROL: coordinator*/
        if ($organizacion >= 4 && $liderazgo >= 4 && $decision >= 3 && $orden >= 3){
            array_push($alumnos[$i]["Rol"], "Coordinator");
        }
        /*Para el ROL: environment*/
        if ($organizacion >= 3 && $liderazgo >= 3 && $decision >= 3 && $orden >= 4){
            array_push($alumnos[$i]["Rol"], "Environment");
        }
        /*Para el ROL: Speaker*/
        if ($liderazgo >= 3 && $decision >= 4) {
            array_push($alumnos[$i]["Rol"], "Speaker");
        }
        /*Para el ROL: Supervisor*/
        if ($organizacion >= 4 && $liderazgo >= 3) {
            array_push($alumnos[$i]["Rol"], "Supervisor");
        }
    }
}
```

- ContarPuntosGrupo():
  Función que una vez creados grupos cuenta las puntuaciones atendiendo a lo explicado arriba (AB=10, C=5, DE=1).

```php
    /**Función que cuenta los puntos que posee un grupo atendiendo
       a los perfiles de sus alumnos
     * @function contarPuntosGrupo
     * @param $grupo
     * @return int
     */
    function contarPuntosGrupo($grupo)
    {
        $ab = 0;
        $c = 0;
        $de = 0;

        for ($i = 0; $i < sizeof($grupo); $i++) {
            if (strcmp($grupo[$i]["Perfil"], "AB") == 0) $ab += 10;
            if (strcmp($grupo[$i]["Perfil"], "C") == 0) $c += 5;
            if (strcmp($grupo[$i]["Perfil"], "DE") == 0) $de += 1;
        }
        return $ab + $c + $de;
    }
```

- creacionGrupos():
  Función que crea los grupos ya cumpliendo todas las características explicadas arriba (requisito para grupo).
  Lo primero que se hace es separar los roles que existen y recorren a esos alumnos separándolos por rol y añadiendo ese campo rol al array de alumnos.
  Como cada grupo lo tiene que integrar 4 personas, se divide la cantidad de alumnos entre 4 para saber cuántos grupos saldrán.
  Se asignará a cada rol una clave numérica para poder identificarlo.
  Posteriormente se añadirán dos campos al array de alumnos, RolesCompletos y ValorGurpo.
  RolesCompletos será un array con 4 datos (correspondiente a los 4 roles) inicializado a 0 que cambiará a 1 cuando se rellene el grupo con un rol.
  Aparte de añadir un rol a cada grupo, "eliminará" al alumno que ya ha sido escogido para un grupo y así no poder volver a elegirlo.
  Se contarán los puntos del grupo y se añadirá a ValorGrupo.
  …
  Ahora hay que balancear los grupos por perfil, es decir que los grupos contiene más puntuación que 20.
  El siguiente balanceo será acorde a con quien no se quiere sentar el alumno.
  Por último, queda el balanceo de la mujeres para intentar que haya más mujeres por grupo.

```php
/** Función que lleva a cabo el proceso de creación y
    división en grupos teniendo en cuenta:
 *  1° Cada grupo debe poseer un alumno con cada uno de los siguientes role
       (Coordinator, Environment ,Speaker, Supervisor).
 *  2° Cada grupo debe tener un perfil total aceptable,
       cumpliendo que tenga 20 o mas puntos en total.
 *  3° Tiene en cuenta las preferencias de cada alumnoa
       la hora de crear los grupos
 *  4° Se tiene en cuenta la creación de grupos donde
       el número de mujeres predomina
 * @function creacionGrupos
 * @param $alumnos
 * @return array*/
    function creacionGrupos($alumnos){
        /*Separación de los alumnos atendiendo a sus roles*/
        $Speakers = array();
        $Coordinators = array();
        $Supervisors = array();
        $Environments = array();
        /*Se recorren todos los alumnos clasificandolos
        atendiendo a su rol*/
        for ($i = 0; $i < sizeof($alumnos); $i++) {
            for ($j = 0; $j < sizeof($alumnos[$i]['Rol']); $j++) {
                if (strcmp($alumnos[$i]['Rol'][$j], 'Speaker') == 0) {
                    array_push($Speakers, $alumnos[$i]);
                }
                if (strcmp($alumnos[$i]["Rol"][$j], "Coordinator") == 0) {
                    array_push($Coordinators, $alumnos[$i]);
                }
                if (strcmp($alumnos[$i]["Rol"][$j], "Supervisor") == 0) {
                    array_push($Supervisors, $alumnos[$i]);
                }
                if (strcmp($alumnos[$i]["Rol"][$j], "Environment") == 0) {
                    array_push($Environments, $alumnos[$i]);
                }
            }

        }
    //Variable que almacena alumnos separados por rol
    $roles = array();
    array_push($roles, $Coordinators);
    array_push($roles, $Environments);
    array_push($roles, $Speakers);
    array_push($roles, $Supervisors);

    /*Cada grupo poseerá 4 integrantes por lo que dividiendo
    la cantidad de alumnos entre 4 obtenemos el número de grupo*/
    $numeroGrupos = sizeof($alumnos) / 4;

    $grupos = array();

    /**
    Posiciones array RolesCompletos
        0 -> Coordinator
        1 ->Environment
        2 ->Speaker
        3 ->Supervisor
     * */
```

```
    //Recorremos los grupos
    for ($i = 0; $i < $numeroGrupos; $i++) {
        /*Añadimos a cada grupo las características:
            "Roles Completos", "Alumnos", "ValorGrupo" */
        array_push($grupos,["RolesCompletos" =>[0, 0, 0, 0]
                            "Alumnos" =>[],
                            "ValorGrupo" => 0]);

        //Recorremos los roles
        for ($k = 0; $k < 4; $k++) {
            //Si posee el grupo algun rol incompleto se intenta rellenar
            if ($grupos[$i]["RolesCompletos"][$k] == 0) {
                //Si el integrante del rol no está vacio
                if (!empty($roles[$k][0])) {
                    /*Meterlo en el grupo y buscar ese alumno en
                    los roles al que pertenece eliminandolo de los mismos*/
                    array_push($grupos[$i]["Alumnos"], $roles[$k][0]);
                    //Ponemos a 1 el rol completado
                    $grupos[$i]["RolesCompletos"][$k] = 1;
                    //El alumno buscado será el que se eliminará en el array de roles
                    $alumnoBuscado = $roles[$k][0];

                    /*Buscar en el array de cada Rol y eliminar el alumno
                    ya insertado en la variable de grupos*/
                    for ($l = 0; $l < 4; $l++) {

                        //Obtenemos la variable que indica donde está el alumno
                        $index = array_search($alumnoBuscado, $roles[$l]);
                        //Si la ha encontrado se elimina del grupo
                        if ($index !== FALSE) {
                            //Se elimina de roles[$i] el elemento index evitando
                            //los huecos entre los elementos
                            array_splice($roles[$l], $index, 1);

                        }
                    }
                }
            }
        }
        /*Contamos los puntos de cada grupo y los añadimos
        a la característica del grupo*/
        $grupos[$i]['ValorGrupo'] = contarPuntosGrupo($grupos[$i]["Alumnos"]);

    }
    /*Las personas que han sobrado las metemos en un grupo aparte para conocer
    las personas que no hacen grupo*/
    $alumnosSobrantes = array();
    //Recorremos los roles de nuevo y quien se halla quedado esas
    //personas no han formado grupo
    for($i = 0;$i<4;$i++){
        if(sizeof($roles[$i])>0) {
            for ($j = 0; $j < sizeof($roles[$i]); $j++) {
                //Los metemos en un array aparte
                array_push($alumnosSobrantes,$roles[$i][$j]);
            }
        }
    }

    //Se balancean los puntos de los grupos
    balancearPerfilesGrupos($grupos, $numeroGrupos);

    //Se balancean los grupos atendiendo a sus preferencias
    balancearPreferenciasGrupos($grupos, $numeroGrupos);

    //Se balancean los grupos para que existan mas mújeres en los grupos que hombres
    balancearMujeresHombres($grupos,$numeroGrupos);


    $alumnosTotales = array();
    array_push($alumnosTotales,$grupos);
    array_push($alumnosTotales,$alumnosSobrantes);

    return $alumnosTotales;
}
```

- balancearPerfilesGrupos():
  Función que balance los grupos para que cumplan la condición de más de 20 puntos. Si algún grupo no la cumple se buscará un candidato para ser cambiado y creará grupo simulados hasta encontrar un grupo valido.

```php
/**
 * Función que balanceará los grupos para que se cumpla que el
 * mayor número de grupos poseen 20 puntos
 * @function balancearPerfilesGrupos
 * @param $grupos
 * @param $numeroGrupos
 */
function balancearPerfilesGrupos(&$grupos, $numeroGrupos)
{
    //Recorremos los grupos para realizar los balanceos
    for ($i = 0; $i < $numeroGrupos; $i++) {
        //Buscamos el primer grupo con cantidad menor de 20 puntos
        if (!empty($grupos[$i]["ValorGrupo"]) && $grupos[$i]["ValorGrupo"] < 20) {

            //Comparacion con otro grupo e intercambio de los elementos
            for ($k = 0; $k < $numeroGrupos; $k++) {

                //Evitamos el intercambio con nosotros mismos
                if ($k != $i) {

                    //Accedemos al alumno del grupo que será intercambiado
                    $grupoSimuladoA = $grupos[$i]; //Grupo menor a 20 inicial
                    $grupoSimuladoB = $grupos[$k];
                    //Puntuaciones tras el intercambio
                    $puntacionSimuladoA = contarPuntosGrupo($grupoSimuladoA["Alumnos"]);
                    $puntacionSimuladoB = contarPuntosGrupo($grupoSimuladoB["Alumnos"]);

                    /*Si la suma de ambos grupos no llega a 40
                    puntos entonces no tiene sentido el intercambio*/
                    if(($puntacionSimuladoA + $puntacionSimuladoB) < 40) break;

                    //Comparamos cada alumno de un rol con otro rol
                    for ($l = 0; $l < sizeof($grupos[$i]["Alumnos"]); $l++) {

                        //Si no está vacio el alumno del intercambio
                        if (!empty($grupoSimuladoB["Alumnos"][$l])) {

                            //Realizamos copias de los grupos
                            $alumnoIntercambio = $grupoSimuladoB["Alumnos"][$l];
                            //Realizamos el intercambio entre los grupos copia
                            $grupoSimuladoB["Alumnos"][$l] = $grupoSimuladoA["Alumnos"][$l];
                            $grupoSimuladoA["Alumnos"][$l] = $alumnoIntercambio;


                            $puntacionSimuladoA = contarPuntosGrupo($grupoSimuladoA["Alumnos"]);
                            $puntacionSimuladoB = contarPuntosGrupo($grupoSimuladoB["Alumnos"]);

                            /*Si no llegan a 40 la suma de los dos grupos
                            entonces no tiene sentido el intercambio*/
                            if(($puntacionSimuladoB+$puntacionSimuladoA)<40) break;

                            /*Si en los grupos en los que se ha intercambiado
                            se igualan o superan los 20 puntos entonces están correctos*/
                            if ($puntacionSimuladoA >= 20 && $puntacionSimuladoB >= 20) {
                                /*Copiamos los grupos copia sobre los grupos
                                originales efectuando la copia*/
                                $grupos[$i] = $grupoSimuladoA;
                                $grupos[$k] = $grupoSimuladoB;

                                //Recalculamos los puntos de cada grupo original tras el intercambio
                                $grupos[$i]["ValorGrupo"] = contarPuntosGrupo($grupos[$i]["Alumnos"]);
                                $grupos[$k]["ValorGrupo"] = contarPuntosGrupo($grupos[$k]["Alumnos"]);

                                break;

                            }
                        }
                    }
                }
            }
        }
    }
```

- busquedaEnGrupo():

  Función que realiza una búsqueda para saber si todos los alumnos del grupo están sentados con la gente que quieren. Si algún alumno está sentado con alguien que no quiere devolvería 1 (true).

```php
/**
 * Función que establece si la persona que posee el arrayNoQuieroSentarme,
 * estaría agusto en ese grupo o no
 * @function busquedaEnGrupo
 * @param $grupo
 * @param $arrayNoQuieroSentarme
 * @return int
 */
function busquedaEnGrupo($grupo, $arrayNoQuieroSentarme){
    //Recorremos cada grupo de alumnos
    for($i=0;$i<sizeof($grupo);$i++){
        /*Recorremos el array de personas con
        las que no se quiere sentar el alumno*/
        for($k=0;$k<sizeof($arrayNoQuieroSentarme);$k++){
            /*Si encuentra alguna persona con la que
            no quiere sentarse devuelve true*/
            if(strcmp($arrayNoQuieroSentarme[$k],$grupo[$i]["Nombre"])==0){
                return 1;
            }
        }
    }
    //Si está agusto en el grupo devulve false
    return 0;
}
```

- contarMujeresGrupo():
  Función que cuenta el número de mujeres por grupo cuando le pasas tu el numero de mujeres que quieres como máximo.
  Te devuelve un 1 si se pueden meter más mujeres en el grupo y un 0 si ya no se puede incluir más mujeres.

```php
    /*
     * Función que cuenta el número de mújeres de un grupo
     * @function contarMujeresGrupo
     * @param $grupos
     * @param $numeroChicasMaxGrupo
     */
    function contarMujeresGrupo($grupo,$numeroChicasMaxGrupo){
        $numeroChicas = 0;
        //Recorremos el grupo contando el numero de chicas
        for($i=0;$i<sizeof($grupo["Alumnos"]);$i++){
            if($grupo["Alumnos"][$i]["Sexo"] == 0) $numeroChicas++;
        }
        /*Si el número de chicas en el grupo es menor al
        máximo entonces se puede insertar mas sino no*/
        return $numeroChicas<$numeroChicasMaxGrupo? 1:0;
    }
```

- balancerMujerHombres():
  Función que balancea el número de mujeres que hay por grupo, intercambiado hombres por mujeres para que el grupo quede mayoritariamente femenino y tendiendo que respetar el reto de preferencias (rol, perfil y no me quiero sentar).

```php
/**
 * Función que realiza los intercambios entre hombres y mujeres de los distintos grupos,
 * para que al intercambiarse queden en cada grupo el mayor número de mujeres,
 * respetando las preferencias, los roles, y los puntos de cada
 * grupo mayores a 20
 * @function balancearMujeresHombres
 * @param $grupos
 * @param $numeroGrupos
 */
function balancearMujeresHombres(&$grupos, $numeroGrupos){

    //Recorremos los grupos para realizar los balanceos
    for ($i = 0; $i < $numeroGrupos; $i++) {
        //  print_r("<h2>Grupo ".$i."</h2>");
        // /Dentro del grupo Recorremos cada alumno en busca de un intercambio o no
        for ($k = 0; $k < sizeof($grupos[$i]["Alumnos"]); $k++) {
            ///Si se encuentra en el grupo uno de los que no quiere sentarse se realiza el intercambio
            //$arrayNoQuieroSentarme = $grupos[$i]["Alumnos"][$k]["No quiero Sentarme"];
            //   print_r("<h4>El alumno ".$grupos[$i]["Alumnos"][$k]["Nombre"]." </h4>>");

            /*Si el que queremos cambiar es un hombre (A) y se cumple que
            haya menos de $numeroChicasMaxGrupo por grupo*/
            if ($grupos[$i]["Alumnos"][$k]["Sexo"] == 1 && contarMujeresGrupo($grupos[$i],3) == 1) {

                //print_r("El alumno" .$grupos[$i]["Alumnos"][$k]["Nombre"] ."no le gusta el grupo<br>");

                //Buscamos otra persona de otro grupo del mismo rol
                for ($l = 0; $l < $numeroGrupos; $l++) {
                    //Evitar los intercambios en el mismo grupo
                    if ($i != $l) {
                        //Accedemos al alumno del grupo que será intercambiado
                        $grupoSimuladoA = $grupos[$i]; //Grupo menor a 20 inicial
                        $grupoSimuladoB = $grupos[$l];

                        if(!empty( $grupoSimuladoB["Alumnos"][$k]) && !empty($grupoSimuladoA["Alumnos"][$k])) {
                            $estudianteIntercambioB = $grupoSimuladoB["Alumnos"][$k];
                            $estudianteIntercambioA = $grupoSimuladoA["Alumnos"][$k];

                            $grupoSimuladoA["Alumnos"][$k] = $estudianteIntercambioB;
                            $grupoSimuladoB["Alumnos"][$k] = $estudianteIntercambioA;

                            //Puntuaciones tras el intercambio
                            $puntacionSimuladoA = contarPuntosGrupo($grupoSimuladoA["Alumnos"]);
                            $puntacionSimuladoB = contarPuntosGrupo($grupoSimuladoB["Alumnos"]);

                            if(($puntacionSimuladoB+$puntacionSimuladoA )< 40) break;
                            //Array de las personas con las que no quiere sentarse B
                            $arrayNoQuieroSentarmeEnB = $estudianteIntercambioB["No quiero Sentarme"];
                            /*Si la persona que entra acepta a los de su grupo y
                            el intercambio produce mas de 20 puntos en cada grupo...*/
                            if (busquedaEnGrupo($grupoSimuladoB["Alumnos"], $arrayNoQuieroSentarmeEnB)
                                && $puntacionSimuladoA >= 20
                                && $puntacionSimuladoB >= 20
                                && $estudianteIntercambioB["Sexo"] == 0
                            ) {
                                $grupos[$i] = $grupoSimuladoA;
                                $grupos[$l] = $grupoSimuladoB;
                                $grupos[$i]["ValorGrupo"] = contarPuntosGrupo($grupos[$i]["Alumnos"]);
                                $grupos[$l]["ValorGrupo"] = contarPuntosGrupo($grupos[$l]["Alumnos"]);

                                break;
                            }
                        }
                    }
                }
            }
        }
    }
}
```

- balancerPreferenciasGrupos():

  Función que hace los intercambios para que las personas se encuentren en el grupo con personas con las que quieran sentarse, además de tener en cuenta el rol, y que el intercambio respete que el grupo posea 20 o más puntos.

```php
/** Función que lleva a cabo los intercambios para que las personas se
 *  encuentren en el grupo con personas con las que quieran sentarse,
 *  además de tener en cuenta el rol, y que el intercambio respete que
 *  el grupo posea 20 o mas puntos
 * @function balancearPreferenciasGrupos
 * @param $grupos
 * @param $numeroGrupos
 */
function balancearPreferenciasGrupos(&$grupos, $numeroGrupos){

    //Recorremos los grupos para realizar los balanceos
    for ($i = 0; $i < $numeroGrupos; $i++) {
        //Dentro del grupo Recorremos cada alumno en busca de un intercambio o no
        for ($k = 0; $k < sizeof($grupos[$i]["Alumnos"]); $k++) {

            /*Si se encuentra en el grupo uno de los que no
            quiere sentarse se realiza el intercambio*/
            $arrayNoQuieroSentarme = $grupos[$i]["Alumnos"][$k]["No quiero Sentarme"];
            //Si hay alguien que no le gusta en el grupo
            if (busquedaEnGrupo($grupos[$i]["Alumnos"], $arrayNoQuieroSentarme) == true) {

                //Buscamos otra persona de otro grupo del mismo rol
                for ($l = 0; $l < $numeroGrupos; $l++) {
                    //Evitar los intercambios en el mismo grupo
                    if ($i != $l) {

                        //Copias de los grupos que van a intercambiar estudiantes
                        $grupoSimuladoA = $grupos[$i]; //Grupo menor a 20 inicial
                        $grupoSimuladoB = $grupos[$l];

                        //Estudiantes intercambiados
                        $estudianteIntercambioB = $grupoSimuladoB["Alumnos"][$k];
                        $estudianteIntercambioA = $grupoSimuladoA["Alumnos"][$k];

                        //Realizamos el intercambio entre los estudiantes simulados
                        $grupoSimuladoA["Alumnos"][$k] = $estudianteIntercambioB;
                        $grupoSimuladoB["Alumnos"][$k] = $estudianteIntercambioA;
```

```php
                        //Puntuaciones tras el intercambio
                        $puntacionSimuladoA = contarPuntosGrupo($grupoSimuladoA["Alumnos"]);
                        $puntacionSimuladoB = contarPuntosGrupo($grupoSimuladoB["Alumnos"]);

                        //Si el intercambio
                        if(($puntacionSimuladoA+$puntacionSimuladoB)<40) break;

                        //Array de las personas con las que no quuiere sentarse el alumno B
                        $arrayNoQuieroSentarmeEnB = $estudianteIntercambioB["No quiero Sentarme"];

                        /*Si la persona que entra acepta a los de su grupo y el intercambio
                        produce mas de 20 puntos en cada grupo...*/
                        if (busquedaEnGrupo($grupoSimuladoB["Alumnos"], $arrayNoQuieroSentarmeEnB)
                            && $puntacionSimuladoA >= 20
                            && $puntacionSimuladoB >= 20){

                            //Se produce el intercambio copiando el grupoSimuladoA al real
                            $grupos[$i] = $grupoSimuladoA;
                            $grupos[$l] = $grupoSimuladoB;
                            $grupos[$i]["ValorGrupo"] = contarPuntosGrupo($grupos[$i]["Alumnos"]);
                            $grupos[$l]["ValorGrupo"] = contarPuntosGrupo($grupos[$l]["Alumnos"]);

                            break;
                        }
                    }
                }
            }
        }
    }
}
```

- introducirAlumnosSobrantes():
  Función que introduce a los alumnos que queden fuera de los grupos ya formados.
  Se intentarán incluir los alumnos nel so grupos que encajan.

```php
/**
 * Función que introduce los alumnos sin agrupar,
 * en los grupos menos favorecidos por los puntos.
 * @function introducirAlumnosSobrantes
 * @param $alumnosSobrantes
 * @param $alumnosAgrupados
 */
    function introducirAlumnosSobrantes(&$alumnosSobrantes,&$alumnosAgrupados){

        //Introducir los alumnos donde encajen
        $puntuacionGrupoMenosPersonas = $alumnosAgrupados[0]["ValorGrupo"];
        $indiceGrupoMenosPersonas = null;
        if(sizeof($alumnosSobrantes) == 0) return 1;

        //Recorremos el grupo en busca del grupo con menos candidatos
        for($i=0;$i<sizeof($alumnosAgrupados);$i++){
            if($puntuacionGrupoMenosPersonas > $alumnosAgrupados[$i]["ValorGrupo"]){
                $puntuacionGrupoMenosPersonas = $alumnosAgrupados[$i]["ValorGrupo"];
                $indiceGrupoMenosPersonas = $i;
            }
        }
        $alumnosSobrantes[0]["SobroEnElgrupo"] = true;
        //Metemos el alumno en el grupo con menos personas
        array_push($alumnosAgrupados[$indiceGrupoMenosPersonas]["Alumnos"],$alumnosSobrantes[0]);

        //Metemos un campo para indicar los alumnos repescados
        $alumnosAgrupados[$indiceGrupoMenosPersonas]["AlumnosRepescados"] = true;

        //Eliminamos el alumno porque ya no es sobrante
        array_splice($alumnosSobrantes, 0, 1);

        /*Hacemos una llamada recursiva para que lo haga con
        todos los alumnosSobrantes hasta que no queden ninguno*/
        introducirAlumnosSobrantes($alumnosSobrantes,$alumnosAgrupados);
    }
```

# Cronograma

## Primer día de trabajo: Dia 28 de Diciembre

Tiempo: de 16:30 a 21:00

Trabajo en grupo vía Skype los 4

- Realización de un json de prueba con 5 alumnos
- Lectura del Json desde codigo
- Evaluación de las características de cada alumno y asignación de perfiles y roles
- En proceso a dia de hoy: reparto de grupos teniendo en cuenta perfiles y roles.

Los diferentes pasos se han pensado en común, usando como plataforma para la actualización de los diferentes archivos a utilizar la nube de Google Drive.

## Segundo día de trabajo: Día 29 de Diciembre

Tiempo: de 10:30 am hasta las 13:30

Trabajo en grupo vía Skype los 4.

- Creación de tabla de roles
- Creación de tabla de grupos
- Asignación de alumnos a grupos teniendo en cuenta el rol

- Eliminación de aquellos alumnos asignados con un rol de la tabla roles ya que poseen más de 1 rol.

Los diferentes pasos se han pensado en común, usando como plataforma para la actualización de los diferentes archivos a utilizar la nube de Google Drive.

## Tercer día de trabajo: Día 3 de Enero 2018

Tiempo: de 16:15 pm hasta las 20:00

Trabajo en grupo vía Skype los 4

- Comprobación de puntos en los diferentes grupos
- Balanceo de grupos teniendo en cuenta el ROL y el PERFIL

Los diferentes pasos se han pensado en común, usando como plataforma para la actualización de los diferentes archivos a utilizar la nube de Google Drive.

## Cuarto día de trabajo: Día 4 de Enero 2018

Tiempo de 16:00 a 18:30

Trabajo en grupo vía Skype los 4

- Balanceo de grupos en relación las preferencias que tienen los alumnos de con quien quieren sentarse y con quien no.

Los diferentes pasos se han pensado en común, usando como plataforma para la actualización de los diferentes archivos a utilizar la nube de Google Drive.

## Quinto día de trabajo: Día 8 de Enero 2018

Tiempo de 16:00 a 00:00

Trabajo en grupo vía Skype los 4

- Balanceo de grupos en relación al sexo.
- Inicio del diseño de la pagina web.
- Inicio y creación del ESB.
- Corrección de fallos del algoritmo.
- Creación de la memoria en documento word.

Los diferentes pasos se han pensado en común, usando como plataforma para la actualización de los diferentes archivos a utilizar la nube de Google Drive.

## Sexto día de trabajo: Día 9 de Enero 2018

Tiempo de 16:00 a 18:30

Trabajo en grupo vía Skype los 4

- Mostrar el listado de los alumnos
- Mostrar el listado de grupos
- Mostrar listado de alumnos que sobran
- Ordenación de código
- Separar el css

Los diferentes pasos se han pensado en común, usando como plataforma para la actualización de los diferentes archivos a utilizar la nube de Google Drive.

## Séptimo día de trabajo: Día 10 de Enero 2018

Tiempo de 16:00 a 21:00

Trabajo en grupo vía Skype los 4

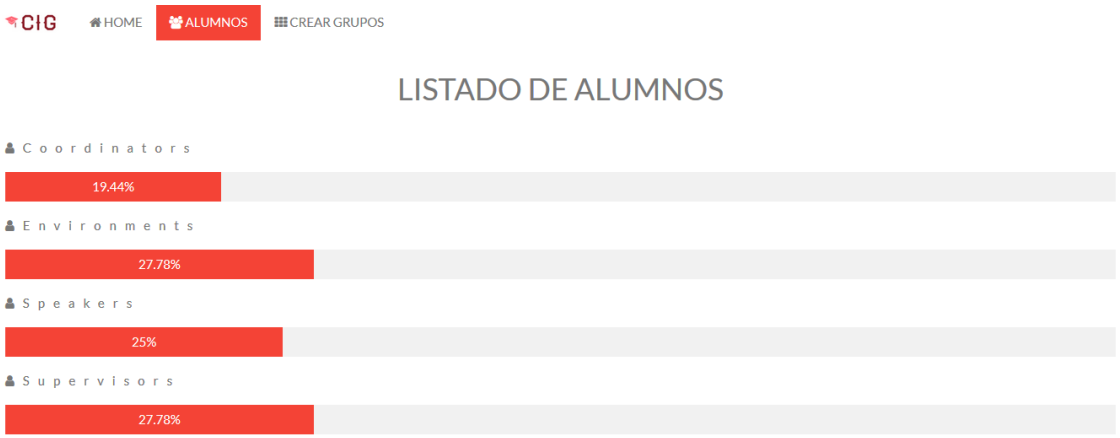
- Añadir alumnos sobrantes a grupos bien formados
- Estilizar diseño de la interfaz de la aplicación
- Comprobación de la existencia de un archivo json
- Página de error
- Pruebas
- Detección de errores

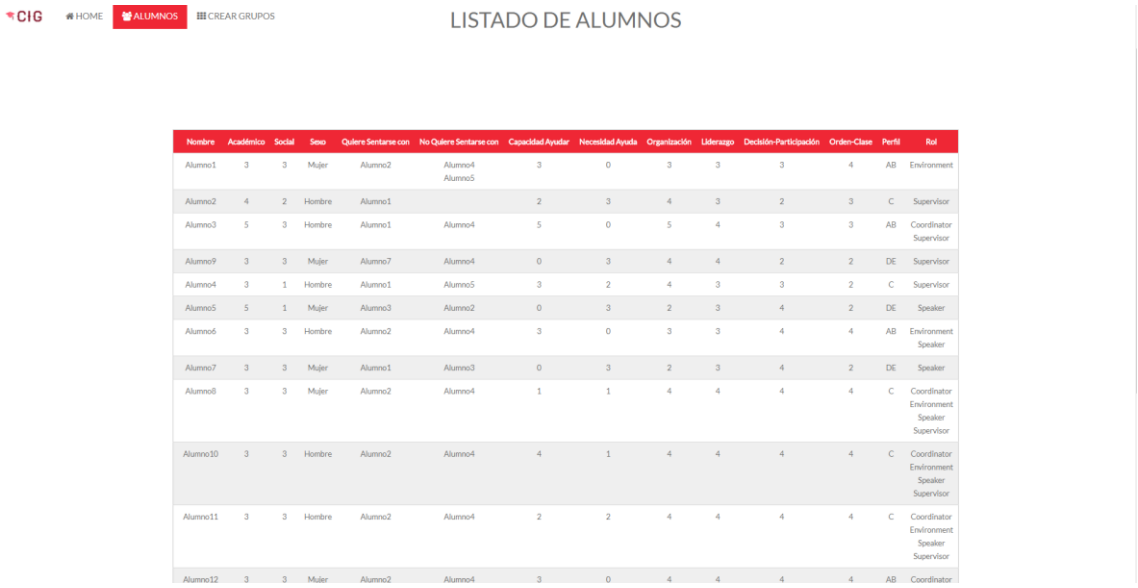
# Pruebas

Para probar la aplicación se llamará al archivo index.php desde la web.


La primera venta que muestra es la home con un portada de la aplicación.

Si pulsamos en el menú alumnos nos mostraría un listado con los alumnos, sus características, su perfil y su rol que ya ha sido asignado. Y un porcentaje de la cantidad de personas que hay de un rol.




| Nombre | Académico | Social | Sexo | Quiere Sentarse con | No Quiere Sentarse con | Capacidad Ayudar | Necesidad Ayuda | Organización | Liderazgo | Decisión-Participación | Orden-Clase | Perfil | Rol |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Alumno1 | 3 | 3 | Mujer | Alumno2 | Alumno4 Alumno5 | 3 | 0 | 3 | 3 | 3 | 4 | AB | Environment |
| Alumno2 | 4 | 2 | Hombre | Alumno1 | | 2 | 3 | 4 | 3 | 2 | 3 | C | Supervisor |
| Alumno3 | 5 | 3 | Hombre | Alumno1 | Alumno4 | 5 | 0 | 5 | 4 | 3 | 3 | AB | Coordinator Supervisor |
| Alumno9 | 3 | 3 | Mujer | Alumno7 | Alumno4 | 0 | 3 | 4 | 4 | 2 | 2 | DE | Supervisor |
| Alumno4 | 3 | 1 | Hombre | Alumno1 | Alumno5 | 3 | 2 | 4 | 3 | 3 | 2 | C | Supervisor |
| Alumno5 | 5 | 1 | Mujer | Alumno3 | Alumno2 | 0 | 3 | 2 | 3 | 4 | 2 | DE | Speaker |
| Alumno6 | 3 | 3 | Hombre | Alumno2 | Alumno4 | 3 | 0 | 3 | 3 | 4 | 4 | AB | Environment Speaker |
| Alumno7 | 3 | 3 | Mujer | Alumno1 | Alumno3 | 0 | 3 | 2 | 3 | 4 | 2 | DE | Speaker |
| Alumno8 | 3 | 3 | Mujer | Alumno2 | Alumno4 | 1 | 1 | 4 | 4 | 4 | 4 | C | Coordinator Environment Speaker Supervisor |
| Alumno10 | 3 | 3 | Hombre | Alumno2 | Alumno4 | 4 | 1 | 4 | 4 | 4 | 4 | C | Coordinator Environment Speaker Supervisor |
| Alumno11 | 3 | 3 | Hombre | Alumno2 | Alumno4 | 2 | 2 | 4 | 4 | 4 | 4 | C | Coordinator Environment Speaker Supervisor |
| Alumno12 | 3 | 3 | Mujer | Alumno2 | Alumno4 | 3 | 0 | 4 | 4 | 4 | 4 | AB | Coordinator |

Si pulsas en el menú crear grupos, te muestra los grupos que han sido formados y además un grupo con los alumnos restantes.


GRUPOS FORMADOS

Grupo 1: 21 pts

| Nombre | Académico | Social | Sexo | Quiere sentarse con | No quiere sentarse con | Capacidad ayudar | Necesidad ayuda | Organización | Liderazgo | Decision-Participación | Orden-Clase | Perfil | Rol |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Alumno8 | 3 | 3 | Mujer | Alumno2 | Alumno4 | 1 | 1 | 4 | 4 | 4 | 4 | C | Coordinator Environment Speaker Supervisor |
| Alumno1 | 3 | 3 | Mujer | Alumno2 Alumno9 | Alumno4 Alumno5 | 3 | 0 | 3 | 3 | 3 | 4 | AB | Environment |
| Alumno7 | 3 | 3 | Mujer | Alumno1 | Alumno3 | 0 | 3 | 2 | 3 | 4 | 2 | DE | Speaker |
| Alumno2 | 4 | 2 | Hombre | Alumno1 | | 2 | 3 | 4 | 3 | 2 | 3 | C | Supervisor |

Grupo 2: 25 pts

| Nombre | Académico | Social | Sexo | Quiere sentarse con | No quiere sentarse con | Capacidad ayudar | Necesidad ayuda | Organización | Liderazgo | Decision-Participación | Orden-Clase | Perfil | Rol |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Alumno10 | 3 | 3 | Hombre | Alumno2 | Alumno4 | 4 | 1 | 4 | 4 | 4 | 4 | C | Coordinator |

Los alumnos mostrados en azul son lo que no entran o no encajan en los grupos por no cumplir las restricciones y han sido añadidos a un grupo elegido.

Grupo 4: 30 pts

| Nombre | Académico | Social | Sexo | Quiere sentarse con | No quiere sentarse con | Capacidad ayudar | Necesidad ayuda | Organización | Liderazgo | Decision-Participación | Orden-Clase | Perfil | Rol |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Alumno13 | 3 | 3 | Mujer | Alumno2 | Alumno4 | 3 | 0 | 4 | 4 | 4 | 4 | AB | Coordinator Environment Speaker Supervisor |
| Alumno14 | 3 | 3 | Mujer | Alumno2 | Alumno4 | 3 | 0 | 4 | 4 | 4 | 4 | AB | Coordinator Environment Speaker Supervisor |
| Alumno15 | 3 | 3 | Mujer | Alumno2 | Alumno4 | 3 | 0 | 3 | 3 | 3 | 4 | AB | Environment |

*Alumnos de color azul: aquellos que se han añadido después de realizar los grupos

Si no existiera el archivo json, se mostraría un error informando del problema: