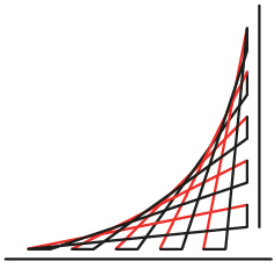


ESCUELA
COLOMBIANA
DE INGENIERÍA
JULIO GARAVITO

DISEÑO DETALLADO DEL SOFTWARE

AECI – ASOCIACION EGRESADO ESCUELA COLOMBIANA DE INGENIERIA

Autores:
Tiffany Estupiñan Londoño
Cristian Camilo Pinzón Hernández
Ricardo Andrés Pinto Rico
Carlos Alberto Ramírez Otero
Martin José Hernández Medina

Co-autor/Asesor:
Profesor Héctor Cadavid R.

PREFACIO

El presente Documento describe el diseño detallado de software del módulo diseñado para AECI – Asociación Egresado Escuela Colombiana de Ingeniería, cuyo objetivo principal es ofrecer una aplicación web de gestión de los usuarios que hagan parte de la asociación.

Alcance

El módulo permite registrar y consultar los egresados y estudiantes que estén a punto de culminar sus estudios dentro de la Escuela Colombiana de Ingeniería con el objetivo de mantener la comunicación con ellos y ofrecer distintos servicios por medio de una afiliación gestionada por el administrador.

TABLA DE CONTENIDOS

1 Definición del Software

1.1 Visión del producto

Actualmente AECI tiene una página web con la información necesaria para que una persona que desea afiliarse conozca los servicios y el proceso necesario para pertenecer a la misma, nuestro software permite gestionar desde distintas vistas (Egresado, Estudiante y Administrador) las funcionalidades que puede realizar.

El administrador tiene un rol importante dentro de la asociación quien puede aceptar o rechazar las solicitudes que ingresan al sistema y a su vez revisar los pagos que hagan los afiliados para determinar si siguen o no en la asociación.

Si el estudiante y el egresado están registrados en el sistema, pueden gestionar sus pagos o renovaciones de su estado de afiliación y generar certificados de afiliación, así como estar seguros de su información por el sistema de seguridad con el que opera el software.

1.2 Descripción del contexto en el cual se utilizará el producto

El producto se utilizará para las personas que quieren estar o pertenezcan a la asociación así mismo los administradores que podrán operar la plataforma.

1.3 Historias de usuario

Las historias de usuario consideradas fueron:

- Solicitud de afiliación.
- Procesar solicitudes de afiliación.
- Generación de certificados de afiliación.
- Reporte de afiliaciones por vencerse.
- Registrar pago afiliación o renovación.
- Procesar pagos afiliaciones o renovaciones.

1.4 Alcance

El módulo debe crear solicitudes de afiliación, poder aceptarlas o rechazarlas juntos a los pagos que realicen los egresados por medio de imágenes adjuntadas al sistema, permite saber las personas que pronto termina de vencer su afiliación para enviar recordatorios y la generación de certificados de afiliación.

1.5 Supuestos

Se supone que:

- Una vez registrado el afiliado, su usuario y contraseña serán su documento de identidad.
- Un afiliado debe tener al menos un correo electrónico de contacto.
- Por el momento existe un administrador.

1.6 Dependencias

Este módulo usa los módulos:

- com.sun.faces: Utiliza un conjunto de etiquetas que soporta primefaces.
- org.glassfish: Implementa tecnologías definidas en la plataforma Java EE necesarias para distintas funcionalidades.
- Org.primefaces: para utilizar un conjunto de etiquetas que soporta primefaces.
- log4j: Crea log de los errores que pueden salir en la aplicación de java.
- Mysql: para permitir la conexión con la base de datos.
- Org.primefaces.themes: para aplicar tema para las vistas de la aplicacion.
- Pax-swissbox-optional-jcl: sirve para ayudar a maven a embeber el bunde
- Com.jolbox: para administrar el pool de conexiones a la base de datos
- Junit: para ejecutar pruebas unitarias
  Javax:mysql-connector-java
- Com.h2database: para crear bases de datos embebidas para las pruebas
- Org.apache.shiro: para ejecutar la seguridad, manejo de perfiles.

1.7 Metodología de Prueba

Las pruebas se encuentran en su propio paquete de pruebas, para hacerlas se definieron las clases de equivalencia correspondientes encontradas como comentarios en cada una de las pruebas:

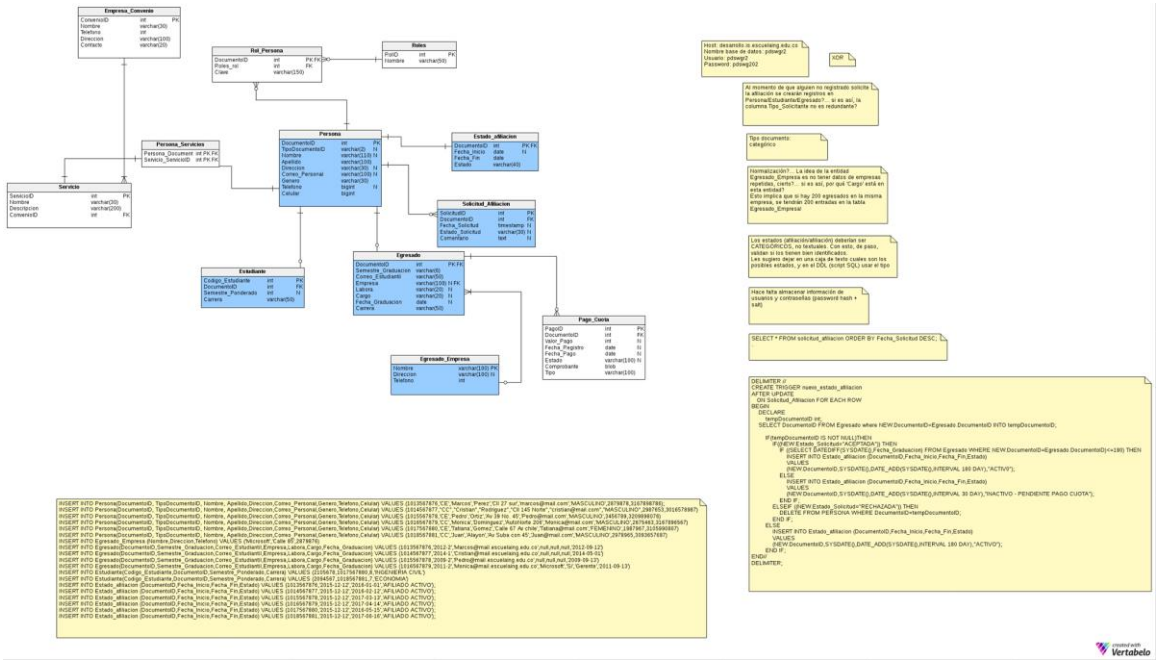
- Reporte por vencerse
- Pago de la afiliación

1.8 Estándares seguidos

Los estándares seguidos fueron los estándares de codificación de Java, la nomenclatura de paquetes, variables, nombre de interfaces, nombres de clases, métodos, constantes, comentarios, declaraciones, sentencias.

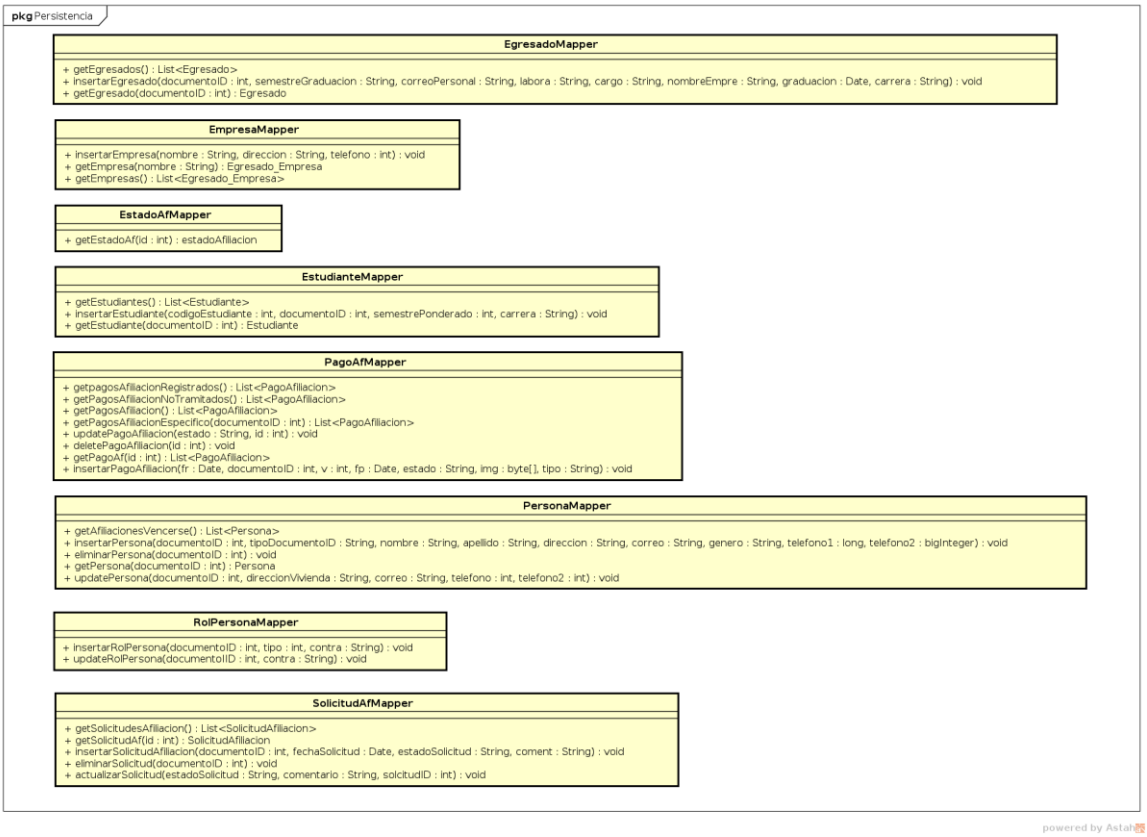
2 Diseño Detallado

2.1 Modelo relacional



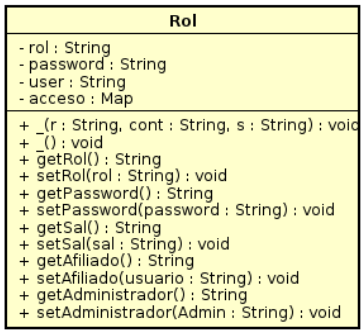
2.2 Diseño por paquetes/capas

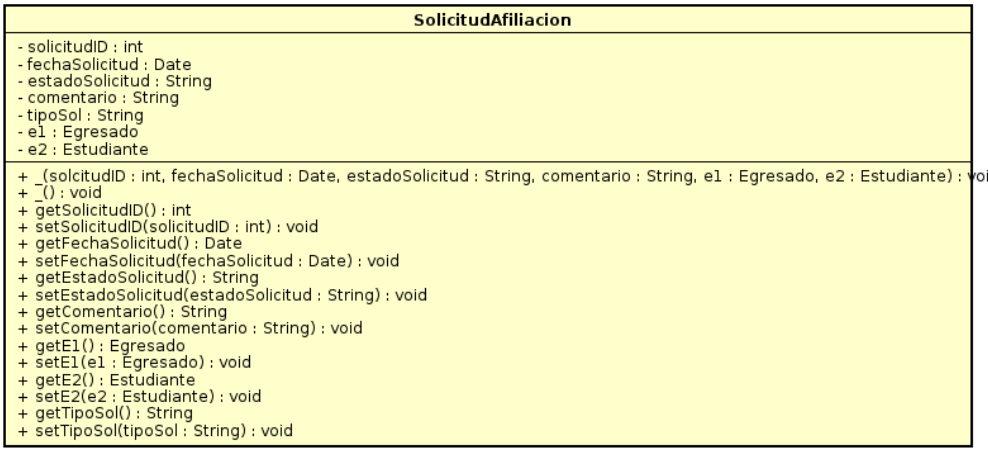
2.2.1 Capa de persistencia

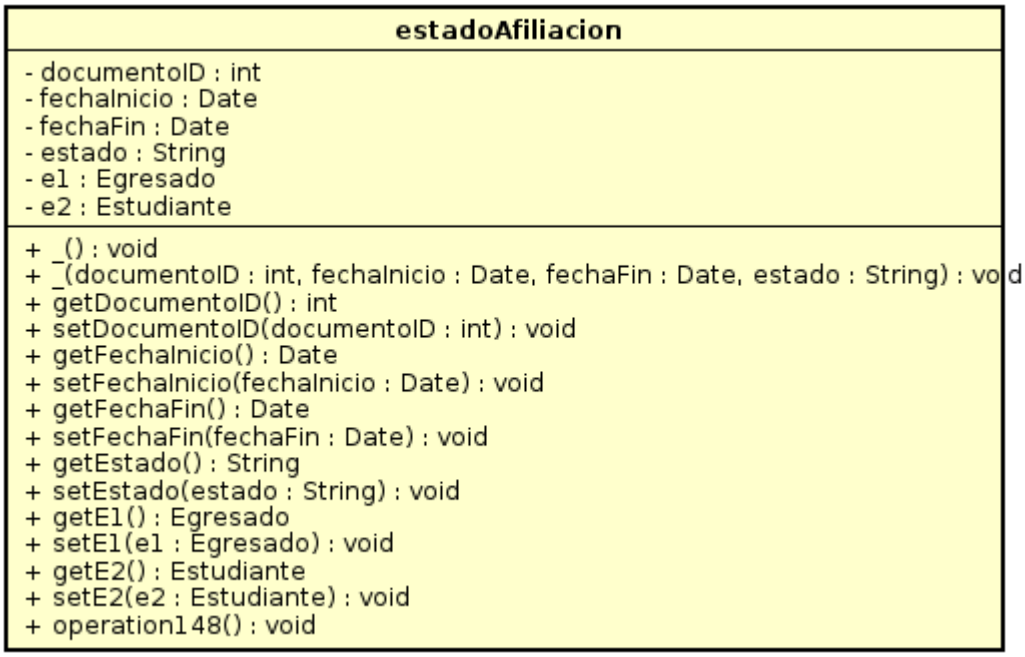


2.2.2 Capa lógica

Entities


Rol
- rol : String
- password : String
- user : String
- acceso : Map
+ _(r : String, cont : String, s : String) : voic
+ _() : void
+ getRol() : String
+ setRol(rol : String) : void
+ getPassword() : String
+ setPassword(password : String) : void
+ getSal() : String
+ setSal(sal : String) : void
+ getAfiliado() : String
+ setAfiliado(usuario : String) : void
+ getAdministrador() : String
+ setAdministrador(Admin : String) : void


SolicitudAfiliacion
- solicitudID : int
- fechaSolicitud : Date
- estadoSolicitud : String
- comentario : String
- tipoSol : String
- e1 : Egresado
- e2 : Estudiante
+ _(solcitudID : int, fechaSolicitud : Date, estadoSolicitud : String, comentario : String, e1 : Egresado, e2 : Estudiante) : void
+ _() : void
+ getSolicitudID() : int
+ setSolicitudID(solicitudID : int) : void
+ getFechaSolicitud() : Date
+ setFechaSolicitud(fechaSolicitud : Date) : void
+ getEstadoSolicitud() : String
+ setEstadoSolicitud(estadoSolicitud : String) : void
+ getComentario() : String
+ setComentario(comentario : String) : void
+ getE1() : Egresado
+ setE1(e1 : Egresado) : void
+ getE2() : Estudiante
+ setE2(e2 : Estudiante) : void
+ getTipoSol() : String
+ setTipoSol(tipoSol : String) : void


estadoAfiliacion
- documentoID : int
- fechaInicio : Date
- fechaFin : Date
- estado : String
- e1 : Egresado
- e2 : Estudiante
+ _() : void
+ _(documentoID : int, fechaInicio : Date, fechaFin : Date, estado : String) : vo d
+ getDocumentoID() : int
+ setDocumentoID(documentoID : int) : void
+ getFechaInicio() : Date
+ setFechaInicio(fechaInicio : Date) : void
+ getFechaFin() : Date
+ setFechaFin(fechaFin : Date) : void
+ getEstado() : String
+ setEstado(estado : String) : void
+ getE1() : Egresado
+ setE1(e1 : Egresado) : void
+ getE2() : Estudiante
+ setE2(e2 : Estudiante) : void
+ operation148() : void

Servicios:


ServiciosSAGECI
- Instance : ServiciosSAGECI
+ _() : void
+ getInstance() : ServiciosSAGECI
+ agregarEstudiante(codigoEstudiante : int, documentoID : int, semestrePonderado : int, telefono1 : int, telefono2 : bigInteger, tipoDocumentoID : String, nombre : String, apellido : String, direccion : String, carrera : String, correo : String, genero : String, rol : Rol) : void
+ agregarEgresado(documentoID : int, telefono1 : bigInteger, telefono2 : bigInteger, tipoDocumentoID : String, nombre : String, apellido : String, direccion : String, correo : String, genero : String, rol : Rol, semestreGrado : String, correoPersonal : String, cargo : String, labora : String, egresadoEmpresa : Egresado_Empresa, fechaGraduacion : Date, carrera : String) : void
+ agregarRolPersona(documentoID : int, tipo : int, contra : String) : void
+ consultarSolicitudAfiliaciones() : List<SolicitudAfiliacion>
+ consultarSolicitudAfiliacion(solicitudID : int) : SolicitudAfiliacion
+ registrarNuevaSolicitud(s : SolicitudAfiliacion) : void
+ agregarComentarioSolicitud(solicitudID : int, c : String) : void
+ consultarEstudiante(documentoID : int) : Estudiante
+ consultarEgresado(documentoID : int) : Egresado
+ consultarPersona(documentoID : int) : Persona
+ actualizarSolicitudAfiliacion(s : SolicitudAfiliacion) : void
+ consultarSolicitudAfiliacionesVencidas() : List<Persona>
+ consultarEstadoAfiliacion(documentoID : int) : estadoAfiliacion
+ consultarPagosAfiliacionesNoTramitados() : List<PagoAfiliacion>
+ consultarPagosAfiliacionesRegistrados() : List<PagoAfiliacion>
+ consultarPagosAfiliaciones() : List<PagoAfiliacion>
+ consultarPagosAfiliacionesEspecifico(id : int) : List<PagoAfiliacion>
+ actualizarPagoAfiliacion(s : PagoAfiliacion) : void
+ borrarPagoAfiliacion(s : PagoAfiliacion) : void
+ agregarPagoAfiliacion(pg : PagoAfiliacion) : void
+ actualizarUsuario(documentoID : int, direccionVivienda : String, correo : String, telefono : int, telefono2 : int) : void
+ actualizarContra(documentoID : int, contra : String) : void
ServiciosSAGECIDAOS
- daof : DaoFactory
- properties : Properties
+ _() : void
+ agregarEstudiante(codigoEstudiante : int, documentoID : int, semestrePonderado : int, telefono1 : int, telefono2 : bigInteger, tipoDocumentoID : String, nombre : String, apellido : String, direccion : String, carrera : String, correo : String, genero : String, rol : Rol) : void
+ agregarEgresado(documentoID : int, telefono1 : bigInteger, telefono2 : bigInteger, tipoDocumentoID : String, nombre : String, apellido : String, direccion : String, correo : String, genero : String, rol : Rol, semestreGrado : String, correoPersonal : String, cargo : String, labora : String, egresadoEmpresa : Egresado_Empresa, fechaGraduacion : Date, carrera : String) : void
+ agregarRolPersona(documentoID : int, tipo : int, contra : String) : void
+ consultarSolicitudAfiliaciones() : List<SolicitudAfiliacion>
+ consultarSolicitudAfiliacion(solicitudID : int) : SolicitudAfiliacion
+ registrarNuevaSolicitud(s : SolicitudAfiliacion) : void
+ agregarComentarioSolicitud(solicitudID : int, c : String) : void
+ consultarEstudiante(documentoID : int) : Estudiante
+ consultarEgresado(documentoID : int) : Egresado
+ consultarPersona(documentoID : int) : Persona
+ actualizarSolicitudAfiliacion(s : SolicitudAfiliacion) : void
+ consultarSolicitudAfiliacionesVencidas() : List<Persona>
+ consultarEstadoAfiliacion(documentoID : int) : estadoAfiliacion
+ consultarPagosAfiliacionesNoTramitados() : List<PagoAfiliacion>
+ consultarPagosAfiliacionesRegistrados() : List<PagoAfiliacion>
+ consultarPagosAfiliaciones() : List<PagoAfiliacion>
+ consultarPagosAfiliacionesEspecifico(id : int) : List<PagoAfiliacion>
+ actualizarPagoAfiliacion(s : PagoAfiliacion) : void
+ borrarPagoAfiliacion(s : PagoAfiliacion) : void
+ agregarPagoAfiliacion(pg : PagoAfiliacion) : void
+ actualizarUsuario(documentoID : int, direccionVivienda : String, correo : String, telefono : int, telefono2 : int) : void
+ actualizarContra(documentoID : int, contra : String) : void

DAO’S:

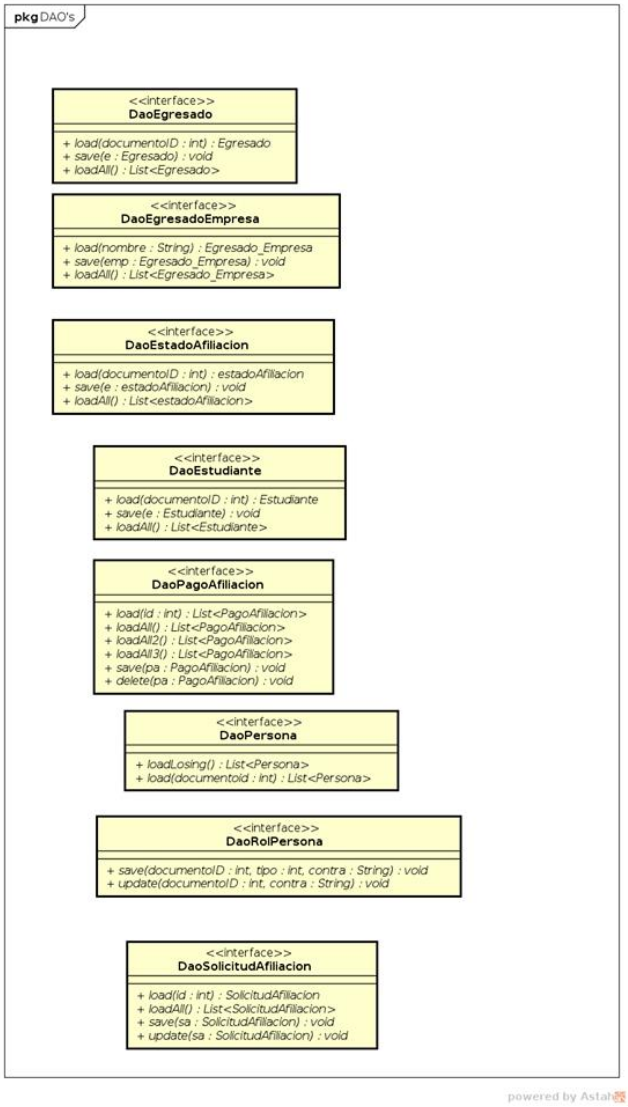

pkg DAO's
<<interface>>
DaoEgresado
+ load(documentoID : int) : Egresado
+ save(e : Egresado) : void
+ loadAll() : List<Egresado>
<<interface>>
DaoEgresadoEmpresa
+ load(nombre : String) : Egresado_Empresa
+ save(emp : Egresado_Empresa) : void
+ loadAll() : List<Egresado_Empresa>
<<interface>>
DaoEstadoAfiliacion
+ load(documentoID : int) : estadoAfiliacion
+ save(e : estadoAfiliacion) : void
+ loadAll() : List<estadoAfiliacion>
<<interface>>
DaoEstudiante
+ load(documentoID : int) : Estudiante
+ save(e : Estudiante) : void
+ loadAll() : List<Estudiante>
<<interface>>
DaoPagoAfiliacion
+ load(id : int) : List<PagoAfiliacion>
+ loadAll() : List<PagoAfiliacion>
+ loadAll2() : List<PagoAfiliacion>
+ loadAll3() : List<PagoAfiliacion>
+ save(pa : PagoAfiliacion) : void
+ delete(pa : PagoAfiliacion) : void
<<interface>>
DaoPersona
+ loadLosing() : List<Persona>
+ load(documentoId : int) : List<Persona>
<<interface>>
DaoRolPersona
+ save(documentoID : int, tipo : int, contra : String) : void
+ update(documentoID : int, contra : String) : void
<<interface>>
DaoSolicitudAfiliacion
+ load(id : int) : SolicitudAfiliacion
+ loadAll() : List<SolicitudAfiliacion>
+ save(sa : SolicitudAfiliacion) : void
+ update(sa : SolicitudAfiliacion) : void
powered by Astah

Diagrama de secuencia de Registrar Egresado

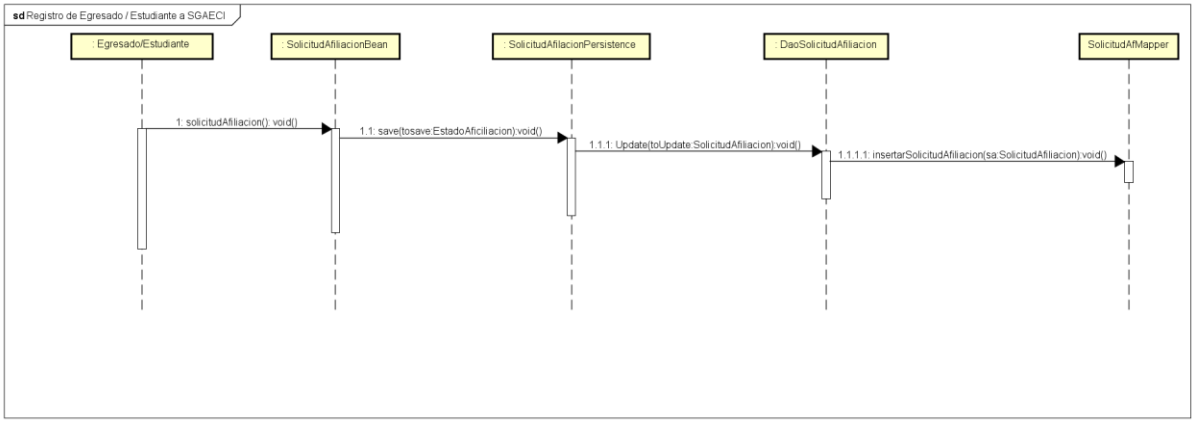
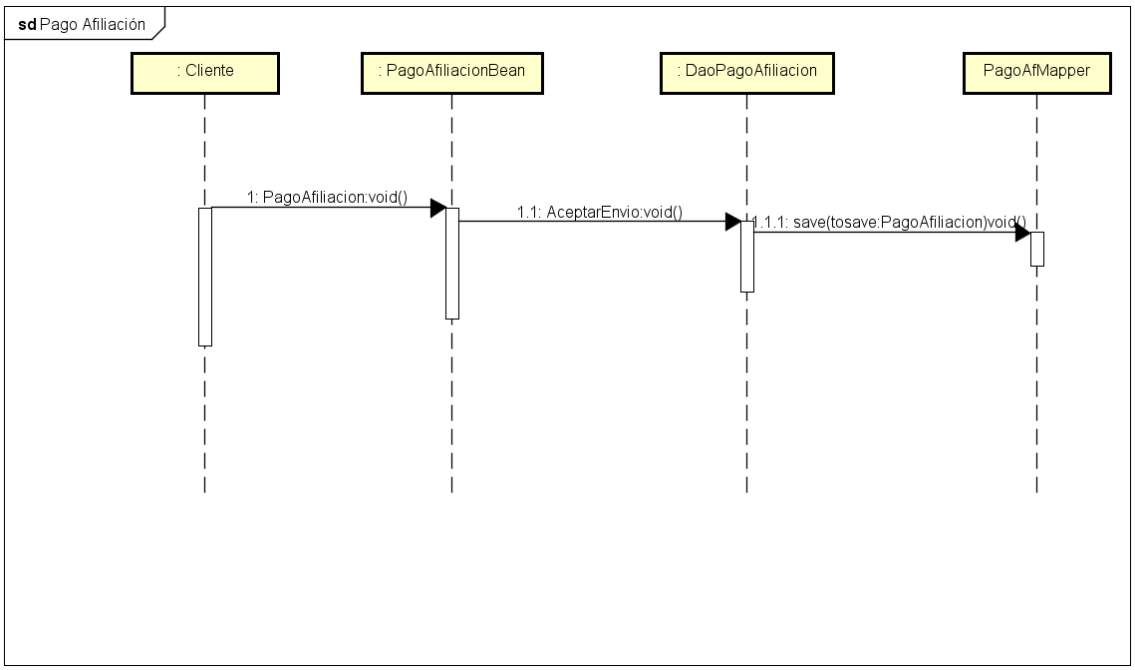
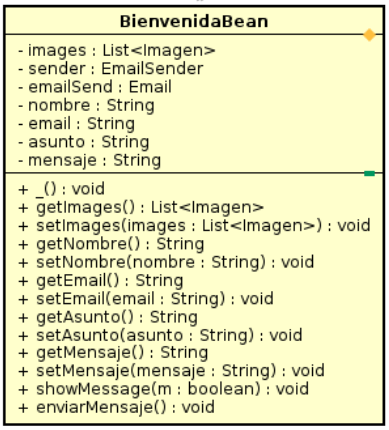
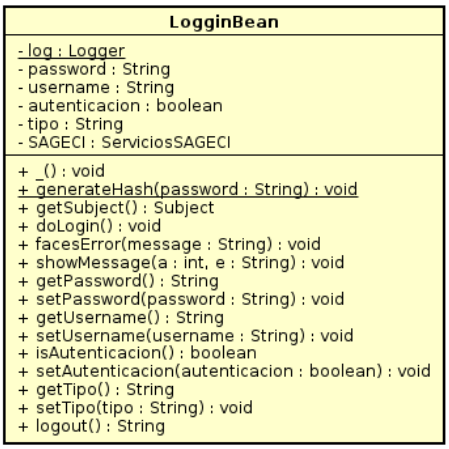


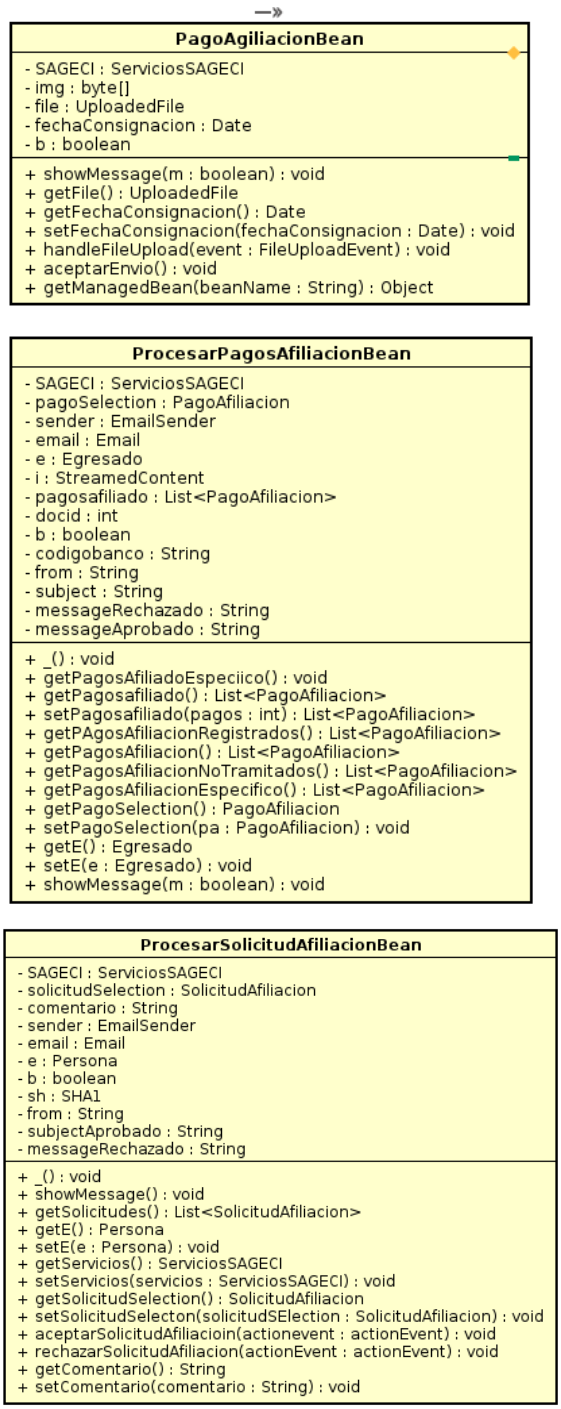
Diagrama de secuencia Pago de Afiliación:

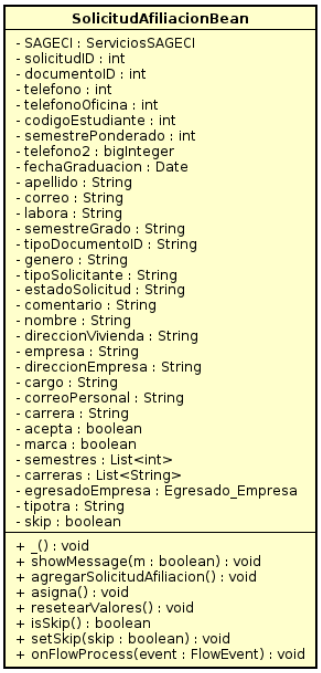


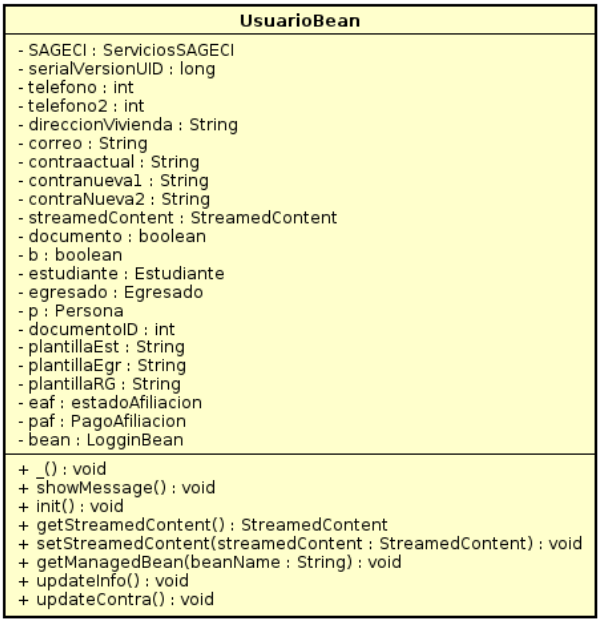
2.2.3 Capa de presentación






PagoAgiliacionBean
- SAGECI : ServiciosSAGECI
- img : byte[]
- file : UploadedFile
- fechaConsignacion : Date
- b : boolean
+ showMessage(m : boolean) : void
+ getFile() : UploadedFile
+ getFechaConsignacion() : Date
+ setFechaConsignacion(fechaConsignacion : Date) : void
+ handleFileUpload(event : FileUploadEvent) : void
+ aceptarEnvio() : void
+ getManagedBean(beanName : String) : Object
ProcesarPagosAfiliacionBean
- SAGECI : ServiciosSAGECI
- pagoSelection : PagoAfiliacion
- sender : EmailSender
- email : Email
- e : Egresado
- i : StreamedContent
- pagosafiliado : List<PagoAfiliacion>
- docid : int
- b : boolean
- codigobanco : String
- from : String
- subject : String
- messageRechazado : String
- messageAprobado : String
+ _() : void
+ getPagosAfiliadoEspeciico() : void
+ getPagosafiliado() : List<PagoAfiliacion>
+ setPagosafiliado(pagos : int) : List<PagoAfiliacion>
+ getPAgosAfiliacionRegistrados() : List<PagoAfiliacion>
+ getPagosAfiliacion() : List<PagoAfiliacion>
+ getPagosAfiliacionNoTramitados() : List<PagoAfiliacion>
+ getPagosAfiliacionEspecifico() : List<PagoAfiliacion>
+ getPagoSelection() : PagoAfiliacion
+ setPagoSelection(pa : PagoAfiliacion) : void
+ getE() : Egresado
+ setE(e : Egresado) : void
+ showMessage(m : boolean) : void
ProcesarSolicitudAfiliacionBean
- SAGECI : ServiciosSAGECI
- solicitudSelection : SolicitudAfiliacion
- comentario : String
- sender : EmailSender
- email : Email
- e : Persona
- b : boolean
- sh : SHA1
- from : String
- subjectAprobado : String
- messageRechazado : String
+ _() : void
+ showMessage() : void
+ getSolicitudes() : List<SolicitudAfiliacion>
+ getE() : Persona
+ setE(e : Persona) : void
+ getServicios() : ServiciosSAGECI
+ setServicios(servicios : ServiciosSAGECI) : void
+ getSolicitudSelection() : SolicitudAfiliacion
+ setSolicitudSelecton(solicitudSElection : SolicitudAfiliacion) : void
+ aceptarSolicitudAfiliacioin(actionevent : actionEvent) : void
+ rechazarSolicitudAfiliacion(actionEvent : actionEvent) : void
+ getComentario() : String
+ setComentario(comentario : String) : void


SolicitudAfiliacionBean
- SAGECI : ServiciosSAGECI
- solicitudID : int
- documentoID : int
- telefono : int
- telefonoOficina : int
- codigoEstudiante : int
- semestrePonderado : int
- telefono2 : bigInteger
- fechaGraduacion : Date
- apellido : String
- correo : String
- labora : String
- semestreGrado : String
- tipoDocumentoID : String
- genero : String
- tipoSolicitante : String
- estadoSolicitud : String
- comentario : String
- nombre : String
- direccionVivienda : String
- empresa : String
- direccionEmpresa : String
- cargo : String
- correoPersonal : String
- carrera : String
- acepta : boolean
- marca : boolean
- semestres : List<int>
- carreras : List<String>
- egresadoEmpresa : Egresado_Empresa
- tipotra : String
- skip : boolean
+ _() : void
+ showMessage(m : boolean) : void
+ agregarSolicitudAfiliacion() : void
+ asigna() : void
+ resetearValores() : void
+ isSkip() : boolean
+ setSkip(skip : boolean) : void
+ onFlowProcess(event : FlowEvent) : void


UsuarioBean
- SAGECI : ServiciosSAGECI
- serialVersionUID : long
- telefono : int
- telefono2 : int
- direccionVivienda : String
- correo : String
- contraactual : String
- contranueva1 : String
- contraNueva2 : String
- streamedContent : StreamedContent
- documento : boolean
- b : boolean
- estudiante : Estudiante
- egresado : Egresado
- p : Persona
- documentoID : int
- plantillaEst : String
- plantillaEgr : String
- plantillaRG : String
- eaf : estadoAfiliacion
- paf : PagoAfiliacion
- bean : LogginBean
+ _() : void
+ showMessage() : void
+ init() : void
+ getStreamedContent() : StreamedContent
+ setStreamedContent(streamedContent : StreamedContent) : void
+ getManagedBean(beanName : String) : void
+ updateInfo() : void
+ updateContra() : void

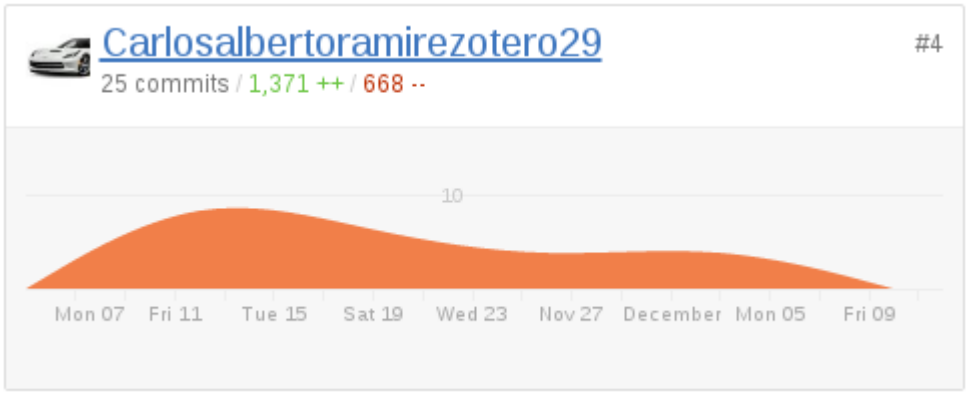

reporteVencerseBean
- SAGECI : ServiciosSAGECI
- personas : List<Persona>
- sender : EmailSender
- email : Email
- from : String
- subjectAprobado : String
+ _() : void
+ getSolicitudesAfiliacioinVencidas() : List<Persona>
+ recordatorio() : void

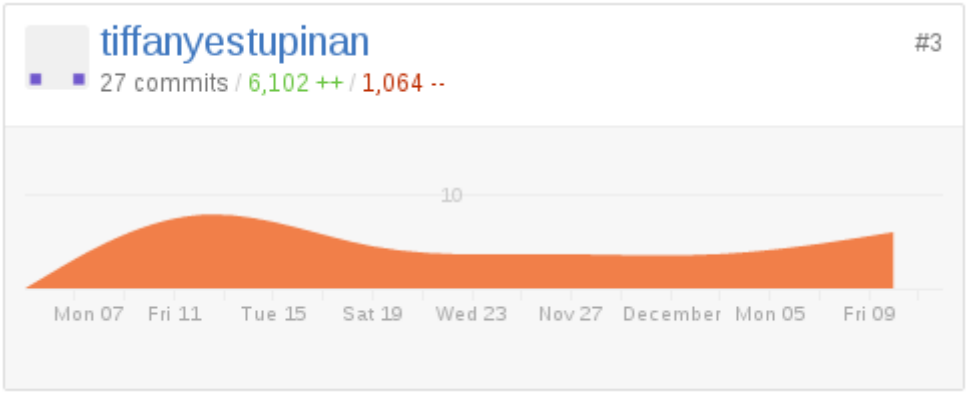
3 Descripción del proceso

3.1 Descripción del proceso de desarrollo utilizado, y de las prácticas consideradas para el mismo

Para el proceso de desarrollo se utilizó el framework de SCRUM, junto con las prácticas que este implica.

3.2 Métricas por desarrollador

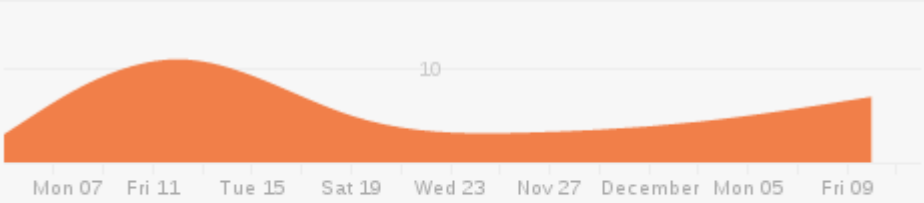




cristianpinzon97

#2

35 commits / 5,612 ++ / 1,384 --



martinjhm271

#1

37 commits / 11,851 ++ / 13,979 --