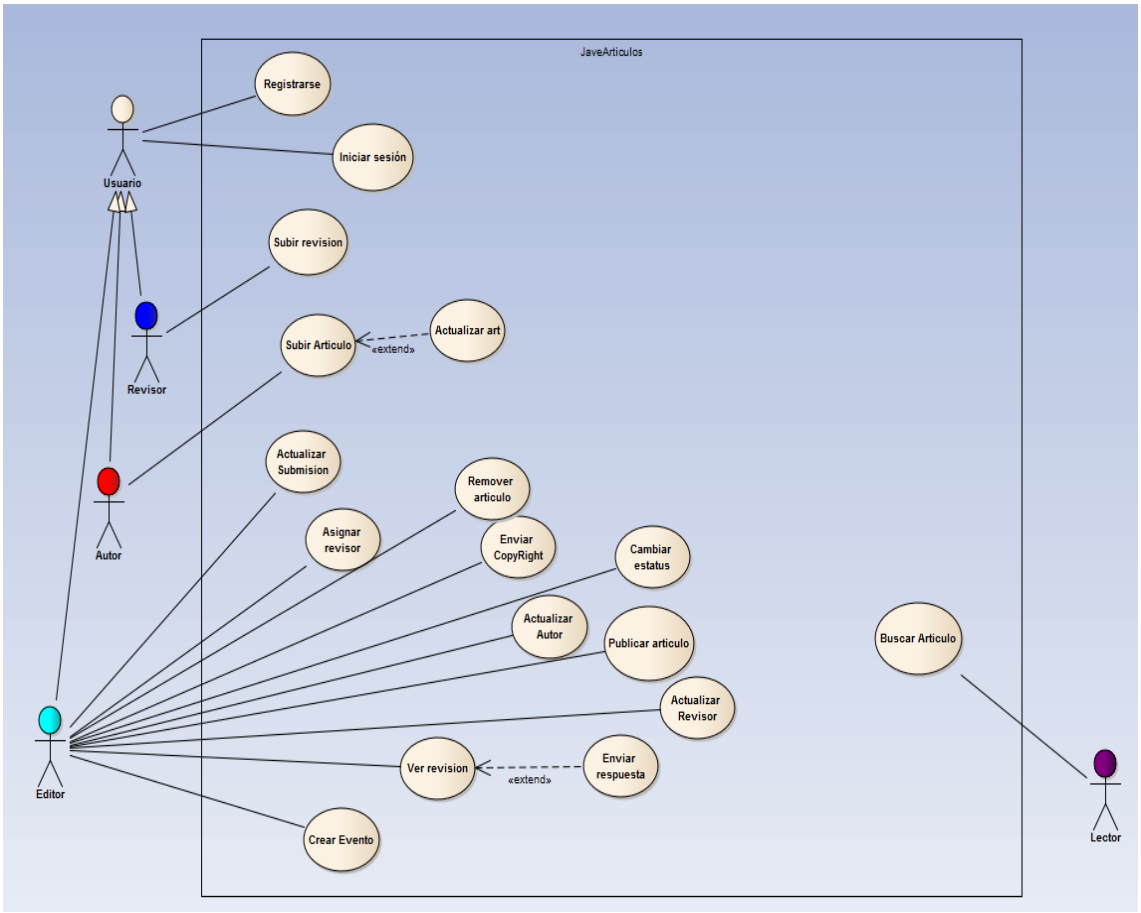
Web Services - JaveArtículos - SOAP

Alejandra Chacón - Jhonan Espejo - Pablo Gaitán

21 de Octubre de 2017

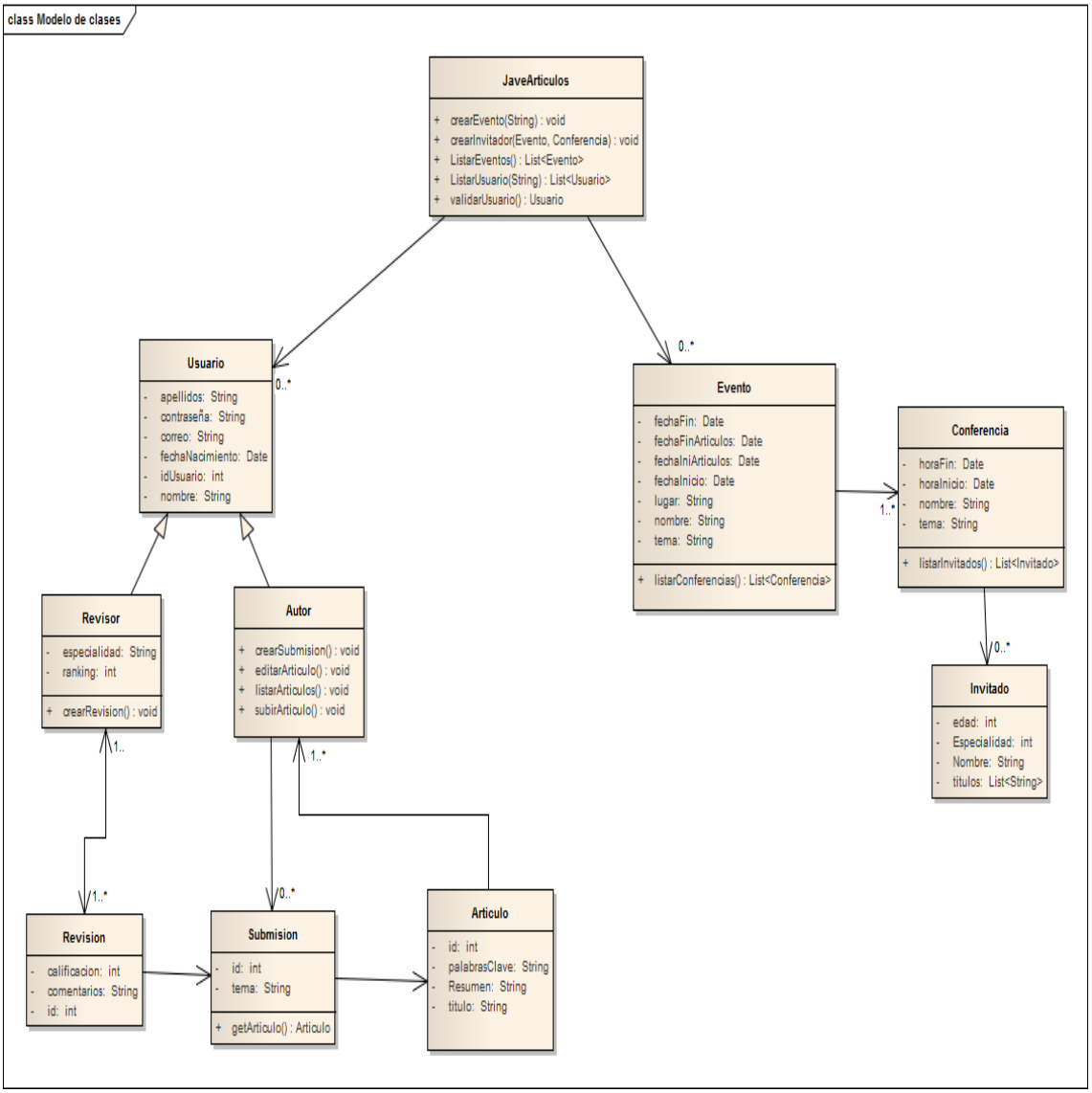
1 Diagramas

1.1 Diagrama de Casos de Uso



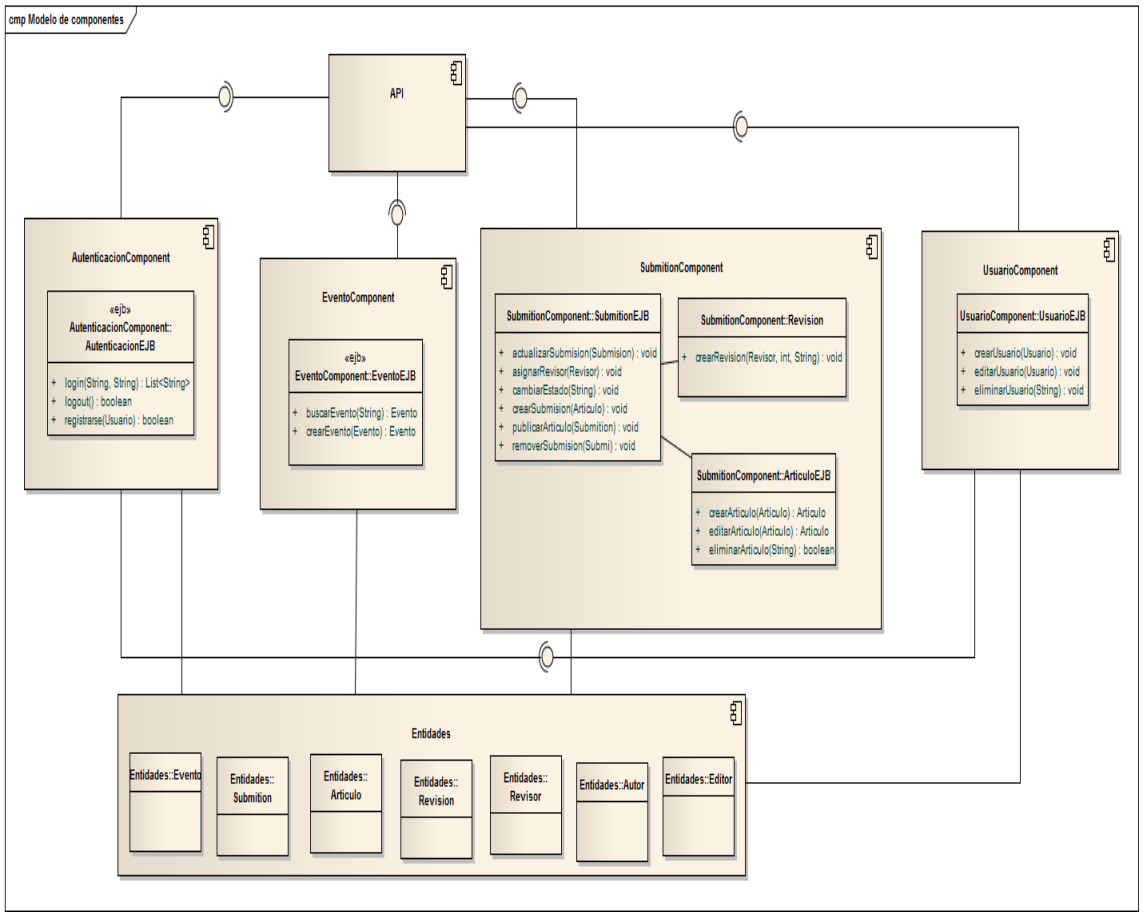
1.2 Diagrama de clases

En este diagrama se muestra las clases que compondrían el sistema de javeArticulos, como se puede ver, la idea es que hayan varios tipos de usuario con roles diferentes dentro del sistema siendo unos autores de papers/articulos, otros revisores y unos últimos editores como administradores del sistema, estos articulos están contenidos dentro de una submision asociada a unos revisores y unos eventos específicos.



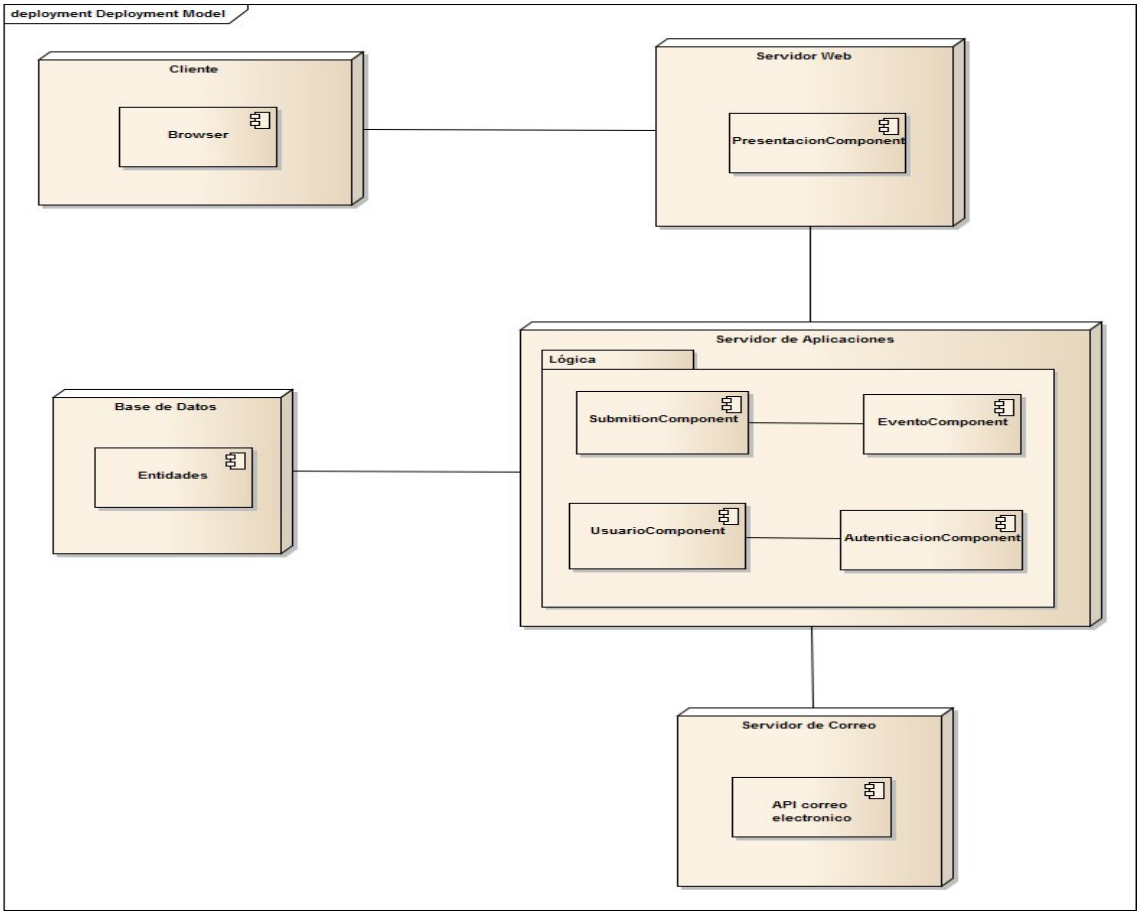
1.3 Diagrama de Componentes

Se desea implementar un paquete API donde estén encapsulados todos los servicios que implementan los diferentes componentes separados por áreas de Submisión, autenticación, usuario y evento que están en constante contacto con el paquete de entidades que manejan la persistencia con la base de datos.



1.4 Diagrama de despliegue

Dentro de este diagrama se espera tener un nodo para la parte de lógica del sistema que publica los servicios a ser consumidos por un cliente por medio de un browser, como extra de la aplicación se desea dejar en nodos separados la comunicación con la base de datos usando el protocolo jdbc y otro aparte para el manejo de los correos electrónicos que necesita el editor para informar a los autores del estado del articulo sometido a revisión.



2 Documentación

Server SOAP

Es el proyecto hecho en Java que ofrece todos los servicios del servidor para luego ser consumidos por el cliente. Este proyecto cuenta con los siguientes paquetes :

2.1 Componentes

Dentro de este paquete están las siguientes clases :

Componente Autenticación
Se compone de un *EntityManagerFactory* para crear un *EntityManager*, esta

clase contiene los siguientes métodos:

- **autenticación:** recibe un correo y una contraseña, ambos strings y a partir del *EntityManager* crear un *Query* de la clase y método *Usuario.autenticación*, configura los parámetros del *Query* y ejecuta la consulta si existe retorna el usuario sino retorna *null*.

Componente Usuario
Se compone de un *EntityManagerFactory* para crear un *EntityManager*, esta clase contiene los siguientes métodos:

- **crearAutor:** que recibe como entradas un usuario y retorna como salida un usuario también.
  En el método se consulta a la base de datos a través del *EntityManager* para saber cuántos usuarios hay creados hasta el momento y así asignarle un id al usuario que se va a crear, luego configura el tipo (autor) y registra el usuario en la base datos.

- **crearEditor:** recibe y devuelve los mismo tipos que el método anterior pero se diferencia en que en la configuración del tipo es editor y no autor.

- **usuarioTipo:** el método recibe un string que hace referencia a un tipo diferente de usuario para hacer una consulta a la tabla de usuarios donde busca por tipo de usuario al especificado por el parámetro y devuelve una lista de usuarios que concuerdan con ese tipo.

Componente Submision
Tiene los mismos atributos que la clase anterior para acceder a la base de datos además de los siguientes métodos:

- **crearArticulo:** que recibe un articulo y un Bigdecimal que hace referencia al id de un usuario.
  El método consulta la tabla de articulos y asocia el articulo con el id del usuario para luego hacerle la persistencia en caso de que el idUsuario exista y devolver el articulo, en caso contrario devuelve *null*.

- **crearSubmision:** recibe un id de articulo y un id de evento, crea una submisión, configura su estado como *Pendiente*, asocia el articulo, el evento y el usuario, más la fecha en la fue creada y se persiste en la base de datos.

- **agregarRevisor:** recibe un id de submisión y un id de revisor, busca al revisor y a la submision y agrega en la revision la submisión asociada, configurando la calificación en -1 para expresar que no está calificada todavía.

- **crearEvento:** recibe un evento, busca los eventos creados para signarle un id y luego lo persiste en la tabla.

- **calificarSubmision:** recibe un id de revisión, una calificación y comentarios (String), busca la revisión con el id, configura los comentarios y la calificación y actualiza la tabla.
- **calcularCalificacion:** recibe un id de submisión, busca la submisión en la tabla de la base de datos y accede a la lista de revisiones de esa submisión sumando todas las calificación y sacando el promedio de estas.
- **articuloUsuario:** recibe un id de usuario, busca al usuario y retorna una lista de articulos que estén asociados a ese usuario.
- **submisionusuario:** recibe un id de usuario, busca al usuario en la lista y retorna otra lista con todas submisiones que coincidan con ese usuario.
- **revisionesSubmision:** recibe un id de submision para buscar la submisión en la tabla de submisión y luego buscar las revisiones que coincidan con ese id de submisión y retorna esa lista.
- **revisionesRevisor:** recibe un id de revisor y primero busca al usuario revisor que tenga ese id y luego retorna una lista de revisiones de ese usuario.
- **submisiones:** devuelve una lista con todas las submisiones de la tabla submision.
- **eventos:** devuelve una lista de todos los eventos registrados en la base de datos.

2.2 API

Dentro de este paquete están las siguientes clases :

API Auth:
Es creado como Web Service con un Web method llamado autenticarse que recibe un correo y una contraseña y autentica al usuario a partir del objeto tipo Componente autenticación.

API Usuario:
Web service que expone tres métodos Crear Autor, crear Revisor, Usuario Tipo (devuelve la lista de usuarios que pertenecen a cierto tipo), cuya implementación está dada por los métodos de la clase Usuario Componente.

API Submision:
Web Service que expone los servicios crear articulo, crear submision, agregar Revisor, calificar submision, calcular Calificación. Servicios provistos por el Submision Componente.

2.3 Entities

Dentro de este paquete se encuentran la siguientes clases:

Artículo

Queries:

- FindAll
- FindByTitulo
- FindByIdarticulo
- FindByResumen
- FindByPalabrasClave

Atributos:

- Titulo
- idArticulo
- Resumen
- palabrasClave
- usuarioIdusuario
- submisionLista

Evento

Queries:

- FindAll
- FindByIdEvento
- FindByTema
- FindByfechaInicio
- FindByfechaFin

Atributos:

- idEvento
- tema

- fechaInicio
- fechaFin
- usuarioLista
- submisionLista

Revisión

Queries:

- FindAll
- FindByIdrevision
- FindByCalificacion
- FindByComentarios

Atributos:

- idRevision
- Calificación
- Comentarios
- submicionIdsumision (Submision type)
- usuarioIdusuario (Usuario type)

Submisión

Queries:

- FindAll
- FindByIdsubmision
- FindByEstado
- FindByFechaSubida

Atributos:

- idsumision
- estado

- fechaSubida
- revisionList
- articuloIdArticulo
- eventoIdevento
- usuarioIdUsuario

Usuario

Queries:

- FindAll
- FindByIdUsuario
- FindByCorreo
- FindByContraseña
- FindByNombres
- FindByApellidos
- FindByFechaNacimiento
- FindByTipo
- Autenticación

Atributos:

- idUsuario
- correo
- contraseña
- nombres
- apellidos
- fechaNacimiento
- tipo
- revisionList
- articuloList
- submisionList

- eventoIdEvento

Main

En esta clase se publican los Endpoints:

- Auth con un address y un objeto implementador ApiAuth
- Usuario con un address y un objeto implementador ApiUsuarios
- Submision con un address y un objeto implementador ApiSubmision

JaveArticulos Cliente

Schema: WSDL-XSD

Auth-WSDL

- **Mensajes:** autenticarse, autenticarseResponse.
- **PortType:** ApiAuth.
  Operaciones:
  - **autenticarse**
- **Servicio:** ApiAuthService.

Auth-XSD
ComplexType
Además de los complexType a continuación definidos evento y usuario.

- **autenticarse** tiene una secuencia de dos elementos arg0 y arg1, ambos de tipo string con un mínimo de ocurrencias de 0.
- **autenticarseResponse** tiene una secuencia de un elemento llamado return de tipo usuario con un mínimo de ocurrencias de 0.

Submision-WSDL

- **Mensajes:**
  - **crearArticulo**
  - **crearSubmision**
  - **agregarRevisor**

  - **crearEvento**
  - **submisionUsuario**
  - **articulosusuario**
  - **eventos**
  - **submisiones**
  - **revisionesSubmision**
  - **calcularCalificación**
  - **calificarSubmision**
  - **crearArticuloResponse**
  - **crearSubmisionResponse**
  - **agregarRevisorResponse**
  - **crearEventoResponse**
  - **submisionUsuarioResponse**
  - **articulosusuarioResponse**
  - **eventosResponse**
  - **submisionesResponse**
  - **revisionesSubmisionResponse**
  - **calcularCalificaciónResponse**
  - **calificarSubmisionResponse**

- **PortType:** ApiSubmision.
  Operaciones:
  - **crearArticulo**
  - **crearSubmision**
  - **agregarRevisor**
  - **crearEvento**
  - **submisionUsuario**
  - **articulosusuario**
  - **eventos**
  - **submisiones**
  - **revisionesSubmision**
  - **calcularCalificación**

- **calificarSubmision**

- **Servicio:** ApiSubmisionService

Submision-XSD

ComplexType

- **calcularCalificaciónResponse** tiene una secuencia que se compone de un name arg0 de tipo entero con un mínimo de ocurrencias de 0.
- **agregarRevisor** tiene una secuencia de dos elementos arg0 y arg1, ambos de tipo decimal con un mínimo de ocurrencias de 0.
- **agregarrevisorResponse** tiene una secuencia de un elemento llamado return de tipo revsion con un mínimo de ocurrencias de 0.
- **Revisión** tiene una secuencia de varios elementos: calificación (entero), comentarios (string), idrevision(decimal),submisionIdSubmision(submision) e usuarioIdUsuario(usuario).
- **Submisión** tiene una secuencia de varios elementos: articuloIdarticulo2 (articulo), estado (string), idEvento (evento),fechasubida(dateTime), idSubmision(decimal) e usuarioIdusuario(usuario).
- **Articulo** tiene una secuencia de varios elementos: idArticulo(decimal), palabrasClave (string), resumen(string),titulo(string) e usuarioIdUsuario(usuario).
- **usuario** tiene una secuencia de varios elementos: apellidos(string), contrasena (string), correo(string),eventoIdevento(evento), fechanacimiento(dateTime), idUsuario(decimal), nombres(string) y tipo(string).
- **evento** tiene una secuencia de varios elementos: fechaInicio (dateTime), fechaFin (dateTime), idEvento(decimal),tema(string).
- **crearSubmision** tiene una secuencia de dos elementos arg0 y arg1 ambos de tipo decimal.
- **crearSubmisionResponse** tiene una secuencia con un elemento llamado return de tipo submision.

- **articuloUsuario** tiene una secuencia de un elemento arg0 de tipo decimal.
- **articuloUsuarioResponse** tiene una secuencia de un solo elemnto return de tipo articulo.
- **crearEvento** tiene una secuencia arg0 de tipo evento.
- **crearEventoResponse** tiene una secuencia de un elemento return de tipo evento.
- **revisionesSubmision** tiene una secuencia de un elemento arg0 de tipo decimal.
- **revisionesSubmisionResponse** tiene una secuencia de un solo elemento return de tipo revisión.
- **revisionesRevisor** tiene una secuencia de un elemento arg0 de tipo decimal.
- **revisionesRevisorResponse** tiene una secuencia de un elemento return de tipo revision
- **Eventos** tiene una secuencia.
- **EventosResponse** tiene una secuencia de un elemento return de tipo evento.
- **submisiones** tiene una secuencia.
- **submisionesResponse** tiene una secuencia de un elemento return de tipo submision.
- **crearArticulo** tiene una secuencia de dos elementos arg0 de tipo articulo y arg1 de tipo decimal.
- **crearArticuloResponse** tiene una secuencia de un elemento return de tipo articulo.
- **submisionUsuario** tiene una secuencia de un elemento arg0 de tipo decimal.
- **submisionUsuariorResponse** tiene una secuencia de un elemento return de tipo submision.
- **calificarSubmision** tiene una secuencia de tres elementos arg0 de tipo decimal, arg1 de tipo entero y arg2 de tipo string.

- **calificarSubmisionResponse** tiene una secuencia de un elemento return de tipo revision.

Usuario-WSDL

- **Mensajes:**
  - **crearAutor**
  - **crearRevisor**
  - **crearEditor**
  - **usuarioTipo**
  - **crearAutorResponse**
  - **crearRevisorResponse**
  - **crearEditorResponse**
  - **usuarioTipoResponse**
- **PortType:** ApiUsuarios.
  Operaciones:
  - **crearAutor**
  - **crearRevisor**
  - **crearEditor**
  - **usuarioTipo**
- **Servicio:** ApiUsuariosService

Usuario-XSD

ComplexType
Además de los complexType previamente definidos evento y usuario.

- **usuarioTipo** tiene una secuencia de un elemento arg0 de tipo string con un mínimo de ocurrencias de 0.
- **usuarioTipoResponse** tiene una secuencia de un elemento llamado return de tipo usuario con un mínimo de ocurrencias de 0.
- **crearEditor** tiene una secuencia de un elemento arg0 de tipo usuario con un mínimo de ocurrencias de 0.

- **crearEditorResponse** tiene una secuencia de un elemento llamado return de tipo usuario con un mínimo de ocurrencias de 0.
- **crearAutor** tiene una secuencia de un elemento arg0 de tipo usuario con un mínimo de ocurrencias de 0.
- **crearAutorResponse** tiene una secuencia de un elemento llamado return de tipo usuario con un mínimo de ocurrencias de 0.
- **crearRevisor** tiene una secuencia de un elemento arg0 de tipo usuario con un mínimo de ocurrencias de 0.
- **crearRevisorResponse** tiene una secuencia de un elemento llamado return de tipo usuario con un mínimo de ocurrencias de 0.