Universidad de San Carlos de Guatemala
Centro Universitario de Occidente
División Ciencias de la Ingeniería
Ingeniería en Ciencias y Sistemas
Modelación y Simulación 2

Laboratorio de Modelación y Simulación 2

PROYECTO I: Etapas de la modelacion y simulacion

18 de septiembre de 2,023

Luis Estuardo Bolaños Gonzalez, 201731766

Contenido

Descripción

Se le contrató como ingeniero en sistemas en la organización “la usurpación”, la cual se dedica a la educación superior, donde necesitan una solución informática que les ayude a tomar las decisiones sobre la creación de horarios de clases, actualmente invierten mucho tiempo en dicha actividad por lo que quisieran tener diferentes propuestas generadas automáticamente y elegir la que mejor se les ajuste a sus necesidades, entre las condiciones de como funciona la organización están las siguientes:

Disponen de un espacio finito de salones (s).
Cada salón tiene una disponibilidad finita de escritorios para los estudiantes (e).
Existen diferentes carreras (c) y cada carrera tiene sus propias materias (m).
Tienen profesores encargados de impartir las materias (p), estos pueden estar asignados a materias fijas o pueden cubrir otras materias si poseen las cualificaciones y hace falta personal.
Tienen asignación de cursos desde el inicio (a), por lo que se sabe que materias cursarán los estudiantes en el semestre.
Características del sistema
Este deberá ser alimentado por archivos de carga de base de datos (scripts) indicando las diferentes variables del sistema (s, e, c, m, p, a) y otras que considere necesarias.
El sistema tendrá las opciones de generar un horario tomando diferentes criterios de prioridad, por ejemplo: en base a las materias disponibles, en base a la disponibilidad de espacios en los salones, en base al horario de contratación de los profesores, entre otros.
El sistema podrá indicar advertencias en los horarios asignados.

Criterios

- Creación de horario por prioridad de materias.
- Creación de horario con cantidad de salón.

Lenguajes de programación

Backend

- NodeJS.

Frontend

- React.

Archivos de entrada

- Salones
- Catedráticos.
- Hora de inicio de un curso.
- Hora de finalización de un curso.
- Total de duración de un curso.
- Carreras

Clases y atributos

- Salon
    - No. de espacios.
    - No. Salon.
- Catedrático
    - Nombre
    - Hora de entrada.
    - Hora de salida.
    - Cualificaciones.
- Curso
    - Nombre
    - Carrera.
    - No. Asignados.
    - Catedrático.
    - Prioritario.
- Carrera
    - Nombre.

Definición del sistema

Desarrollar un algoritmo de optimización que, utilizando los parámetros y restricciones mencionados, genere un horario óptimo que maximice la satisfacción de los estudiantes y catedráticos, minimice los conflictos y aproveche eficientemente los recursos disponibles, como los salones y el tiempo de los catedráticos.

Sistema: El horario de cursos de un establecimiento educativo (universidad, colegio, curso particular, etc).

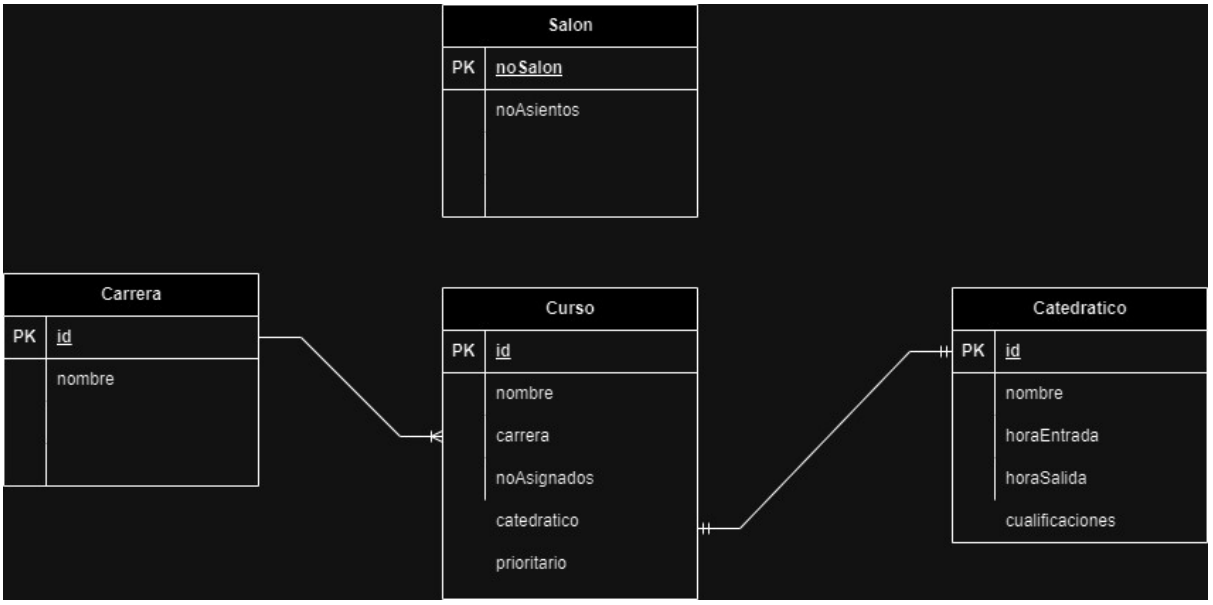
Se desea modelar un sistema de generación de horario automático para la impartición de clases en un establecimiento, para la generación de dicho horario se toman las siguientes entradas:

Entradas:

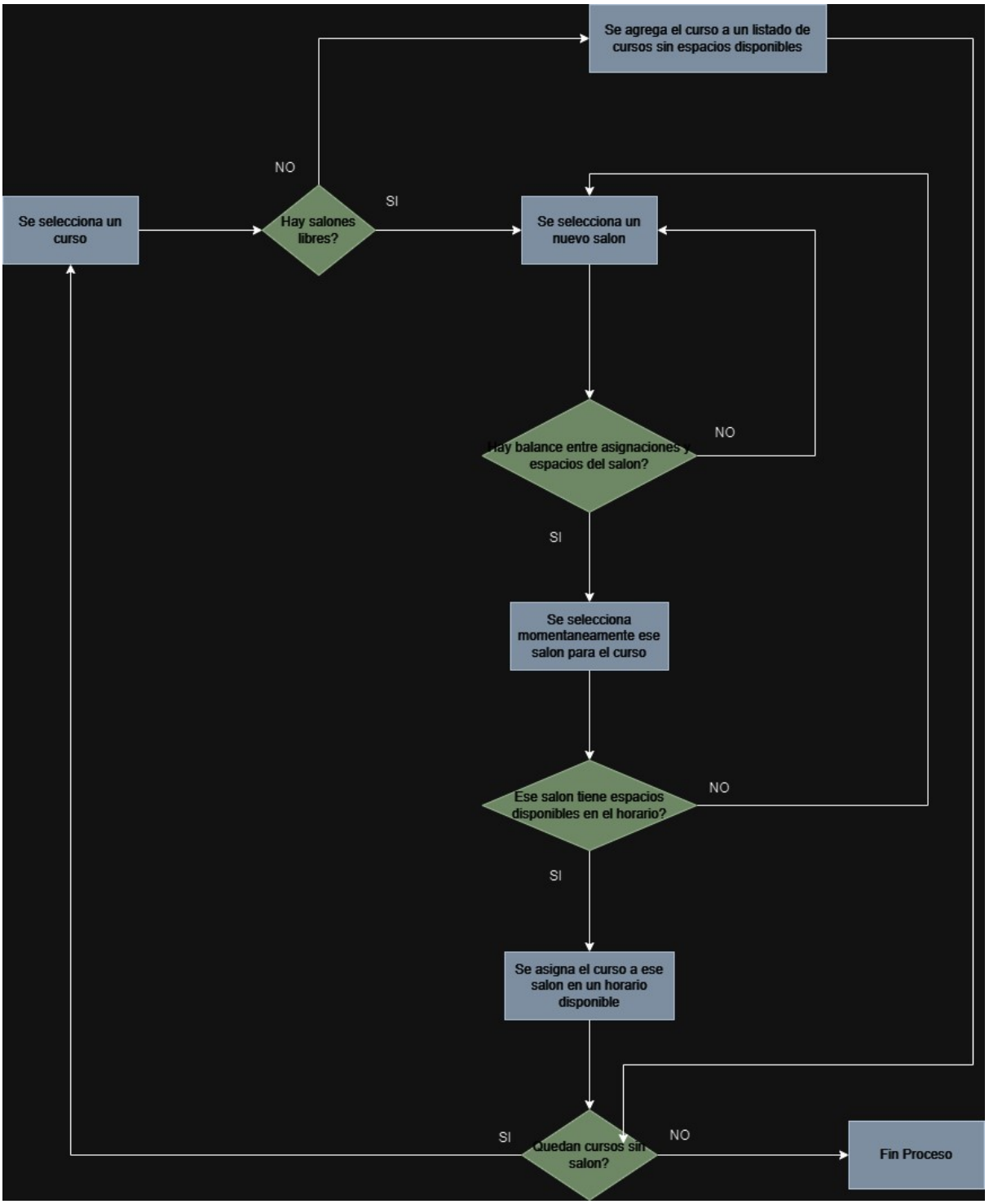
- Catedráticos encargados de impartir las clases.
    - Hora de entrada de los catedráticos.
    - Hora de salida de los catedráticos.
- Cursos existentes dentro del establecimiento
    - Total de estudiantes asignados por cursos.
    - Nivel de prioridad para encontrarle un espacio dentro del horario al curso.
- Salones disponibles en el establecimiento.
    - Total de espacios/asientos dentro de cada salón.

Formulación del modelo

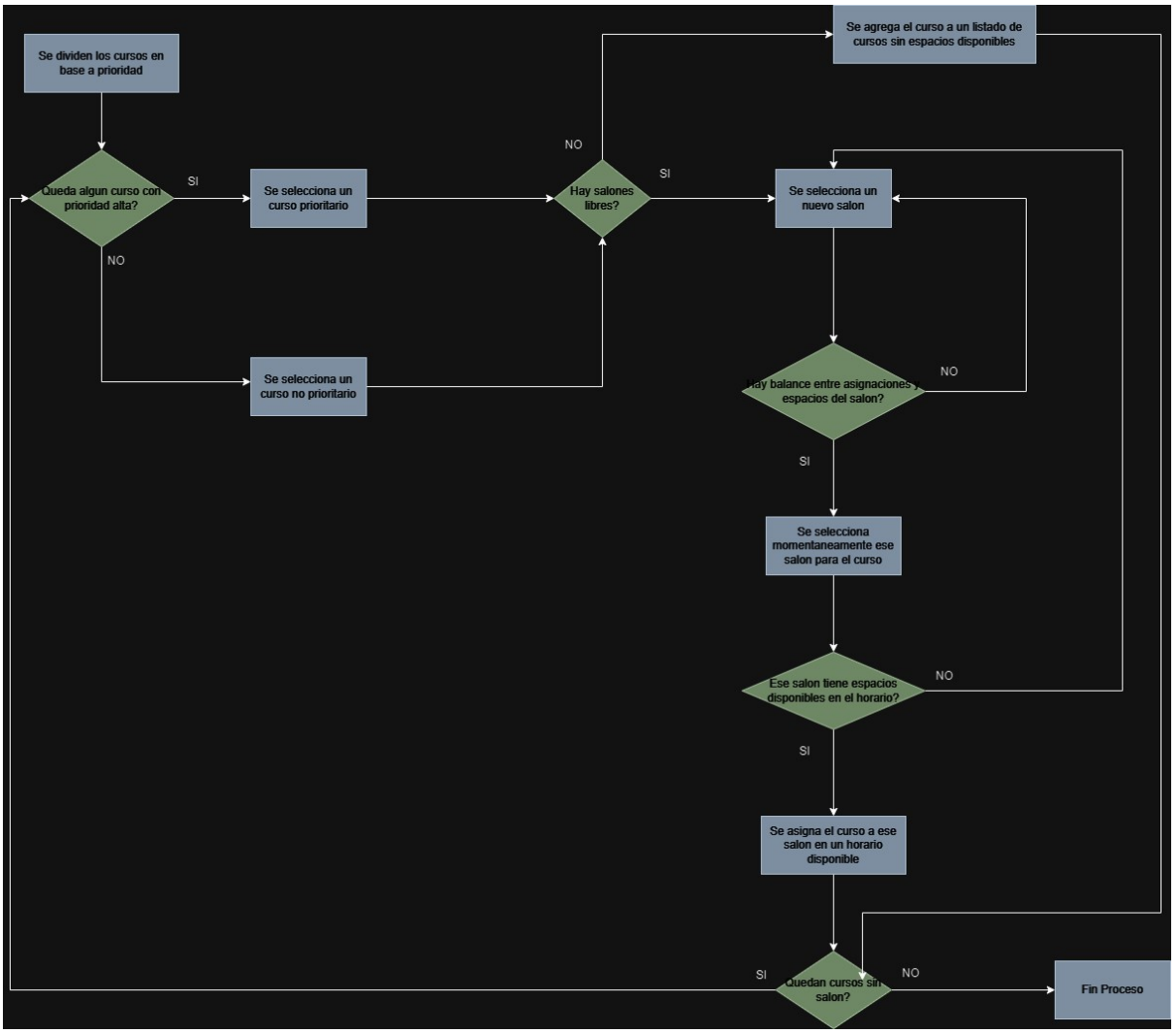
Para la definición del modelo se desarrolló una base de datos teniendo en cuenta las entradas que tendrá el sistema para su correcto funcionamiento, dichas entradas se establecieron en la fase anterior.



Adjunto a este modelo se desarrollaron los algoritmos que intervienen para la generacion de horarios según el criterio de prioridad que se elija.



Algoritmo para la creación de un horario por prioridad de cursos



Algoritmo para la creación de un horario por prioridad de espacios por salón

Colección de datos

Los datos que se necesitan para el correcto funcionamiento del sistema son los siguientes:

- Catedráticos encargados de impartir las clases.
    - Hora de entrada de los catedráticos.
    - Hora de salida de los catedráticos.
- Cursos existentes dentro del establecimiento
    - Total de estudiantes asignados por cursos.
    - Nivel de prioridad para encontrarle un espacio dentro del horario al curso.
- Salones disponibles en el establecimiento.
    - Total de espacios/asientos dentro de cada salón.

Dichos datos se obtienen de una base de datos.

Implementación del modelo en la computadora

Para la implementación de los algoritmos y modelos previamente creados se decidió dividir el sistema en diferentes partes, cada parte desarrollada en un diferente lenguaje de programación.

Backend

- NodeJS como entorno de desarrollo.
- Sequelize como ORM para facilitar la comunicación entre backend y DB.
- Express como framework para el desarrollo de la API que conecte backend con frontend.

Frontend

- Javascript para el desarrollo del frontend.
- ReactJS como framework para la felicitación del desarrollo de las vistas.

Base de datos

- MySQL como motor SQL para la creación de la base de datos.

Verificación

Se adjuntan capturas del desarrollo del modelo como forma de verificación de que el desarrollo cumple con las expectativas impuestas por los algoritmos creados previamente.

```
const encontrarSalon = (curso, salones, salonesDescartados) => {
    let salonElegido = { noSalon: 0, asientosVacios: -1 };
    for(let i = 0; i < salones.length; i++){
        if(salonesDescartados.includes(i)) continue;
        if(curso.noAsignados === salones[i].noAsientos){
          //con este salon se procede a verificar si tiene espacios disponibles en el horario
          salonElegido = { noSalon: i, asientosVacios: 0 };
          break;
        } else if(curso.noAsignados < salones[i].noAsientos){
            let aux = { noSalon: i, asientosVacios: salones[i].noAsientos - curso.noAsignados }
            if(salonElegido.asientosVacios === -1){
                salonElegido = aux;
            } else {
                if(aux.asientosVacios < salonElegido.asientosVacios ){
                    salonElegido = aux;
                }
            }
        }
    }
    //No hay salon que pueda cumplir con la cantidad correcta de asientos para este curso
    if(salonElegido.asientosVacios === -1){
       console.log('Error para el curso ',curso.carrera, ' ',curso.nombre_curso,salonElegido.asientosVacios);
       cursosSinHogar.push({
        carrera: curso.carrera,
        curso: curso.nombre_curso
       })
    } else {
      //Para este punto ya se tiene el salon a verificar si tiene espacios vacios
      const salon = salones[salonElegido.noSalon];
      let encontrado = false;
      for (let i = 0; i < salon.espacios.length; i++) {
        if (salon.espacios[i].curso === undefined) {
          salon.espacios[i].curso = curso.carrera + ' ' + curso.nombre_curso;
          encontrado = true;
          break;
        }
      }
      if (!encontrado) {
        salonesDescartados.push(salonElegido.noSalon);
        encontrarSalon(curso, salones, salonesDescartados, curso.carrera);
      }
    }
}
```

```
const generarHorario = (dataCursos, dataSalones) => {
    let result = {};
    const asignacionesCompletas = [];
    limpiarData(dataCursos, dataSalones);
    console.log('hola');
    separarCursosPorPrioridad(dataCursos)
    cursosPrioritarios.forEach(curso => {
      encontrarSalon(curso, salones, [])
    });
    cursosNoPriotirarios.forEach(curso => {
      encontrarSalon(curso, salones, [])
    })

    salones.forEach((salon) => {
      salon.espacios.forEach((espacio) => {
        console.log("Salon", salon.noSalon, "-", espacio.hora, espacio.curso);
        asignacionesCompletas.push({
          salon: salon.noSalon,
          horario: espacio.hora,
          curso: espacio.curso,
        });
      });
    });


    result = {
        asignaciones: asignacionesCompletas,
        cursosSinHogar: cursosSinHogar
    }
    return result;
}
```

```
const separarCursosPorPrioridad = (dataCursos) => {
    dataCursos.forEach(curso => {
        if(curso.prioritario === 1){
            cursosPrioritarios.push(curso);
        } else if(curso.prioritario === 0) {
            cursosNoPriotirarios.push(curso);
        }
    })
}
```

```
const generarHorarioPorEspacios = async(req, res) => {
    try {
        const [dataCursosAsc, metadata] = await sequelize.query(
          "SELECT curso.id AS curso_id, carrera.nombre AS carrera, curso.nombre AS nombre_curso, curso.noAsignados AS noAsignados, curso.prioritario AS prioritario FROM carrera JOIN curso
        );
        const [dataCursosDesc, metadata2] = await sequelize.query(
          "SELECT curso.id AS curso_id, carrera.nombre AS carrera, curso.nombre AS nombre_curso, curso.noAsignados AS noAsignados, curso.prioritario AS prioritario FROM carrera JOIN curso
        );
        const salones = await Salon.findAll({
            raw: true
        });
        const asignaciones = service.generarHorario3(dataCursosAsc, salones);
        const asignacionesDesc = service.generarHorario3(dataCursosDesc, salones);

        const result = {
            opciones: [
                { ...asignaciones},
                { ...asignacionesDesc}
            ],
            noSalones: salones.length
        }
        return res.status(200).send(result);
    } catch(error) {
        return res.status(500).send(error.message);
    }
}
```

Experimentación

Se utilizaron los siguientes scripts de base de datos para poner a prueba el sistema previamente creado:

- Carreras

```
"id";"nombre"
1;"Ingenieria mecanica"
2;"Ingenieria en sistemas"
```

- Catedráticos

```
"id";"nombre";"horaEntrada";"horaSalida";"cualificaciones"
1;"Juan Perez";2;6;"matematicas,fisica,quimica"
2;"Carlos Ramirez";2;4;"quimica,programacion,compiladores"
3;"Manuel Gramajo";3;5;"mecanica,matematica,fisica"
4;"Fernando Martinez";2;6;"contabilidad,mecanica,seminario"
```

- Cursos

```
"id";"nombre";"carrera";"noAsignados";"catedratico";"prioritario"
1;"Curso 1";1;25;1;1
2;"Curso 2";1;14;3;1
3;"Curso 3";1;16;3;1
4;"Curso 4";1;31;1;0
5;"Curso 5";1;27;4;0
6;"Curso 6";1;10;1;1
7;"Curso 1";2;11;1;0
8;"Curso 2";2;9;2;1
9;"Curso 3";2;34;2;1
10;"Curso 4";2;39;2;1
11;"Curso 5";2;22;2;1
12;"Curso 6";2;19;4;1
```

- Salones

```
"noSalon";"noAsientos"
1;30
2;20
3;40
```

Interpretación

Una vez el sistema fue puesto a prueba con diferentes datos (cursos, carreras, catedráticos, salones), se puede llegar a la conclusión de que la creación del horario se adecua a lo establecido con los algoritmos, hay cosas que se pueden mejorar como por ejemplo

- Lanzar más posibilidades de horarios para que el usuario elija cual es el más conveniente para todas las partes.
- Más criterios de prioridad para la generación de nuevos horarios.

Como decisión luego de ver los resultados se puede indicar que el sistema es aceptable y puede ser utilizado en situaciones reales con los establecimientos interesados en el sistema teniendo en cuenta de las mejoras que este puede tener con el futuro y los posibles factores que el sistema no aborda por completo actualmente para la generación de horarios.

Documentación

Se adjuntan manuales e imágenes de los algoritmos y diseños de base de datos.