Pontificia Universidad Javeriana
Departamento de Ingeniería de Sistemas
Estructuras de Datos
Proyecto del curso, 2017-10

1 Descripción del problema

El Sistema de Información Universitaria (SIU) es la columna vertebral de la Universidad en cuanto al manejo de la información de sus estudiantes, profesores, administrativos y otros estamentos implicados. Se encarga de la gestión de recursos humanos, procesos financieros y administrativos, procesos académicos, y la administración de proyectos de investigación y consultoría. Prácticamente todas las actividades realizadas dentro de la Universidad pasan de alguna manera a través del SIU, y en el mundo digital en el que estamos inmersos actualmente, alguna falla en el sistema implica ralentización en los procesos y pérdidas importantes de tiempo y recursos.

El SIU se compone de diferentes subsistemas especializados, algunos de ellos son:

- Sistema de Administración de Estudiantes (SAE): se encarga de la administración de toda la información referente a los estudiantes, planes de estudio, inscripción de asignaturas, horarios y listas de clase, etc.
- Sistema de Administración de Proyectos (SIAP): permite la gestión de los proyectos de investigación y consultoría llevados a cabo por los diferentes estamentos de la Universidad.
- Sistema de Registro Académico de Profesores (RAP): registra toda la información relacionada con los profesores y su quehacer en la Universidad: contratos, planes de trabajo, nómina, etc.

Uno de los procesos más importantes dentro de la Universidad tiene que ver con la programación de asignaturas y la inscripción de estudiantes, para planificar las labores de cada semestre. De esta forma, el conocimiento que cada estudiante debe adquirir a lo largo de su carrera está distribuído en diferentes asignaturas, y de acuerdo a la demanda semestral, cada asignatura puede ofrecerse a través de varios grupos (denominados clases), con días y horarios diferentes, para que el estudiante pueda escoger y armar su horario semestral de la forma más conveniente posible.

En el semestre, la primera parte del proceso, que corresponde a la oferta de asignaturas, implica definir para cada asignatura la cantidad de clases a ofertar, y para cada clase, definir días, horas y salones donde éstas se llevarán a cabo. La información básica que se almacena de cada asignatura es su identificador individual (que se fija al momento de incorporar la asignatura al catálogo) y el nombre que la describe. Para cada clase, en el semestre se asigna un identificador individual y se identifican las sesiones de esa clase, donde cada sesión tiene asignado un día de la semana, una hora de inicio, una hora de finalización y un salón disponible del campus, el cual está identificado por el edificio y el número de salón. Un ejemplo de esta información se presenta a continuación:

ID asignatura	**Nombre asignatura**
3195	Fundamentos de Programación

ID clase: 6622

Sesión	**Día**	**Hora inicio**	**Hora fin**	**Salón**
1	Lunes	9:00am	11:00am	SA067-P705
2	Miércoles	9:00am	11:00am	SC009-P101

ID clase: 6623

Sesión	**Día**	**Hora inicio**	**Hora fin**	**Salón**
1	Martes	7:00am	9:00am	SC003-P404
2	Jueves	7:00am	9:00am	SA027-P503

Una vez están ofertadas todas las asignaturas y clases, se procede a la segunda parte del proceso, la inscripción de las asignaturas por parte de los estudiantes para generar su horario de clases semestral. De esta forma, al finalizar las citas de inscripción, cada estudiante (identificado con un ID individual y su nombre completo) tiene asociado un conjunto de diferentes clases inscritas. A continuación se presenta un ejemplo de esta información:

ID estudiante 10047724	**Nombre estudiante** Ortiz Almanza, José Camilo			
Clase 1 4185	**Clase 2** 22586	**Clase 3** 4190	**Clase 4** 4196	**Clase 5** 4085

El sistema SAE provee las herramientas necesarias para llevar a cabo los procesos aquí descritos, sin embargo, tiene un par de complicaciones. La primera tiene que ver con la cantidad de información que el sistema debe almacenar para cada uno de los procesos. Lo que se ha descrito aquí son los datos mínimos requeridos, sin embargo, por la interacción con los otros subsistemas, la información almacenada incluye bastantes más datos de los necesarios. La segunda es que el sistema puede llegar a ser algo restrictivo si se desean realizar otros análisis interesantes sobre esta información, como por ejemplo proyecciones de inscripciones de asignaturas en semestres futuros, o la forma en que los estudiantes se relacionan a través de las clases inscritas. De esta forma, el Departamento de Ingeniería de Sistemas está considerando desarrollar un sencillo sistema alterno, que pueda alimentarse con la información proveída por el SIU, y que permita explorar la información en otras maneras interesantes.

1.1 Descripción de la información de entrada

La información proveída por el SIU/SAE con respecto al proceso de programación de asignaturas e inscripción de estudiantes para un semestre particular está contenida en dos archivos de texto plano separados por comas (`.csv`): `listaClasesAsignacionAula-`*`IDsem`*`.csv` y `listaClasesInscritasEstud-`*`IDsem`*`.csv`; donde *`IDsem`* representa el identificador de 4 dígitos del semestre con el que se va a trabajar. A continuación se describe la información incluída en cada uno de los archivos.

`listaClasesAsignacionAula-`*`IDsem`*`.csv`
- `Ccl Lvo`: identificador del ciclo lectivo (semestre)
- `Grado`: nivel académico (pregrado, posgrado)
- `Org Acad`: organización académica (identificador departamento o facultad)
- `Descr Fmal`: descripción formal de la organización académica (nombre)
- `Catálogo`: identificador del catálogo de asignaturas
- `ID Curso`: identificador de la asignatura
- `Asignatura`: nombre de la asignatura
- `Nº Clase`: identificador de la clase
- `Sección`: consecutivo de la clase dentro de la asignatura
- `Sesión`: tipo de la sesión
- `Componente`: tipo de componente de la sesión dentro de la asignatura
- `Estado Clase`: estado actual de la clase
- `Est Inscr`: estado actual de las inscripciones para la clase
- `Lun`: si la clase se ofrece o no el día lunes
- `Mart`: si la clase se ofrece o no el día martes
- `Miérc`: si la clase se ofrece o no el día miércoles
- `Jue`: si la clase se ofrece o no el día jueves
- `Vier`: si la clase se ofrece o no el día viernes
- `Sáb`: si la clase se ofrece o no el día sábado
- `Dom`: si la clase se ofrece o no el día domingo

- Hora Inicio Clase: hora de inicio de la clase
- Hora Fin Clase: hora de finalización de la clase
- F Inicial: fecha de inicio de la clase
- F Final: fecha de finalización de la clase
- ID Instalación: identificador del salón asignado a la clase
- Capacidad Aula Asignada: capacidad en puestos de trabajo del salón asignado a la clase
- Capacidad Aula Solicitada: capacidad en puestos de trabajo solicitada para la clase
- Capacidad Inscripción: capacidad máxima de inscripciones para la clase
- Descr: descripción de la inscripción
- Tot Inscr: cantidad de estudiantes inscritos en la clase

listaClasesInscritasEstud-*IDsem*.csv
- Ciclo Lectivo: identificador del ciclo lectivo (semestre)
- Cd Actual d Baja d Ciclo: indicador de baja del estudiante
- ID Estudiante: identificador único del estudiante
- Nombre Estudiante: apellidos y nombres del estudiante
- Grado Académico: nivel académico del estudiante (pregrado, posgrado)
- Grado: indicador de grado del estudiante
- Cd Prog Acad Base: indicador del programa académico base del estudiante
- Cd Prog Acad: indicador del programa académico del estudiante
- Nombre Actual Prog Acad: nombre completo del programa académico
- ID Curso: identificador de la asignatura inscrita
- Nº Clase: identificador de la clase inscrita
- Estado Actual d No. Clase: estado actual de la clase inscrita
- Nombre del Curso: nombre de la asignatura inscrita
- Componente d Curso: tipo de componente de la clase
- ID Org Acad d Curso: identificador de la organización académica que ofrece la asignatura
- Nombre Actual Org Acad d Curso: nombre de la organización académica
- Cred Inscritos: créditos actuales de la asignatura
- Calif Oficial Actual: calificación obtenida para la asignatura
- Sistema Calif: indicador del sistema de calificación de la asignatura
- Estado Actual d Inscrip Clase: estado de la inscripción del estudiante
- F Últ Inscr: fecha en la que se realizó la inscripción
- F Últ Baja: fecha en la que se le dió de baja al estudiante de la asignatura

2 Descripción del proyecto

El objetivo del presente proyecto es construir un sistema de apoyo para la programación de asignaturas y la inscripción de estudiantes en el Departamento de Ingeniería de Sistemas, a partir de la información proveída por el sistema SIU/SAE. El sistema se implementará como una aplicación que recibe comandos textuales, agrupados en componentes con funcionalidades específicas. A continuación se describen los componentes individuales que conforman el proyecto.

2.1 Componente 1: organización de la información.

Objetivo: Los algoritmos implementados en este componente servirán para gestionar la inicialización del sistema y verificar la adecuada conexión de los datos. Este componente se implementará con las siguientes funciones:

- **comando:** `cargar_info_asign` *listaClasesAsignacionAula-IDsem.csv*
  posibles salidas en pantalla:
  (Semestre ya inicializado) `Información de asignaturas y clases para el semestre` *IDsem* `ya ha sido cargada.`
  (Archivo no existe) `El archivo` *listaClasesAsignacionAula-IDsem.csv* `no existe o es ilegible.`
  (Resultado exitoso) `Información de asignaturas y clases para el semestre` *IDsem* `ha sido cargada exitosamente.`

descripción: Carga en el sistema la información de asignaturas y clases programadas para un semestre específico. El comando debe estructurar la información a partir del archivo de forma que sea fácil recuperar los datos mínimos posteriormente. La información debe ser verificada para no incluir registros repetidos o incompletos.

- **comando:** `cargar_info_estud` *`listaClasesInscritasEstud-IDsem.csv`*
  posibles salidas en pantalla:
  (Semestre ya inicializado) `Información de inscripciones de estudiantes para el semestre` *`IDsem`* `ya ha sido cargada.`
  (Semestre no coincide) `Semestre` *`IDsem`* `no coincide con los datos de asignaturas y clases ya cargados.`
  (Archivo no existe) `El archivo` *`listaClasesInscritasEstud-IDsem.csv`* `no existe o es ilegible.`
  (Resultado exitoso) `Información de inscripciones de estudiantes para el semestre` *`IDsem`* `ha sido cargada exitosamente.`
  descripción: Carga en el sistema la información de inscripciones de estudiantes para un semestre específico. El semestre debe coincidir con aquel para el cual ya se cargó la información de asignaturas y clases programadas, para evitar problemas de conexión de la información. El comando debe estructurar la información a partir del archivo de forma que sea fácil recuperar los datos mínimos posteriormente. Además, la información debe ser verificada para no incluir registros repetidos o incompletos.

- **comando:** `horario_estud` *`ID_estud`*
  posibles salidas en pantalla:
  (Información incompleta) `La información necesaria (programación de asignaturas, inscripciones de estudiantes) no ha sido cargada completamente.`
  (Estudiante no existe) `El estudiante con identificación` *`ID_estud`* `no está registrado en el sistema.`
  (Resultado exitoso) `El horario semanal para el estudiante` *`nombre_completo_estud`* `es:`
  descripción: El comando permite conocer el horario de clases inscrito por un estudiante, discriminado por los días de la semana. Para cada día, deben aparecer en orden cronológico las clases de las asignaturas, indicando el salón donde se llevarán a cabo. El comando debe verificar que el estudiante tenga información cargada en el sistema para poder generar el horario de clases.

- **comando:** `salir`
  posibles salidas en pantalla:
  (No tiene salida por pantalla)
  descripción: Termina la ejecución de la aplicación.

2.2 Componente 2: asignaturas y prerrequisitos.

Objetivo: Los algoritmos implementados en este componente servirán para administrar las asignaturas y sus prerrequisitos, y así analizar posibles escenarios futuros de acuerdo a esta información. Este componente se implementará con las siguientes funciones:

- **comando:** `cargar_info_prerreq` *`listaPrerrequisitos.csv`*
  posibles salidas en pantalla:
  (Prerrequisitos ya inicializados) `Información de prerrequisitos para las asignaturas ya ha sido cargada.`
  (Información incompleta) `La información necesaria (programación de asignaturas) no ha sido cargada completamente.`
  (Archivo no existe) `El archivo` *`listaPrerrequisitos.csv`* `no existe o es ilegible.`
  (Resultado exitoso) `Información de asignaturas y sus prerrequisitos ha sido cargada exitosamente.`
  descripción: Registra en el sistema la información de los prerrequisitos para cada asignatura a partir del archivo dado, en el cual se registran pares de identificadores de asignaturas (asignatura - prerrequisito). La información de asignaturas y requisitos se encuentra en el documento oficial de la Dirección de Carrera de Ingeniería de Sistemas (http://ingenieria.javeriana.edu.co/documents/2838900/2841579/Pre-Requisitos_CarreraIngSistemas(Act-sep-2016_v0).pdf).

- **comando:** `proy_demanda` *ID_asig*
  posibles salidas en pantalla:
  (Información incompleta) `La información necesaria (programación de asignaturas, prerrequisitos, inscripciones de estudiantes) no ha sido cargada completamente.`
  (Asignatura no existe) `La asignatura con identificador` *ID_asig* `no está registrada en el sistema.`
  (Resultado exitoso) `La demanda proyectada para la asignatura` *nombre_asignatura* `es:`
  (Resultado adverso) `No existe suficiente información para calcular la demanda proyectada para la asignatura` *nombre_asignatura* `.`
  descripción: El comando permite calcular la demanda proyectada para la asignatura indicada, a partir de la información actual de estudiantes inscritos en las asignaturas prerrequisito. Para esto, se requiere calcular el árbol de prerrequisitos de la asignatura dada, que almacene asignaturas e inscritos en cada nodo, y que conecte los nodos de acuerdo a los prerrequisitos ya registrados con el comando `cargar_info_prerreq`. La demanda proyectada para cada semestre futuro dependerá de la cantidad de inscritos actuales en cada una de las asignaturas prerrequisito, teniendo en cuenta la distancia sobre el árbol (en semestres) entre la asignatura actual y cada prerrequisito.

- **comando:** `proy_semestres` *ID_estud*
  salida en pantalla:
  (Información incompleta) `La información necesaria (programación de asignaturas, prerrequisitos, inscripciones de estudiantes) no ha sido cargada completamente.`
  (Estudiante no existe) `El estudiante con identificación` *ID_estud* `no está registrado en el sistema.`
  (Resultado exitoso) `Las asignaturas proyectadas para el estudiante` *nombre_estudiante* `son:`
  (Resultado adverso) `No existe suficiente información para generar la proyección de asignaturas para el estudiante` *nombre_estudiante* `.`
  descripción: El comando permite generar una proyección de las asignaturas a cursas en semestres futuros por el estudiante dado. Para esto, se utilizan las asignaturas actualmente inscritas por el estudiante, y para cada una de ellas se genera un árbol de posibles asignaturas futuras, de acuerdo a la información de prerrequisitos ya registrada con el comando `cargar_info_prerreq`. Luego, se combina la información de estos árboles para informar las posibles asignaturas a cursar por el estudiante en cada semestre futuro.

2.3 Componente 3: red social.

Objetivo: Los algoritmos implementados en este componente servirán para generar la red social de los estudiantes inscritos en un semestre, y así analizar conexiones entre ellos. Este componente se implementará con las siguientes funciones:

- **comando:** `red_social`
  salida en pantalla:
  (Información incompleta) `La información necesaria (programación de asignaturas, inscripciones de estudiantes) no ha sido cargada completamente.`
  (Resultado exitoso) `La red social de los estudiantes ha sido generada exitosamente.`
  descripción: Utilizando la información de asignaturas programadas e inscripciones de estudiantes, el sistema genera un grafo que representa la red social del semestre actual, donde cada estudiante se conecta a otros si y sólo sí se encuentran cursando la misma clase (son compañeros de asignatura y de clase).

- **comando:** `grados_separacion` *ID_estud_1 ID_estud_2*
  salida en pantalla:
  (Red social no generada) `La red social de los estudiantes aún no ha sido generada.`
  (Estudiante no existe) `El estudiante con identificación` *ID_estud_1* `(y/o` *ID_estud_2*`) no está registrado en el sistema.`
  (Resultado exitoso 1) `El estudiante` *nombre_estud_1* `conoce al estudiante` *nombre_estud_2* `de forma directa.`
  (Resultado exitoso 2) `El estudiante` *nombre_estud_1* `conoce al estudiante` *nombre_estud_2* `a través de` *n* `conocidos intermedios, los cuales son:`
  (Resultado adverso) `El estudiante` *nombre_estud_1* `no conoce al estudiante` *nombre_estud_2* `.`
  descripción: Dados los identificadores de dos estudiantes, el comando inspecciona la red social generada

para el semestre en busca de los conocidos conectados que permitan llegar de un estudiante a otro, confirmando la hipótesis de los "seis grados de separación". El comando debe verificar que los estudiantes estén cargados en el sistema y que el grafo con la red social ya haya sido generado. Si los estudiantes están conectados de forma directa, o no existe conexión alguna entre ellos, el sistema debe informar la situación con el mensaje apropiado.

2.4 Interacción con el sistema

La interfaz de la aplicación a construir debe ser una consola interactiva donde los comandos correspondientes a los componentes serán tecleados por el usuario, de la misma forma que se trabaja en la terminal o consola del sistema operativo. El indicador de línea de comando debe ser el caracter '`$`'. Se debe incluir el comando '`ayuda`' para indicar una lista de los comandos disponibles en el momento. Así mismo, para cada comando se debe incluir una ayuda de uso que indique la forma correcta de hacer el llamado, es decir, el comando '`ayuda` *`comando`*' debe existir. Cada comando debe presentar en pantalla los mensajes de resultado especificados antes, además de otros mensajes que se consideren necesarios. Los comandos de los diferentes componentes deben ensamblarse en un único sistema (es decir, funcionan todos dentro de un mismo programa, no como programas independientes por componentes).

3 Evaluación

Las entregas se harán en la correspondiente actividad de UVirtual o Moodle, hasta la media noche del día anterior al indicado para la entrega. Se debe entregar un archivo comprimido (únicos formatos aceptados: .zip, .tar, .tar.gz, .tar.bz2, .tgz) que contenga dentro de un mismo directorio (**sin estructura de carpetas interna**) los documentos (único formato aceptado: .pdf) y el código fuente (.h, .hxx, .hpp, .c, .cxx, .cpp). Si la entrega contiene archivos en cualquier otro formato, será descartada y no será evaluada, es decir, la nota definitiva de la entrega será de **0 (cero) sobre 5 (cinco)**.

3.1 Entrega 1 (jueves 16 y viernes 17 de marzo de 2017)

Componente 1 completo y funcional. Esta entrega tendrá una sustentación durante la correspondiente sesión de clase. Esta entrega se compone de:

- (30%) Documento de diseño. El documento de diseño debe seguir las pautas de ingeniería que usted ya conoce: descripción de entradas, salidas y condiciones para el procedimiento principal y las operaciones auxiliares. Para la descripción de los TADs utilizados, debe seguirse la plantilla definida en clase. Además, se exigirá un(os) esquemático(s) que resuma(n) el(los) problema(s) y su(s) solución(ones) gráficamente.
- (55%) Código fuente compilable en el compilador `gnu-g++` (versión 4.0.0, como mínimo). Este porcentaje de la entrega será un promedio de la evaluación de cada comando.
- (15%) Sustentación con participación de todos los miembros del grupo.

3.2 Entrega 2 (jueves 27 y viernes 28 de abril de 2017)

Componentes 1 y 2 completos y funcionales. Esta entrega tendrá una sustentación durante la correspondiente sesión de clase. Esta entrega se compone de:

- (15%) Completar la funcionalidad que aún no haya sido desarrollada de la primera entrega. Se debe generar un acta de evaluación de la entrega anterior que detalle los elementos faltantes y cómo se completaron para la segunda entrega.
- (25%) Documento de diseño. El documento de diseño debe seguir las pautas de ingeniería que usted ya conoce: descripción de entradas, salidas y condiciones para el procedimiento principal y las operaciones auxiliares. Para la descripción de los TADs utilizados, debe seguirse la plantilla definida en clase. Además, se exigirá un(os) esquemático(s) que resuma(n) el(los) problema(s) y su(s) solución(ones) gráficamente.
- (45%) Código fuente compilable en el compilador `gnu-g++` (versión 4.0.0, como mínimo). Este porcentaje de la entrega será un promedio de la evaluación de cada comando.

- (15%) Sustentación con participación de todos los miembros del grupo.

3.3 Entrega 3 (jueves 1 y viernes 2 de junio de 2017)

Componentes 1, 2 y 3 completos y funcionales. Esta entrega tendrá una sustentación (del proyecto completo) durante el jueves 1 o el viernes 2 de junio de 2017 entre las 8am y las 6pm. Esta entrega se compone de:

- (15%) Completar la funcionalidad que aún no haya sido desarrollada de la primera y segunda entregas. Se debe generar un acta de evaluación de la entrega anterior que detalle los elementos faltantes y cómo se completaron para la tercera entrega.
- (25%) Documento de diseño. El documento de diseño debe seguir las pautas de ingeniería que usted ya conoce: descripción de entradas, salidas y condiciones para el procedimiento principal y las operaciones auxiliares. Para la descripción de los TADs utilizados, debe seguirse la plantilla definida en clase. Además, se exigirá un(os) esquemático(s) que resuma(n) el(los) problema(s) y su(s) solución(ones) gráficamente.
- (45%) Código fuente compilable en el compilador `gnu-g++` (versión 4.0.0, como mínimo). Este porcentaje de la entrega será un promedio de la evaluación de cada comando.
- (15%) Sustentación con participación de todos los miembros del grupo.