Universidad de La Laguna

Máster Universitario en Ciberseguridad e Inteligencia de Datos

Desarrollo de servicios web con fuentes de datos abiertas

Web services development with open data sources

Sergio Manuel Rodríguez Vega

La Laguna, 16-06-2021

D. **Carlos Pérez González**, con N.I.F. 45452719G profesor Titular de Universidad adscrito al Departamento de Matemáticas, Estadística e Investigación Operativa de la Universidad de La Laguna, como tutor

D. **Sergio Fernando Rodríguez Alonso**, con N.I.F. 42093441Z profesor Titular de Universidad adscrito al Departamento de Matemáticas, Estadística e Investigación Operativa de la Universidad de La Laguna, como cotutor

C E R T I F I C A (N)

Que la presente memoria titulada:

"Desarrollo de servicios web con fuentes de datos abiertas"

ha sido realizada bajo su dirección por D. **Sergio Manuel Rodríguez Vega**, con N.I.F. 79061112S.

Y para que así conste, en cumplimiento de la legislación vigente y a los efectos oportunos firman la presente en La Laguna a Fecha 16-06-2021

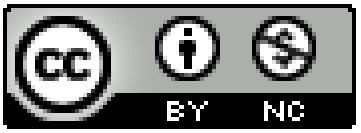
Agradecimientos

A familia y amigos por la colaboración en el testeo o recomendaciones de posibles mejoras, así como al tutor por la coordinación y orientación del proyecto.

Licencia

.

* Si quiere permitir que se compartan las adaptaciones de tu obra y NO quieres permitir usos comerciales de tu obra indica:



Resumen

El objetivo principal del presente proyecto es la obtención de servicio web, desde el cual se pueda obtener la información referente a los alumnos egresados de la Universidad de La Laguna. Este servicio cuenta con los mecanismos necesarios para ofrecer al usuario final la información de la manera más sencilla posible. Es con este fin que se implementan los diferentes parámetros de búsqueda de la API para un mayor ajuste de los resultados.

Para el desarrollo ha sido necesario realizar un estudio comparativo de los diferentes tipos de servicios web y API para poder determinar qué tipo se aplica mejor a las circunstancias actuales.

Los datos se obtienen directamente de la Universidad por medio de sus registros acerca de los alumnos. La selección de las interfaces utilizadas se ha realizado tras previo estudio de las ofertas presentes en el mercado actual, seleccionando las que mejor se ajustan a las necesidades del proyecto.

Palabras clave: API, REST, SOAP, Servicio Web

Abstract

The main objective of this project is to obtain a web service, from which information regarding the graduate students of the University of La Laguna can be obtained. This service has the necessary mechanisms for the end user to get information by different API search parameters are implemented for a better adjustment of the results.

For the development it has been necessary to carry out a comparative study of the different types of web services and APIs in order to determine which type is best applied to the current circumstances.

The data is obtained directly from the University through its records about the students. The selection of the interfaces used has been made after a previous study of the offers present in the current market, selecting those that best suit of the project

Keywords: API, REST, SOAP, Web service

Índice general

Índice de figuras

Índice de tablas

Capítulo 1: Introducción

1.1 Antecedentes

Los servicios web nacen de la necesidad de implementar un estándar de comunicación entre las plataformas de la época (PC, Mac ...) y los lenguajes de programación (PHP, Java...).

Con anterioridad se produjo un intento de implementar dichos estándares, con un resultado desfavorable, debido a no contar con el suficiente éxito para asentarse en la sociedad, algunos de estos son DCOM y CORBA, este último fracasó debido a las limitaciones para aplicaciones de alto nivel, en las cuales se requiere de seguridad o administración de las transacciones.

El segundo gran problema presente de esa época era el uso de RPC (Remote Procedure Call) para establecer la comunicación entre nodos. Esto, además de presentar problemas a nivel de seguridad, posee la dificultad de implementación en entornos web, lo cual es casi imposible debido a los bloqueos que presentan los firewalls, los cuales no permiten el tráfico de este tipo de mensajes.

Finalmente los Servicios Web o Web Services surgen para poder lograr la comunicación entre plataformas tan necesaria para la transición web. En la actualidad gran cantidad de sistemas denominados legacy están transitando hasta convertirse en Servicios Web.

1.1.1 ¿Qué es un Servicio Web?

Un servicio web (web service o web services) es una tecnología que utiliza un conjunto de protocolos y estándares que sirven para intercambiar datos entre aplicaciones. Distintas aplicaciones de software desarrolladas en lenguajes de programación diferentes, y ejecutadas sobre cualquier plataforma, pueden utilizar los servicios web para intercambiar datos en redes de ordenadores como Internet. La interoperabilidad se consigue mediante la adopción de estándares abiertos. Las organizaciones OASIS y W3C son los comités responsables de la arquitectura y reglamentación de los servicios Web.

Los servicios web que utilizamos ya no son sistemas autónomos cerrados. Trabajan juntos, los datos fluyen libremente entre ellos. Esto se logra mediante el uso de API de servicios web. [1]

1.1.2 Ventajas e inconvenientes

La principal ventaja que presentan los Servicios Web es que la comunicación no depende de una plataforma en específico, por lo cual el cliente y el servidor han de presentar una serie de similitudes o rasgos en común para poder comunicarse. Para ello, los servicios recurren a formatos estandarizados interpretados por los sistemas.

No obstante la principal desventaja se encuentra presente en el uso de dichos formatos estandarizados. En concreto, XML es un formato bastante voluminoso el cual genera grandes paquetes de datos, lo cual puede degenerar en conexiones lentas. Otra posibilidad que permite conectar a dos sistemas a través de Internet son las API web. Aunque, por lo general, son más rápidas, someten a cliente y servidor a especificaciones más concretas, con lo que la interoperabilidad se ve limitada.

Ventajas	Inconvenientes
Aportan interoperabilidad entre aplicaciones de software independientemente de sus propiedades o de las plataformas sobre las que se instalen. Los servicios Web fomentan los estándares y protocolos basados en texto, que hacen más fácil acceder a su contenido y entender su funcionamiento. Permiten que servicios y software de diferentes compañías ubicadas en diferentes lugares geográficos puedan ser combinados fácilmente para proveer servicios integrados.	Para realizar transacciones, no pueden compararse en su grado de desarrollo con los estándares abiertos de computación distribuida como CORBA (Common Object Request Broker Architecture). Su rendimiento es bajo si se compara con otros modelos de computación distribuida, tales como Java Remote Method Invocation (RMI), CORBA o Distributed Component Object Model (DCOM). Al apoyarse en HTTP, pueden esquivar medidas de seguridad basadas en firewall cuyas reglas tratan de bloquear o auditar la comunicación entre programas a ambos lados de la barrera.

Tabla 1.1: Ventajas e inconvenientes de los Servicios Web

1.2 API

Una API (Interfaz de Programación de Aplicaciones) es un conjunto de métodos para acceder a datos en sistemas que de otro modo estarían cerrados. Las API brindan a los programadores y desarrolladores las herramientas necesarias para crear software y servicios con datos y servicios de fuentes externas.[2]

Entre las principales funciones de las API se encuentra facilitar el trabajo a desarrolladores y programadores, ahorrando tiempo y dinero, esto mediante la simplificación de procesos. Por ejemplo a la hora de crear una tienda online, permite ahorrar el tiempo de desarrollar desde cero el sistema de pago o incluso la gestión del inventario de los productos. Se podrán utilizar APIs previamente desarrolladas para ese fin.

Para conectarse a las API y crear aplicaciones que utilicen los datos o las funciones que estas ofrecen, se puede utilizar una plataforma de integración distribuida que conecte todos los elementos, incluidos los sistemas heredados.

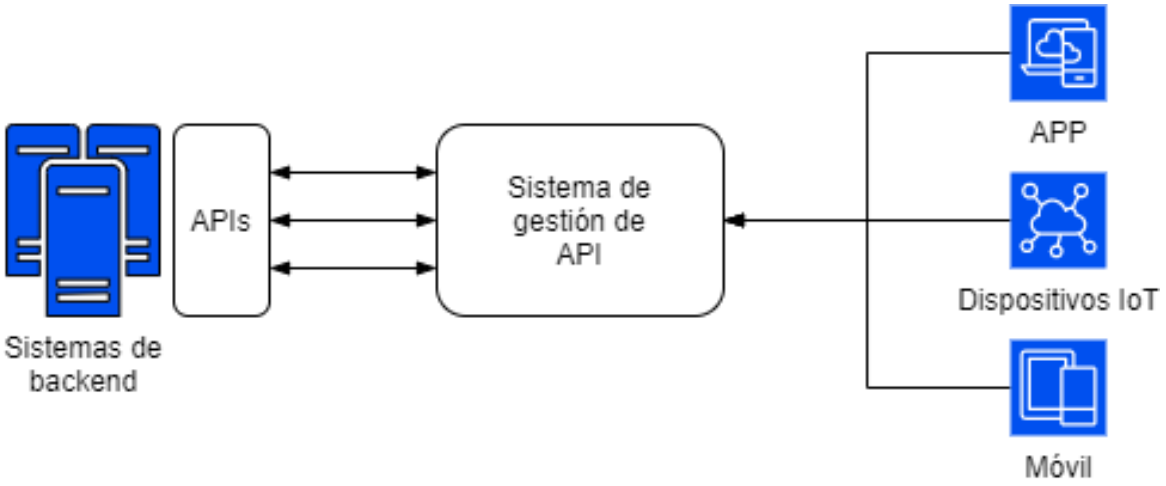


Figura 1.1: Funcionamiento de las API

1.2.1 API de Servicio Web

Una API de Servicio Web es aquella diseñada específicamente para un tipo de aplicaciones concretas. Estas API de Servicio Web son en la actualidad uno de los sellos distintivos del ecosistema web actual: un ecosistema basado en la apertura y el intercambio. Esto se encuentra presente en: Facebook, Tumblr … Los servicios web que utilizamos ya no son sistemas autónomos cerrados. Trabajan juntos, los datos fluyen libremente entre ellos. Logrado mediante el uso de API de servicios web. [2]

1.3 SOAP

SOAP es un protocolo estándar propuesto por el W3C [3] para interconectar servicios web, y eso se extiende a la llamada remota del procedimiento (XML-RPC). Por lo tanto, SOAP puede ser considerado como una evolución del mismo, mucho más complejo y desarrollado, el cual permite realizar llamadas de procedimientos remotos a rutinas distribuidas o servicios basados en una interfaz XML. Es debido a esto que los usuarios de SOAP pueden acceder a objetos y métodos alojados en servidores remotos, utilizando un mecanismo estándar que hace transparentes los detalles de implementación, como el lenguaje de programación.

SOAP fue diseñado como un protocolo de acceso a objetos en 1998 por Dave Winer, Don Box, Bob Atkinson y Mohsen Al-Ghosein por Microsoft, donde Atkinson y Al-Ghosein trabajaban en aquel entonces. La especificación SOAP actualmente es mantenida por el XML Protocol Working Group del World Wide Web Consortium.

SOAP envía y recibe mensajes mediante XML [4][5] encabezados HTTP-in ajustados. Las interfaces de métodos accesibles por servicios SOAP se encuentran definidas por WSDL (Lenguajes de Descripción de Servicios Web) [6]. Los WSDL de un servicio web consisten en una descripción de sus interfaces en XML, todo esto en un fichero que describe el nombre de los métodos, los parámetros o el tipo de respuesta que el servicio debe retornar. El uso de estos WSDL permite que el servicio sea especificado para diferentes lenguajes de programación, como Java.

Es debido a esto que, SOAP constituye un protocolo de alto nivel, facilitando la tarea de distribuir objetos entre diferentes servidores, y evitando las dificultades derivadas de definir el formato de mensaje, ni la llamada explícita a servidores remotos

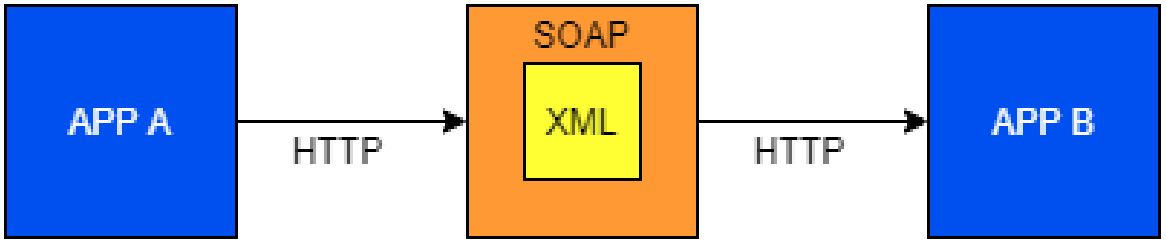


Figura 1.2: Funcionamiento de SOAP

Ventajas	Inconvenientes
Manejar entornos informáticos distribuidos Manejo de errores incorporado Extensibilidad Independiente del idioma, la plataforma y el transporte Estándar predominante para servicios web Soporte de otros estándares (WSDL, WS- *)	Más detallado Más difícil de desarrollar, requiere herramientas Conceptualmente más difícil, más "pesado" que REST

Tabla 1.2: Fortalezas y debilidades de SOAP

1.4 REST

Tras varios años, los arquitectos de Internet han encontrado un método alternativo para construir servicios web en forma de Transferencia de estado representacional (REST) [7].

REST es un estilo de arquitectura software para distribución de sistemas hipermedia. El término se introdujo y definió la Transferencia de Estado Representacional en 2000 por Roy Fielding en su tesis doctoral [8]. Fielding es uno de los principales autores del hipertexto de la especificación del Protocolo de transferencia (HTTP) [9].

Aunque REST se describió inicialmente en el contexto de HTTP, no se limita a ese protocolo. Las arquitecturas RESTful pueden basarse en otros protocolos de la capa de aplicación si ya proporcionan un vocabulario rico y uniforme. Las aplicaciones RESTful maximizan el uso de la interfaz bien definida y preexistente, además de otras capacidades integradas proporcionadas por el protocolo de red elegido, y minimizan la adición de nuevas características específicas de la aplicación encima.

En un entorno REST los clientes no se preocupan por el almacenamiento de datos, que permanece interno a cada servidor, por lo que se mejora la portabilidad del código del cliente. Los servidores no se preocupan por la interfaz de usuario o el estado del usuario, por lo que los servidores pueden ser más simples y escalables. Los servidores y clientes también se pueden reemplazar y desarrollar de forma independiente, siempre que no se modifique la interfaz. Por último, los servidores pueden ampliar o personalizar temporalmente la funcionalidad de un cliente transfiriendo lógica que puede ejecutar.

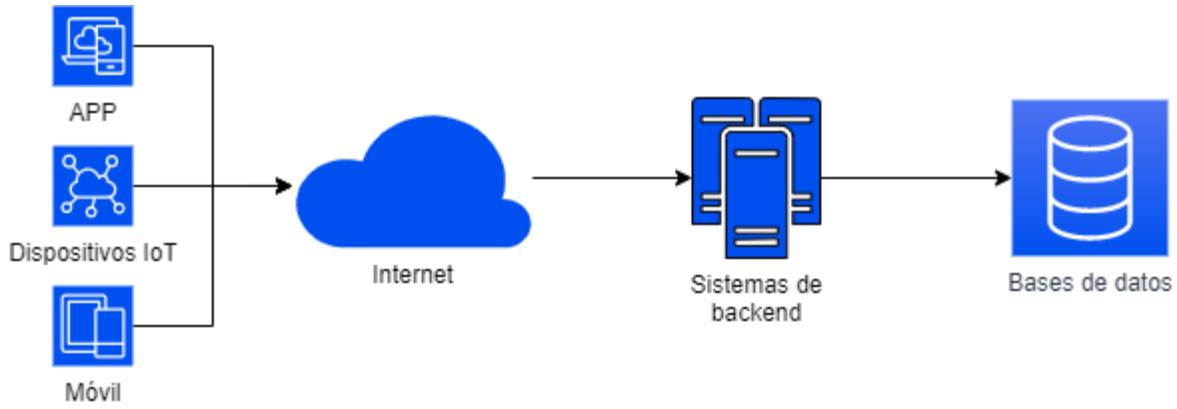


Figura 1.3: Funcionamiento de REST

Ventajas	Inconvenientes
Independiente del idioma y la plataforma	Asume un modelo de comunicación punto a punto
Mucho más sencillo de desarrollar que SOAP	No utilizable para entornos informáticos distribuidos
Pequeña curva de aprendizaje, menor dependencia de herramientas	Falta de soporte de estándares para la seguridad, etc.
Conciso, sin necesidad de una capa de mensajería adicional	Vinculado al modelo de transporte HTTP
Más cercano en diseño y filosofía a la Web	

Tabla 1.3: Fortalezas y debilidades de REST

1.5 SOAP vs REST

Tanto SOAP, como REST son adecuadas para el desarrollo de sistemas paralelos. Sin embargo, SOAP es más pesado que REST, debido a la verbosidad de las comunicaciones SOAP (XML aumenta el tiempo necesario para analizar los mensajes).

Por otro lado, la tecnología REST no podría usarse para implementar un GA distribuido siguiendo el modelo de isla, ya que no admite el uso o definición de procedimientos asincrónicos, mientras que SOAP sí lo admite.

Llegados a este punto se puede concluir que cada tecnología tiene sus ventajas y desventajas. No obstante, podemos definir algunas aplicaciones o servicios específicos en las cuales una de ellas posee una evidente ventaja frente a la otra a la hora de la implementación.

REST es más adecuado en los siguientes casos:

- El ancho de banda y los recursos son limitados
- Se necesitan operaciones CRUD (crear, leer, actualizar y eliminar) sin estado (no es necesario continuar con la operación)
- La información se puede almacenar en caché debido al funcionamiento totalmente sin estado del enfoque REST

SOAP es una buena solución en los siguientes casos:

- La aplicación necesita procesamiento e invocación asíncronos
- La aplicación necesita un nivel garantizado de fiabilidad y seguridad
- Ambas partes (proveedor y consumidor) deben acordar el formato de intercambio (especificaciones rígidas)
- La aplicación necesita información contextual y gestión del estado conversacional (operaciones con estado)

A partir de estas implementaciones REST, se diseñan varios caminos de mejora: cambiar los modelos para que se mueva más computación a los clientes, dejando el servidor como un solo centro para el intercambio de información entre los clientes; ese intercambio de información deberá reducirse al mínimo. Eso hará que este modelo se acerque más al modelo de isla, con solo las políticas de migración reguladas por el servidor. De esa manera, el cuello de botella del servidor casi se elimina.

1.6 Alcance

El alcance de este proyecto tiene como finalidad el desarrollo de un servicio web o una API, la cual sea capaz de recopilar y ofrecer de manera sencilla la información de los alumnos egresados de la Universidad de La Laguna de los que se tenga constancia hasta la fecha. Para la obtención de los datos se utilizan aquellos almacenados y generados por la propia universidad.

El desarrollo de este proyecto será llevado a cabo mediante el uso de las herramientas JDeveloper, haciendo el uso de lenguajes de programación XQuery, XML y WDSL[20]. Por otro lado, para el alojamiento de la base de datos se utiliza Oracle Database, con apoyo de la herramienta SQLDeveloper para su gestión.

1.7 Objetivos

El objetivo principal del proyecto es obtener un servicio fácil de usar, de gran velocidad, versátil y sencillo. Con la finalidad de cumplir este objetivo se presentan los siguientes puntos:

- Conocer las diferentes posibilidades de servicio web
- Estudiar y conocer los diferentes tipos de API
- Conocer las técnicas de desarrollo de API
- Conocer los mecanismos de conexión con bases de datos
- Seguridad de los servicios
- Desarrollo del servicio

Capítulo 2: Herramientas

Uno de los aspectos principales y por consiguiente más importantes del proyecto es la selección de herramientas que serán utilizadas durante el desarrollo, pues estas pueden contribuir de manera beneficiosa en el correcto desarrollo.

2.1 Eclipse

La Fundación Eclipse proporciona un entorno maduro, escalable y amigable para las empresas en pro de la colaboración e innovación de software de código abierto. La Fundación alberga el Eclipse IDE, Jakarta EE y más de 350 proyectos de código abierto, incluidos tiempos de ejecución, herramientas y marcos para una amplia gama de dominios tecnológicos como Internet de las cosas, automoción, geoespacial, ingeniería de sistemas y muchos otros [10].

Eclipse fue desarrollado originalmente por IBM como el sucesor de su familia de herramientas para VisualAge. Es una plataforma de software compuesto por un conjunto de herramientas de programación de código abierto multiplataforma para desarrollar lo que el proyecto llama "Aplicaciones de Cliente Enriquecido", opuesto a las aplicaciones "Cliente-liviano" basadas en navegadores. Esta plataforma típicamente ha sido usada para desarrollar entornos de desarrollo integrados (IDE), como el IDE de Java llamado Java Development Toolkit (JDT) y el compilador (ECJ) que se entrega como parte de Eclipse [11].

Figura 2.1: Logotipo de Eclipse

2.2 SQLDeveloper

Oracle SQL Developer es un entorno de desarrollo integrado gratuito destinado a la gestión SQL de bases de datos Oracle, simplificando además el desarrollo en casos de implementaciones tradicionales y cloud. SQL developer ofrece adicionalmente un desarrollo completo en todo su espectro de aplicaciones PL/SQL, así como una hoja de trabajo para la ejecución de consultas y scripts, por otro lado una solución completa de modelado de los datos y finalmente una plataforma de migración para la migración de bases de datos de terceros hacia oracle [12].

Adicionalmente proporciona soluciones para diversidad de perfiles de entre los cuales debemos destacar:

- Desarrollador
- DBA
- Arquitecto de aplicaciones y modelador de datos
- Administrador y el desarrollador de aplicaciones web
- Migraciones de bases de datos de terceros

2.3 Oracle Service Manager

Oracle Web Services Manager (OWSM) proporciona un marco de políticas para administrar y proteger los servicios web de manera uniforme en toda su organización. Proporciona capacidades para crear, hacer cumplir, ejecutar y monitorear políticas de servicios web, como seguridad, mensajería confiable, MTOM y políticas de direccionamiento. OWSM puede ser utilizado tanto por desarrolladores, en tiempo de diseño, como por administradores de sistemas en entornos de producción [14].

Además proporciona agilidad empresarial para responder a las amenazas de seguridad y las brechas de seguridad al permitir que los cambios de políticas se apliquen en tiempo real sin la necesidad de interrumpir los procesos comerciales en ejecución.

Profiere la "seguridad de la primera milla" a través de agentes de cliente para proteger a los clientes de servicios web, y la "seguridad de la última milla" a través de agentes de servidor que aseguran los servicios web. Si sus servicios web son accesibles sólo desde dentro de la intranet corporativa, generalmente aún requieren autenticación y autorización. Además, a menudo se requiere auditoría para abordar el cumplimiento normativo.

2.4 Oracle Service Bus

Oracle Service Bus (OSB) ha sido diseñado para la conexión, mediación y administración de las interacciones entre servicios, aplicaciones y múltiples estancias de ESB por medio de una red de servicios en cuanto a empresa.

Por otro lado permite la integración de servicios controlados por la configuración, con ruteo basado en identidades y contenido inteligente, así como una mejora la productividad del desarrollador debido a la integración de servicios de código libre.

Oracle Service Bus también brinda transporte nativo para aplicaciones empaquetadas y planificación de recursos empresariales, junto con la conectividad a aplicaciones basadas en el servidor e IBM WebSphere.

Oracle Service Bus brinda capacidades incorporadas para la virtualización de servicios, Web Service Security (WS-Security) y el cumplimiento de políticas en torno a la regulación y la agrupación de servicios a fin de cumplir con los requerimientos de Confiabilidad, Disponibilidad y Desempeño (RASP) y evitar la sobrecarga de servicios de back-end.

2.5 XQuery

XQuery es un lenguaje diseñado para la redacción de consultas sobre colecciones definidas en XML. Este lenguaje abarca desde archivos XML hasta bases de datos relacionales con capacidad de conversión de sus registros a XML. La principal función es la extracción de información de un conjunto de datos organizados en estructura tipo árbol de etiquetas XML. Por lo tanto podemos decir que XQuery es independiente a los datos de origen.

XQuery es un lenguaje funcional, lo que implica que, en lugar de ejecutar una lista de comandos al estilo de un lenguaje procedimental clásico, cada consulta es evaluada a modo de expresión y retorna un resultado, al igual que ocurre en SQL. Dicho esto, diversas expresiones pueden ser combinadas de manera flexible para crear unas nuevas con un grado superior de complejidad y mayor potencia semántica.

Podemos concluir que dicho lenguaje está destinado a ser el futuro estándar de consultas sobre documentos XML. Actualmente, es un conjunto de borradores en el cual trabaja el grupo W3C[13]. No obstante, a pesar de no tener una redacción definitiva ya existen o están en proceso numerosas implementaciones de motores y herramientas que lo soportan.

2.6 Jdeveloper

JDeveloper es un entorno de desarrollo integrado (IDE) desarrollado por Oracle y disponible de forma gratuita desde 2005. Se trata de un entorno que cubre todo el ciclo de vida del desarrollo software, proporcionando funcionalidad para las fases de diseño, codificación, depuración y despliegue, mediante una aproximación visual y declarativa.

Las principales características de JDeveloper son las siguientes:

- Ofrece funcionalidades para el desarrollo en Java, JavaScript, SQL, PL/SQL, HTML, XML, PHP y BPEL.
- Proporciona editores WYSIWYG para HTML, JSP, JSF y Swing.
- Genera de forma automática POJOs o EJB en relación con tablas de una base de datos.
- Incluye soporte para la interacción con Maven y Ant.

2.7 WebLogic

Oracle WebLogic es un servidor de aplicaciones Java EE, además de un servidor web HTTP, desarrollado por BEA Systems, posteriormente adquirida por Oracle Corporation. Este servidor se encuentra desarrollado tanto para sistemas Unix, Linux, Microsoft Windows, como para otras plataformas.

WebLogic puede utilizar Oracle, DB2, Microsoft SQL Server, y otras bases de datos que se ajusten al estándar JDBC. El servidor WebLogic es compatible con WS-Security y cumple con los estándares de J2EE 1.3 desde su versión 7 y con la J2EE 1.4 desde su versión 9 y Java EE para las versiones 9.2 y 10.x.

Los componentes principales que ofrece WebLogic son los siguientes:

- Portal, que incluye el servidor de comercio y el servidor de personalización
- Weblogic Integration
- Weblogic Workshop, una IDE para Java
- JRockit, una máquina virtual Java (JVM) para las CPU de Intel.

Además de lo antes mencionado WebLogic Server incluye interoperabilidad .NET.

Figura 2.2: Logo de Oracle WebLogic

Capítulo 3: Desarrollo del servicio

En este apartado se expone el desarrollo completo del proyecto, se tratarán temas, como el desarrollo de funcionalidades del servicio, la metodología aplicada o el desarrollo de la interfaz y la razón de estas elecciones.

3.1 Metodología

Cuando hablamos de metodologías hacemos referencia a un conjunto de procedimientos utilizados con la finalidad de cumplir un objetivo u objetivos que se ha determinado para la resolución de un proyecto o investigación. Por tanto metodología puede definirse como el estudio o elección de un método adecuado aplicable a determinado objeto.

Por otro lado cabe resaltar que no es correcto definir como metodología a cualquier procedimiento, pues se trata de un concepto de gran amplitud, siendo preferible usar la denominación de método en dichos casos.

3.1.1 Metodologías ágiles

A la hora de hablar de metodologías ágiles hacemos referencia a aquellas que nos permiten adaptar el flujo de trabajo a las condiciones específicas del proyecto en cada circunstancia, aportando así flexibilidad e inmediatez de respuesta ante las posibles adversidades durante el desarrollo.

Esencialmente, en la actualidad la tendencia de las empresas es optar o apostar por dichas metodologías a la hora de gestionar los proyectos de forma flexible, autónoma y eficaz, reduciendo así los costes e incrementando la productividad.

A día de hoy se pueden destacar cinco de estas metodologías como las más aplicadas en las empresas, teniendo en cuenta que cada una de ellas presenta una serie de ventajas dependiendo del tipo de proyecto que se desea afrontar.

- Extreme Programming XP
- SCRUM
- Kanban
- Agile Inception
- Design Sprint (Google)

La aplicación de este tipo de metodologías aporta una serie de ventajas entre las cuales podemos destacar: mejora de la calidad del producto, mayor motivación de los trabajadores, trabajo colaborativo, uso de métricas más relevantes o mayor control y capacidad de predicción entre otras.

3.1.2 Scrum

Para el desarrollo del presente proyecto y tras un análisis de las posibilidades se ha optado por la aplicación de una metodología ágil, en este caso se ha utilizado "Scrum", debido a las ventajas presentes frente al resto en cuanto a las necesidades presentes.

Scrum es un método para trabajar en equipo a partir de iteraciones o Sprints. Así pues, Scrum es una metodología ágil, por lo que su objetivo será controlar y planificar proyectos con un gran volumen de cambios de última hora, en donde la incertidumbre sea elevada.

Se suele planificar por semanas. Al final de cada Sprint o iteración, se va revisando el trabajo validado de la anterior semana. En función de esto, se priorizan y planifican las actividades en las que invertimos nuestros recursos en el siguiente Sprint.

La metodología Scrum se centra en ajustar sus resultados y responder a las exigencias reales y exactas del cliente. De ahí, que se vaya revisando cada entregable, ya que los requerimientos van variando a corto plazo. El tiempo mínimo para un Sprint es de una semana y el máximo es de cuatro semanas.

3.2 Obtención de datos

El primer paso necesario es identificar antes de nada la fuente de datos que nos proporcionará la información requerida. Para poder identificar correctamente dónde se encuentran los datos a consultar, es necesario la cooperación de personal que conozca bien cómo está organizada la información.

Una vez identificada la fuente será necesario desarrollar una consulta óptima con tiempos de respuesta razonables para el volumen de datos a extraer.

Para optimizar los tiempos de respuesta, en el caso de que estemos consultando una base de datos relacional, podremos apoyarnos en la creación de índices y vistas materializadas que reduzcan el tiempo total de consulta.

La información utilizada por el servicio es obtenida de los documentos generados por la propia universidad de La Laguna donde se recopila la información de los alumnos a lo largo del curso académico.

En nuestro caso la información requerida es aquella que hace referencia a los alumnos egresados, siendo la estructura la siguiente.

Variable	Campo	Tipo
CURSOACAD_REF	Curso académico de referencia.	Varchar2(7)
TIPO_DOC_IDENTIDAD	Tipo de documento de identidad que presenta el individuo.	Varchar2(1)
NUMERO_DOCUMENTO	Número del documento de identidad que presenta el individuo.	Varchar2(15)
LETRA_DOCUMENTO	Letra que acompaña al número del documento de identidad que presenta el individuo.	Varchar2(1)
APELLIDO_1	Primer apellido de la persona	Varchar2(255)
APELLIDO_2	Segundo apellido de la persona	Varchar2(255)
NOMBRE	Nombre de la persona	Varchar2(255)
SEXO	Sexo del individuo.	Varchar2(1)
NACIONALIDAD	País de la nacionalidad del individuo.	Varchar2 (3)
PAIS_NACIMIENTO	País de nacimiento.	Varchar2 (3)
FECHA_NACIMIENTO	Fecha de nacimiento.	Varchar2 (10)
UNIVERSIDAD	Código de la universidad.	Varchar2(3)
COD_TITULACION	Código de estudios en los que se ha titulado la persona.	Varchar2(4)
FECHACCESOSUE	Año en el que el titulado ingresó por primera vez en el sistema.	Varchar2(4)

FECHACCESOEST	Año en el que la persona accedió a los estudios en los que se ha titulado.	Varchar2(4)
MODACCESOEST	Forma de admisión de la persona en los estudios en los que la persona se ha titulado (***).	Varchar2(2)
NOTAINGRESOEST	Nota de ingreso de la persona en los estudios en los que se ha titulado	Number(5,2)
FECHAEGRESO	Convocatoria y año de egreso de la persona en los estudios en los que se ha titulado.	Varchar2(7)
NOTAEGRESOEST4	Nota de egreso de la persona en los estudios en los que se ha titulado	Number(5,2)
NOTAEGRESOEST10	Nota de egreso de la persona en los estudios en los que se ha titulado	Number(5,2)
ESTUPADRE	Nivel de estudios del padre	Varchar2(1)
ESTUMADRE	Nivel de estudios de la madre	Varchar2(2)
OCUPADRE	Ocupación del padre	Varchar2(2)
OCUMADRE	Ocupación de la madre	Varchar2(2)
TRABAJOALUM	Realización de trabajo	Varchar2(1)
MUNIFAM	Municipio del domicilio de residencia del núcleo familiar	Varchar2(5)
CODPOSTFAM	Código postal del domicilio de residencia del núcleo familiar	Varchar2(5)

PAISFAM	País del domicilio de residencia del núcleo familiar	Varchar2 (3)

Tabla 3.1: Diseño tabla egresados

3.2.1 Creación de la base de datos

Para la creación de la base de datos se ha optado por utilizar la herramienta Oracle SQL developer, esto debido a que facilita la creación de bases de datos nativas de oracle, en cuanto a la configuración del fichero jndi se ha utilizado WebLogic.

El proyecto ya cuenta con una estructura definida para la implementación de la base de datos, dando como resultado la siguiente tabla:

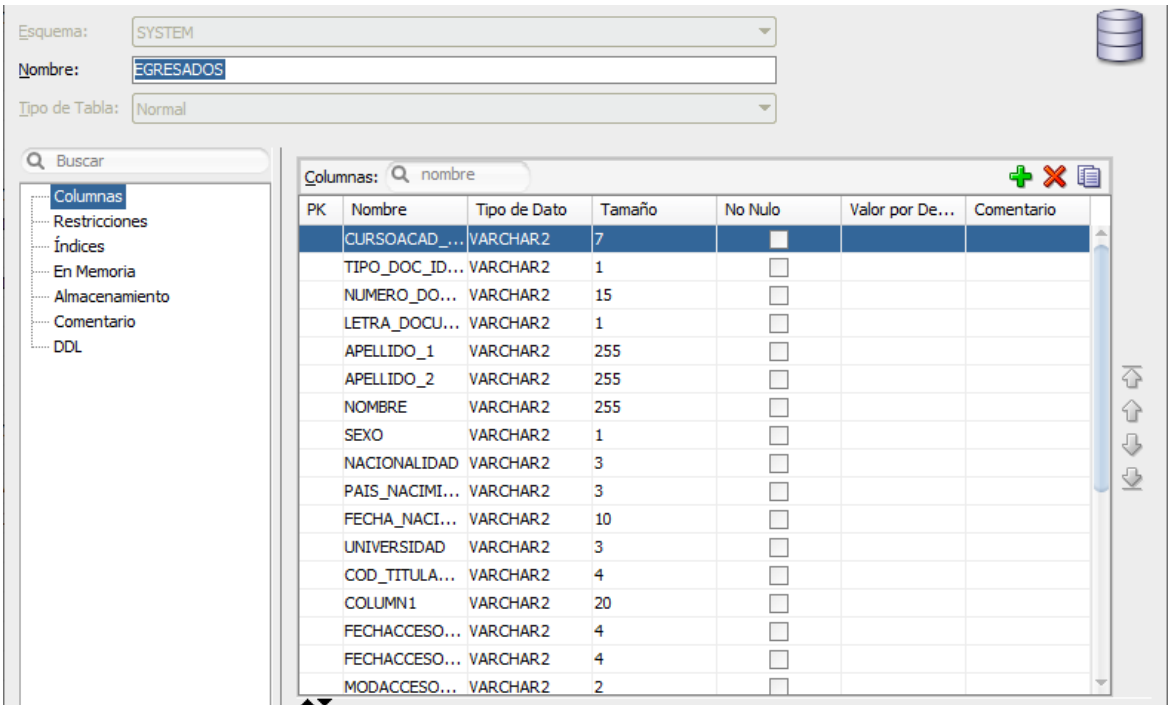


Figura 3.1: Estructura de la tabla egresados

3.3 Estructura

El proyecto generado inicialmente cuenta con una estructura muy sencilla generada automáticamente por JDeveloper, la cual consta de un fichero “pom.xml”, el fichero de estructura de los componentes “TFM.jws” y una carpeta

con el contenido del proyecto, en nuestro caso “ull-alumno”.

3.3.1 Estructura de directorios

Las carpetas ayudan a estructurar los proyectos y así organizar los diferentes artefactos que crearemos más adelante. La estructura de carpetas también será visible después de la implementación de un proyecto en la consola OSB. Entonces, en tiempo de ejecución, si alguien necesita navegar a un determinado artefacto a través de la consola, una estructura de carpetas inteligente puede hacer la vida mucho más fácil.

El significado de la estructura de carpetas que usaremos en este proyecto se enumera en la siguiente tabla [15]:

Carpeta	Contenido
Business	Artefactos y servicios comerciales
Proxy	Servicio proxy
Resources	Recursos
Tests	Tests
XQueries	Scripts XQuery

Tabla 3.2: Estructura de directorios

3.4 Desarrollo

En este apartado se expone el desarrollo completo, se tratarán temas, como el desarrollo de funcionalidades de la aplicación, la creación de los adaptadores a bases de datos, finalizando con la creación de los endpoints y el porqué de estas elecciones

3.4.1 Creación del JNDI

Cuando hablamos de JNDI hacemos referencia a Java Naming and Directory Interface, la cual es una interfaz de programación la cual proporciona las

funcionalidades de nombrado y directorio a las aplicaciones escritas usando Java. Está definido para ser independiente de cualquier implementación de servicio de directorio. Así se puede acceder a una gran variedad de directorios de una forma común

La API JNDI permite escribir código Java capaz de realizar operaciones sobre directorios. Se trata de una API uniforme a todos los tipos de servicios de directorios, siendo similar a JDBC.

Para establecer nuestro propio JNDI enlazado a nuestra base de datos necesitamos acceder a la consola de weblogic y establecer un nuevo servicio de orígenes de datos.

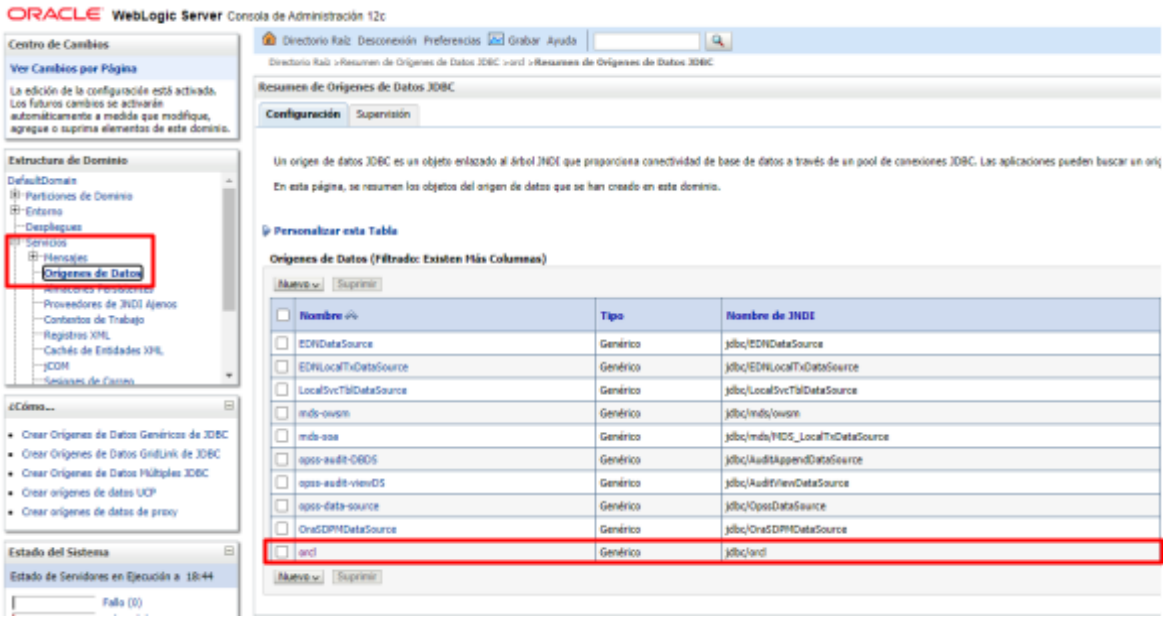


Figura 3.2: Configuración de weblogic

Una vez realizada esta configuración es necesario ir a la sección de despliegues y establecer el pool de conexiones salientes de nuestra aplicación donde enlazamos el fichero jdbc generado.

3.4.2 Definición de tipos de datos

Una vez que tenemos la estructura de directorios el siguiente paso es establecer

el Schema, en el cual se definirán los tipos de datos usados tanto para el Request y el Response del servicio el cual queda de la siguiente manera.

Dentro del elemento “Egresados” se encuentran definidos los tipos de datos de todos los elementos de la base de datos que serán retornados como respuesta del servicio.

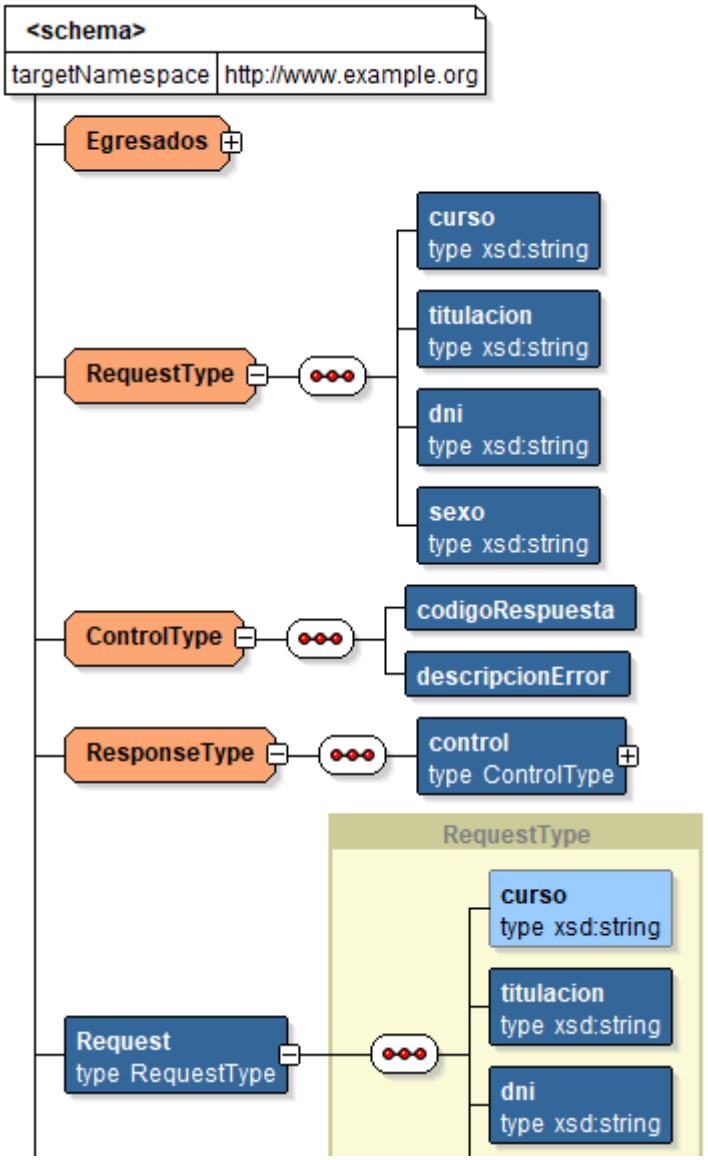


Figura 3.3: Tipos de datos visual

En cuanto al código se han definido los grupos de elementos en un “complextype”, donde se agrupan los elementos y se define el tipo de los mismos, como se puede observar en la siguiente imagen.

```
<?xml version="1.0" encoding="windows-1252" ?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns="http://www.example.org"
            targetNamespace="http://www.example.org" elementFormDefault="qualified">
  <xsd:complexType name="Egresados">
    <xsd:sequence>
      <xsd:element name="cursoacadRef" type="xsd:string" minOccurs="0" nillable="true"/>
      <xsd:element name="tipoDocIdentidad" type="xsd:string" minOccurs="0" nillable="true"/>
      <xsd:element name="numeroDocumento" type="xsd:string" nillable="true"/>
      <xsd:element name="letraDocumento" type="xsd:string" minOccurs="0" nillable="true"/>
      <xsd:element name="apellido1" type="xsd:string" minOccurs="0" nillable="true"/>
      <xsd:element name="apellido2" type="xsd:string" minOccurs="0" nillable="true"/>
      <xsd:element name="nombre" type="xsd:string" minOccurs="0" nillable="true"/>
      <xsd:element name="sexo" type="xsd:string" minOccurs="0" nillable="true"/>
      <xsd:element name="nacionalidad" type="xsd:string" minOccurs="0" nillable="true"/>
      <xsd:element name="paisNacimiento" type="xsd:string" minOccurs="0" nillable="true"/>
      <xsd:element name="fechaNacimiento" type="xsd:string" minOccurs="0" nillable="true"/>
      <xsd:element name="universidad" type="xsd:string" minOccurs="0" nillable="true"/>
      <xsd:element name="codTitulacion" type="xsd:string" minOccurs="0" nillable="true"/>
      <xsd:element name="fechaccesosue" type="xsd:string" minOccurs="0" nillable="true"/>
      <xsd:element name="fechaccesoest" type="xsd:string" minOccurs="0" nillable="true"/>
      <xsd:element name="modaccesoest" type="xsd:string" minOccurs="0" nillable="true"/>
      <xsd:element name="notaingresoest" type="xsd:decimal" minOccurs="0" nillable="true"/>
      <xsd:element name="fechaegreso" type="xsd:string" minOccurs="0" nillable="true"/>
      <xsd:element name="notaegresoest4" type="xsd:decimal" minOccurs="0" nillable="true"/>
      <xsd:element name="notaegresoest10" type="xsd:decimal" minOccurs="0" nillable="true"/>
      <xsd:element name="estupadre" type="xsd:string" minOccurs="0" nillable="true"/>
      <xsd:element name="estumadre" type="xsd:string" minOccurs="0" nillable="true"/>
      <xsd:element name="ocupadre" type="xsd:string" minOccurs="0" nillable="true"/>
      <xsd:element name="ocumadre" type="xsd:string" minOccurs="0" nillable="true"/>
      <xsd:element name="trabajoalum" type="xsd:string" minOccurs="0" nillable="true"/>
      <xsd:element name="munifam" type="xsd:string" minOccurs="0" nillable="true"/>
      <xsd:element name="codpostfam" type="xsd:string" minOccurs="0" nillable="true"/>
```

Figura 3.4 : Tipos de datos XSD

3.4.3 Adaptadores JCA

El objetivo principal de un adaptador de base de datos es exponer las tablas SOA

y SQL con la mayor transparencia y de la manera menos intrusiva posible. Desde el punto de vista de integración, una solución genérica centrada en lo estándar tiene el mayor alcance.

Al exponer bases de datos a SOA, también se trata de combinar las tecnologías de SQL y XML, el primero un lenguaje ideal para consultar información, el segundo un formato ideal para transportar y representar información. Si bien la compatibilidad con procedimientos almacenados es menos estándar en todas las bases de datos, Oracle Database Adapter brinda compatibilidad con procedimientos almacenados [16].

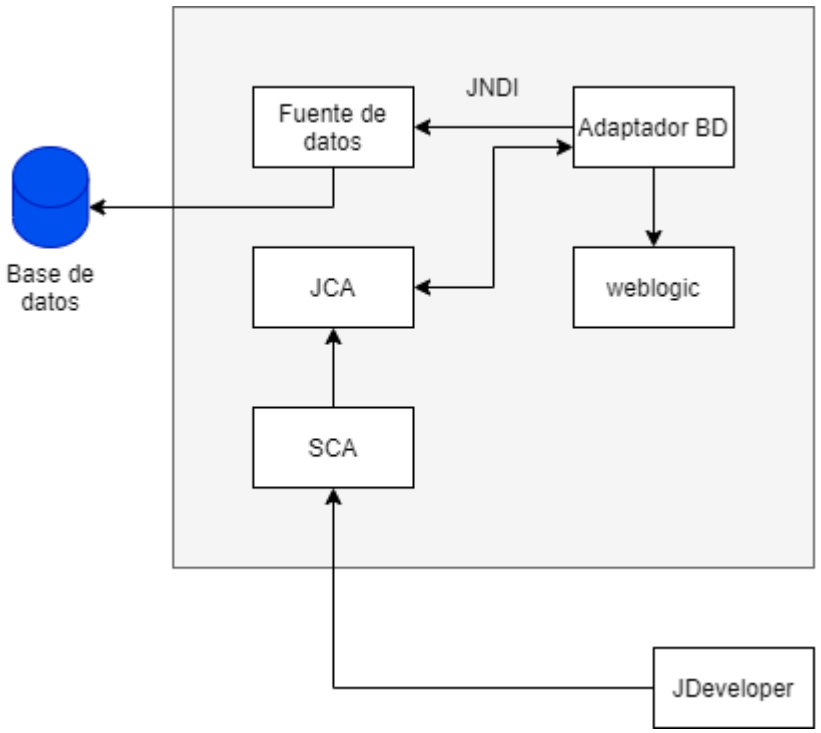


Figura 3.5: Funcionamiento del adaptador de BD

El Adaptador de base de datos de Oracle consta de varias instancias; cada instancia representa una conexión a un punto final de la base de datos. Diferentes procesos SOA pueden apuntar a la misma instancia de adaptador, mientras que diferentes puntos finales de servicio en un proceso SOA pueden apuntar a diferentes instancias de adaptador.

En nuestro caso se ha implementado un adaptador con la base de datos para cada petición, generando como resultado un fichero de extensión .jca cuya estructura es la siguiente.

```
<adapter-config name="DBReference" adapter="db" wsdlLocation="DBReference.wsdl" xmlns
  <connection-factory UIConnectionName="TFMdb" location="eis/db/tfm"/>
  <endpoint-interaction portType="DBReference_ptt" operation="DBReferenceSelect">
    <interaction-spec className="oracle.tip.adapter.db.DBReadInteractionSpec">
      <property name="DescriptorName" value="DBReference.Egresados"/>
      <property name="QueryName" value="DBReferenceSelect"/>
      <property name="MappingsMetaDataURL" value="DBReference-or-mappings.xml"/>
      <property name="ReturnSingleResultSet" value="false"/>
      <property name="GetActiveUnitOfWork" value="false"/>
    </interaction-spec>
  </endpoint-interaction>

</adapter-config>
```

Figura 3.6: Código fuente de un JCA

Business Service Testing - DBReference

Back | Close

Request Document

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/">
    <soap:Header xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
    </soap:Header>
    <soapenv:Body>
        <dbr:DBReferenceSelect_titulacionInputParameters xmlns:dbr="http://xmlns.oracle.com/pcbpel/adapter/db/top/DBReference">
            <dbr:titulacion>M577</dbr:titulacion>
        </dbr:DBReferenceSelect_titulacionInputParameters>
    </soapenv:Body>
</soapenv:Envelope>
```

Response Document

```
<soap-env:Envelope xmlns:soap-env="http://schemas.xmlsoap.org/soap/envelope/">
    <soap-env:Body>
        <EgresadosCollection xmlns="http://xmlns.oracle.com/pcbpel/adapter/db/top/DBReference" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
            <Egresados>
                <cursoacadRef>2018/19</cursoacadRef>
                <tipoDocIdentidad>1</tipoDocIdentidad>
                <numeroDocumento>31485***</numeroDocumento>
                <letraDocumento>Y</letraDocumento>
                <apellido1>LUPIAÑEZ</apellido1>
                <apellido2>ALPUENTE</apellido2>
                <nombre>LUCIA</nombre>
                <sexo>M</sexo>
                <nacionalidad>724</nacionalidad>
                <paisNacimiento>724</paisNacimiento>
                <fechaNacimiento>25/01/1994</fechaNacimiento>
```

Figura 3.7: Test del adaptado de la base de datos

3.4.4 Creación de endpoints

Cuando hablamos de un endpoint o punto final de comunicación hacemos referencia a un tipo de nodo de red. Se trata de una interfaz expuesta por un comunicador o un canal de comunicación, como lo puede ser por ejemplo un tema de un foro o incluso un grupo dentro de un sistema de comunicación como puede ser WhatsApp.

El primer paso que se debe realizar es definir un fichero WSDL(Web Services Description Language), el cual es un formato del Extensible Markup Language que se utiliza para describir servicios web[17]. Para permitir que las operaciones definidas en dicho documento sean accesibles es necesario crear un servicio virtual denominado “Proxy Service”.

Una vez establecido lo antes mencionado el cliente será capaz de consumir el servicio a través de una URI(Uniform Resource Identifier) expuesta por el servicio. Dentro del servicio Proxy se realizan transformaciones y/o ruteos, luego los datos transformados son enviados al servicio final (Business Service), este procesa la información y nuevamente regresa el resultado al servicio virtual, para después ser entregado al cliente.

Tener en cuenta que antes de ser devuelto al cliente se realiza la transformación de la respuesta para que concuerde con la respuesta que hemos definido y que espera el cliente.

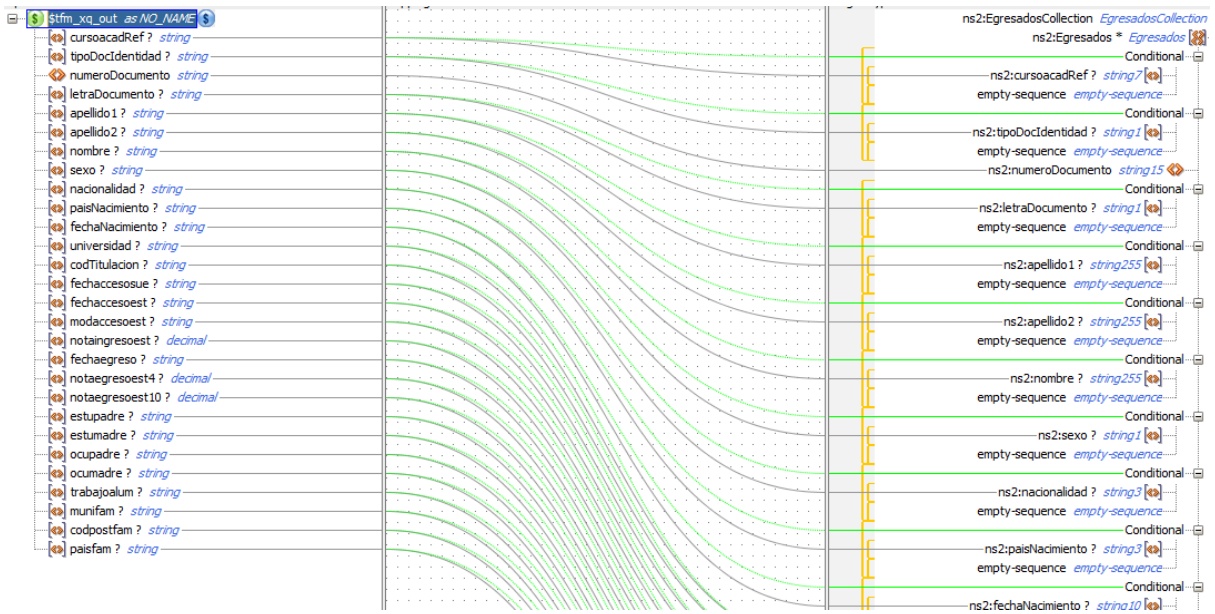


Figura 3.8: Esquema Xquery

```
xquery version "1.0" encoding "utf-8";
(:: OracleAnnotationVersion "1.0" ::)
declare namespace ns1="http://www.example.org";
(:: import schema at "../Resources/datatypes.xsd" ::)
declare namespace ns2="http://xmlns.oracle.com/pcbpel/adapter/db/top/getAlumnos_db";
(:: import schema at "../Resources/getAlumnos_db_table.xsd" ::)
declare variable $tfm_xq_out as element() (:: element(*, ns1:Egresados) ::) external;
declare function local:func($tfm_xq_out as element() (:: element(*, ns1:Egresados) ::)) as element() (:: schema-element(ns2:Egresad
    <ns2:EgresadosCollection>
        <ns2:Egresados>
            {
                if ($tfm_xq_out/ns1:cursoacadRef)
                then <ns2:cursoacadRef>{fn:data($tfm_xq_out/ns1:cursoacadRef)}</ns2:cursoacadRef>
                else ()
            }
            {
                if ($tfm_xq_out/ns1:tipoDocIdentidad)
                then <ns2:tipoDocIdentidad>{fn:data($tfm_xq_out/ns1:tipoDocIdentidad)}</ns2:tipoDocIdentidad>
                else ()
            }
            <ns2:numeroDocumento>{fn:data($tfm_xq_out/ns1:numeroDocumento)}</ns2:numeroDocumento>
            {
                if ($tfm_xq_out/ns1:letraDocumento)
                then <ns2:letraDocumento>{fn:data($tfm_xq_out/ns1:letraDocumento)}</ns2:letraDocumento>
                else ()
            }
            {
                if ($tfm_xq_out/ns1:apellido1)
```

Figura 3.9: Código XQuery

3.5 Seguridad del servicio

Es de gran importancia establecer seguridad al servicio para evitar la posibilidad de que sea atacado, para esto se utiliza OWSM (Oracle Web Services Manager), que es un marco de tiempo de ejecución para la creación de políticas de seguridad, administración y gobernanza. Se crean políticas, las adjunta a servicios en Service Bus y las aplica en varios puntos del ciclo de vida de la mensajería con agentes OWSM [18].

Las políticas de OWSM habilitan:

- WS-AT
- WS-RM
- WS-Security / WS-SC.
- No se pueden utilizar políticas integradas en WSDL.

Para securizar los servicios web se utiliza una política a nivel de servicio. Esta política utiliza las credenciales de la cabecera de SOAP WS-Security UsernameToken para autenticar a los usuarios. Solo se soporta el mecanismo de texto sin formato. Las credenciales se autentican en el almacén de identidades configurado. Esta política se puede asociar a cualquier punto final basado en SOAP. El proveedor de autorización utilizado es XACML Authorization Provider.

Para cada aplicación que necesite consumir uno o varios servicios web se le creará un usuario y se dará a este usuario permiso solamente sobre aquellas operaciones que necesite.

Service Bus y OWSM utilizan determinados servicios para la autenticación y autorización. OWSM usa Java Platform Security (JPS), por lo que Service Bus usa proveedores JPS para políticas OWSM. Service Bus también utiliza Common Security Services (CSS), que forma parte de Oracle Platform Security Services (OPSS), para otros aspectos de la seguridad de los mensajes.

3.5.1 Proveedores de JPS

Al utilizar las políticas de Oracle Web Services Manager, se aplica lo siguiente:

- Las políticas de OWSM utilizan proveedores SAML de JPS y no de WebLogic Server. Para obtener información sobre cómo configurar SAML con OWSM, consulte "Configuración de SAML" en Protección de servicios web y administración de políticas con Oracle Web Services Manager .
- Para la autenticación, OWSM utiliza el módulo de inicio de sesión JPS, que a su vez llama a los proveedores de autenticación configurados en WebLogic Server.
- OWSM y Service Bus admiten Java Keystore (JKS) y Keystore Service (KSS) proporcionados por Oracle Platform Security Services. Para las políticas OWSM, una mejor práctica es configurar el almacén de claves en JPS, con el servidor WebLogic y el almacén de claves JPS haciendo referencia al mismo tipo de almacén de claves. Por ejemplo, si utiliza un almacén de claves de archivo JKS, JPS y WebLogic Server deben apuntar al mismo archivo. Si usa un almacén de claves KSS, JPS y WebLogic Server deben apuntar a la misma configuración de KSS.
- Un almacén de claves JPS sirve como almacén de claves y almacén de confianza para las políticas OWSM.

3.5.2 Proveedores de CSS

Service Bus utiliza los siguientes proveedores:

- Proveedores de CSS para hacer cumplir las políticas de WLS 9
- Proveedores de CSS para hacer cumplir la seguridad del transporte
- Proveedores de autorización de WebLogic Server para políticas de autorización
- Proveedores de autenticación personalizados de WebLogic Server y afirmadores de identidad para políticas de autenticación personalizadas
- Proveedores de credenciales y mapeadores de WebLogic Server
- Almacén de claves y almacén de confianza de WebLogic Server para las políticas de WLS 9
- Autenticación y afirmación de identidad a través de agentes de Oracle Web Services Manager

3.5.3 Protección de servicios REST mediante OAuth

A partir de la versión 12.2.1 de OSB se permite la protección de servicios mediante OAuth OWSM, en caso de que estos tengan un endpoint de tipo REST[19].

Esta versión de Service Bus admite casos de uso de OAuth de dos formas, siendo estas: Autenticación de cliente y Autenticación de cliente + usuario.

Autenticación del cliente:

- Credenciales del cliente: nombre de usuario y contraseña de AppID en el encabezado HTTP de autenticación (federated=false) (con y sin SSL)
- Credenciales de cliente: JWT (JSON Web Token) (federated=true) (con y sin SSL)

Autenticación cliente + usuario:

- Credenciales del cliente: nombre de usuario y contraseña de AppID en el encabezado HTTP de autenticación + Credenciales de usuario: JWT (federated=false) (con y sin SSL)
- Identidad de cliente - JWT + Identidad de usuario - JWT (federated=true) (con y sin SSL)

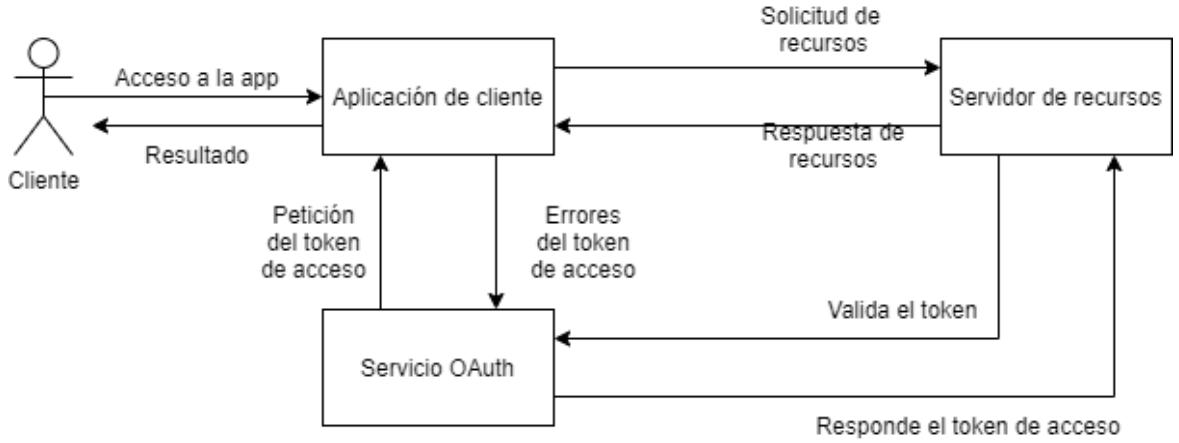


Figura 3.10: Funcionamiento de OAuth

3.6 Funcionalidades

Dentro de este apartado se procederá a definir los procedimientos que se han desarrollado atendiendo a los requisitos solicitados y las futuras necesidades de los interesados.

3.6.1 getStudentsInfo

Este método está destinado a obtener la información referente a los alumnos egresados, es por esto que se han definido cuatro variables para realizar una búsqueda más específica, siendo estos parámetros los siguientes:

- **Curso académico:** Utilizado para indicar el curso académico del cual deseamos obtener la información de egresados.
- **Titulación:** Indica la titulación, ya sea Máster o Grado que se desea obtener.
- **Género:** Indica el género del alumno deseado para poder realizar estudios de género por titulación
- **DNI:** Necesario en caso de solicitar la información de un único alumno para todas sus titulaciones

Cabe resaltar que estos parámetros pueden ser usados de manera simultánea para obtener una respuesta lo más ajustada posible.

3.6.2 getStudentsTotal

Este método está destinado a obtener el número de alumnos egresados por cada titulación de la universidad, es por este motivo que no requiere de parámetros para su correcto funcionamiento.

3.6.3 getStudentsByYear

Este método está destinado a obtener el número de alumnos egresados por cada titulación de la universidad al igual que el método anterior, pero con la peculiaridad de que se añade un parámetro donde se filtra por curso académico:

- **Curso académico:** Utilizado para indicar el curso académico del cual deseamos obtener la información de egresados.

3.6.4 getStudentsGender

Este método está destinado a obtener la estadística de alumnos que posee la universidad por diferencia de sexo, esto destinado a posibles estudios de género, los cuales están al alza. Es por esto antes mencionado que no es necesario establecer ningún parámetro.

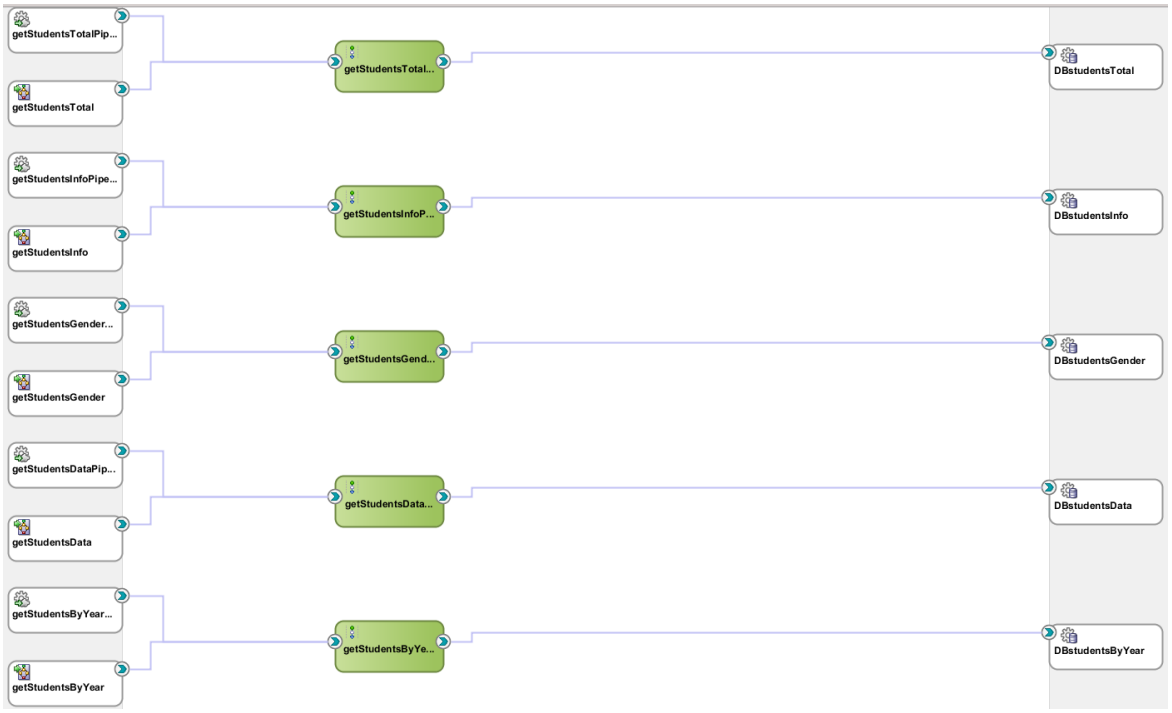


Figura 3.11: Flujo de métodos

3.7 Aplicación de ejemplo

A modo de mostrar una simulación de aplicación en entorno real del servicio se ha creado una página web donde se realizan las consultas y se muestran los resultados obtenidos por la API REST.

3.7.1 Seguridad de la aplicación

En la actualidad el método más aplicado para cifrar contraseñas es un doble MD5, donde se cifra una cadena por medio MD5 y el resultado obtenido es cifrado nuevamente por el mismo método. Esto se hace debido a que MD5 es un sistema muy vulnerable, que permite descifrar una gran cantidad de datos en un tiempo muy corto por medio de la fuerza bruta, siendo que existen páginas web que lo hacen. Como alternativa se ha propuesto la utilización de salt, donde se amplían los tiempos de carga de las contraseñas.

En criptografía, cuando se habla de salt se hace referencia a una cadena aleatoria que se agrega a una palabra de entrada, esto para generar un hash diferente al que se genera solo con la palabra. MD5 realmente no ofrece esta función en el algoritmo criptográfico, pero puede concatenar dos cadenas para obtener el mismo resultado

El principal motivo para utilizar salt en conjunto con MD5, es que favorece la seguridad ante ataques de fuerza bruta. En la siguiente tabla podemos observar el número de intentos necesarios para comprobar todas las posibilidades de contraseña basándonos en el número de caracteres, teniendo en cuenta mayúsculas, minúsculas y números.

Caracteres	Combinaciones
3	3 140.608
4	7.311.616
5	380.204.032
6	19.770.609.664

.

Tabla 3.3: Combinaciones de contraseñas

3.7.2 Autenticación

Debido al concepto del proyecto es necesario establecer un control de los usuarios que van a utilizar la aplicación. Para cumplir este objetivo se ha implementado un sistema de autentificación, donde el usuario inicia sesión mediante un usuario y contraseña. Este sistema almacena los datos dentro de una tabla específica.

Contenido: Este apartado consta de una vista para el inicio de sesión. Esta vista posee un formulario donde se introducirán los datos, además de un texto de enlace para navegar entre ambas vistas.

Funciones: Permite determinar qué usuarios son ya conocidos para la aplicación retornando sus datos y permitiendo establecer las relaciones pertinentes con los otros modelos.

Diseño: En cuanto al diseño se presenta un formulario con los campos necesarios para el acceso del usuario, con un botón que comienza la consulta con la base de datos.

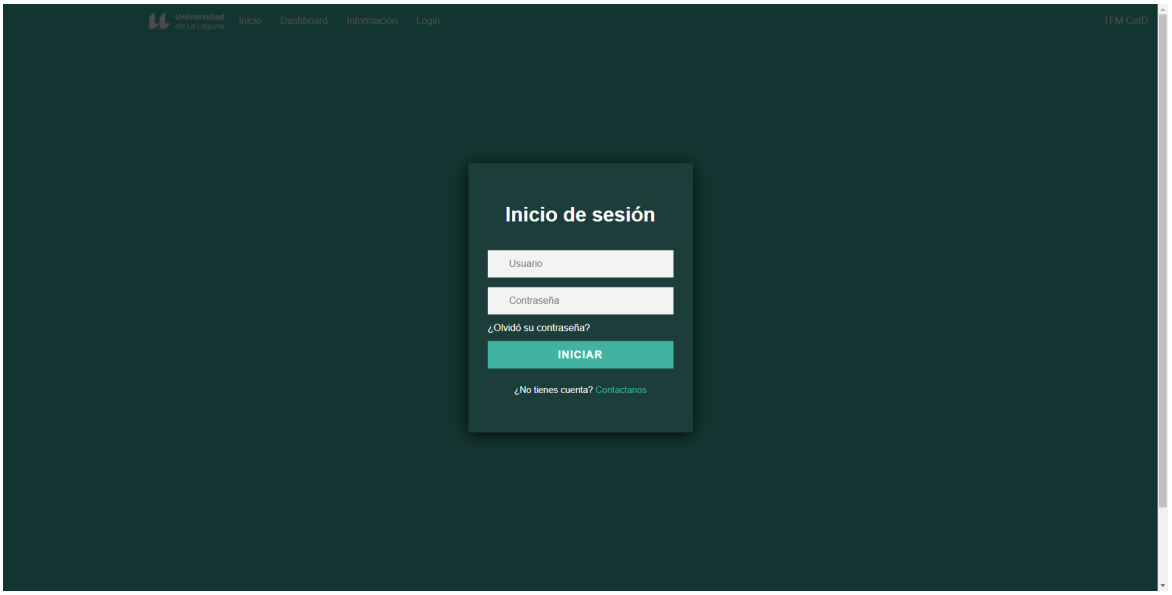


Figura 3.11: Página de inicio

3.7.3 Obtención de datos

El motivo principal de la creación de esta página de apoyo es la representación visual de los datos obtenidos por la API. Para cumplir este objetivo se ha implementado una página web con un filtro donde se seleccionan los parámetros que se desean obtener. Esta página muestra los contenidos en una tabla y permite descargar los datos.

Contenido: Este apartado consta de una vista, donde se muestran un filtro de búsqueda de los parámetros, seguido de un filtro de búsqueda de los contenidos obtenidos, junto con una tabla donde están los datos y finalmente un botón de descarga.

Funciones: Permite obtener los datos deseados especificando los parámetros deseados y una posterior descarga de los mismos.

Diseño: En cuanto al diseño nos encontramos con un filtro de búsqueda sencillo, seguido de una tabla de contenidos y finalmente un botón de descarga.

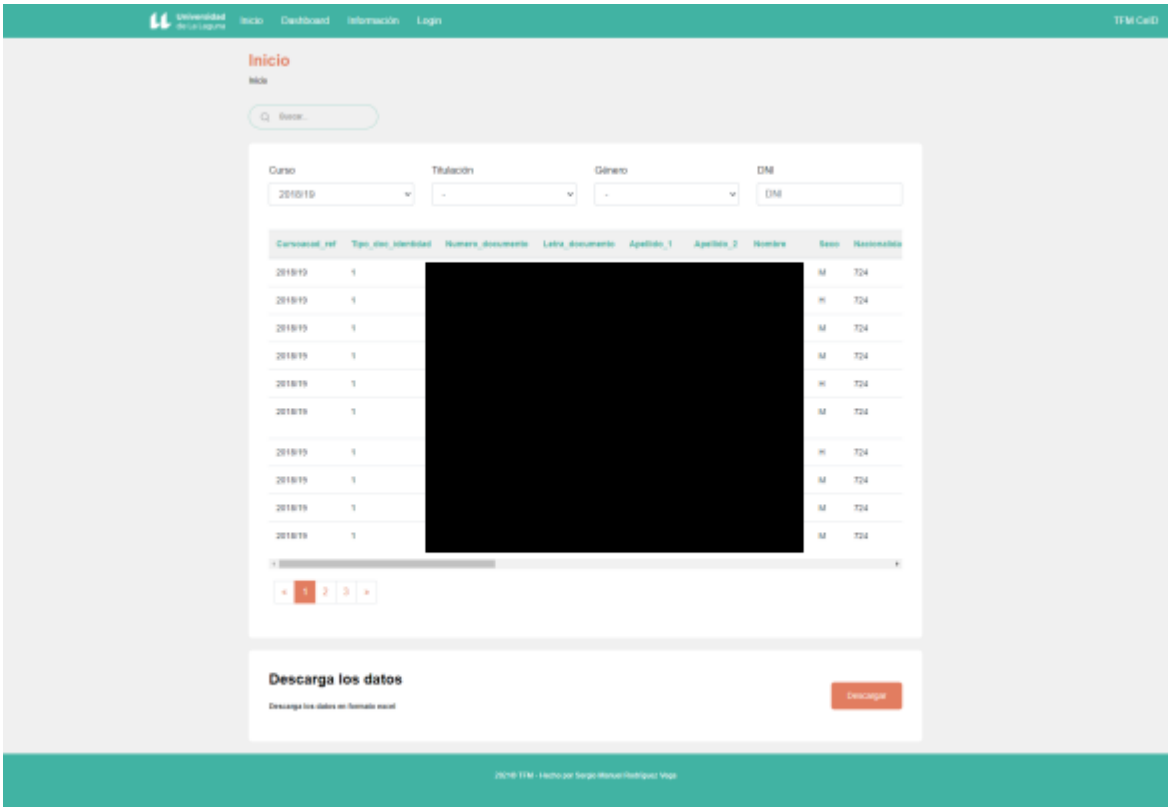


Figura 3.12: Página de búsqueda

3.7.4 Dashboard

El motivo principal de la creación de este apartado es la representación visual de una aplicación de los datos proporcionados por la API. Para cumplir con este objetivo se han implementado varios gráficos representativos de los datos.

Contenido: Este apartado consta de una vista, donde se muestran varios gráficos y paneles de información.

Funciones: Permite visualizar los datos de diversas formas y una posterior descarga de los mismos.

Diseño: En cuanto al diseño nos encontramos con una serie de gráficos, junto con un botón de descarga de datos, seguidos finalmente de paneles de información.

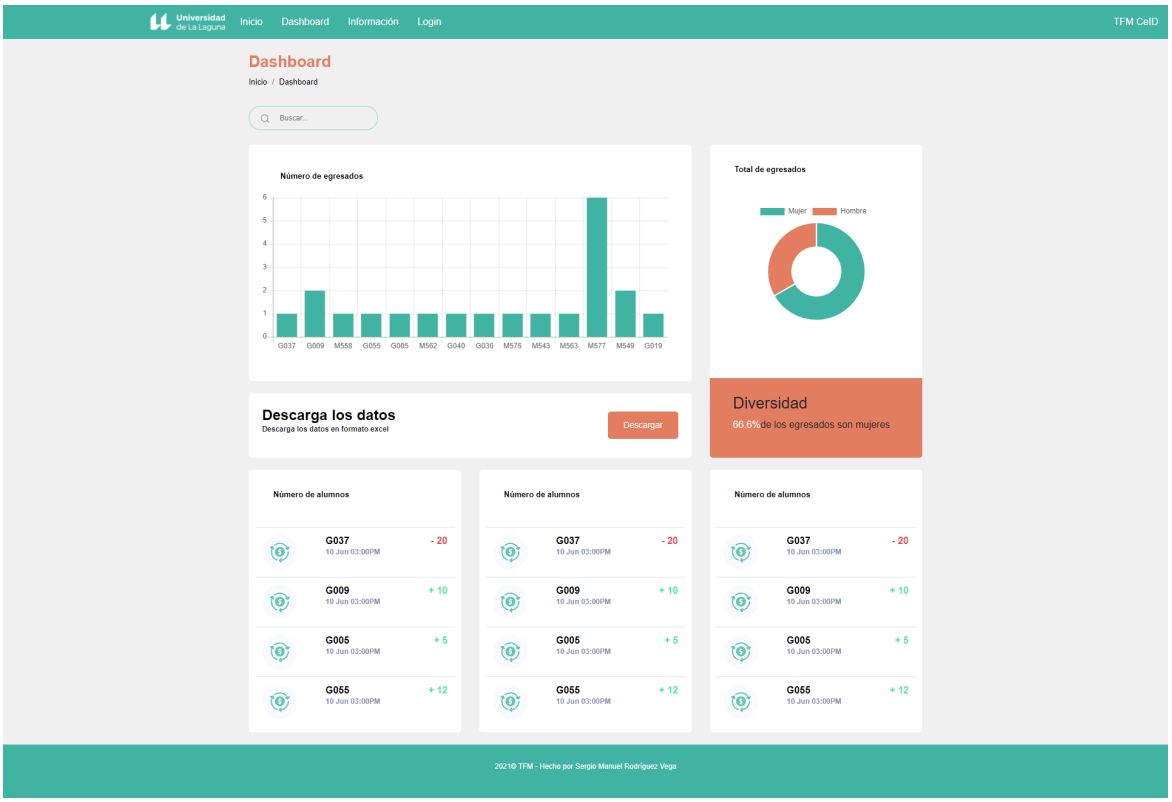


Figura 3.13: Página de dashboard

Capítulo 4: Conclusiones y líneas futuras

Durante el desarrollo del proyecto se ha realizado un estudio, tanto de la diversidad de servicios web que existen, como las nuevas tecnologías aplicadas al desarrollo de los mismos, siendo que por indicaciones del proyecto ya se sabían qué herramientas habían de ser utilizadas. Durante el proceso se han producido una diversidad de conflictos que han desembocado en una gran variedad de cambios, siendo esto una fuente de inspiración para la mejora de las funcionalidades originalmente propuestas. Dicho esto el desarrollo del proyecto se ha llevado a cabo en dos apartados muy distinguidos:

En primer lugar, la configuración de la base de datos, donde se ha recreado una simulación del contenido real que puede ofrecer la Universidad de La Laguna, replicando las bases de datos necesarias y rellenando estas con datos simulados para su posterior uso. La comunicación con esta se realiza mediante un JCA o un DBAdapter, siendo que en este último se configuran las tablas a las que se desea acceder, los parámetros que se van a utilizar y las propias consultas.

Una vez establecida la conexión con la base de datos, se ha procedido al desarrollo del servicio en sí, el cual se ha realizado en su mayoría desde la herramienta JDeveloper, puesto que era uno de los requisitos iniciales, siendo utilizados así los documentos WSDL para la descripción de los servicios y los ficheros XML para la definición de los mismos. Por otro lado se han generado diversos ficheros XQuery para establecer la relación de los datos obtenidos de la base de datos, con la respuesta esperada previamente definida, permitiendo la posibilidad de descartar elementos no deseados, Para finalizar se han definido dos endpoints para la aplicación, dando así como resultado una API en SOAP y otra en REST.

Para finalizar y a modo de establecer una representación más visual del servicio se ha desarrollado una página web desde la cual se puede probar de manera cómoda como sería la aplicación del resultado final.

Basándonos en los objetivos propuestos inicialmente podemos concluir que el proyecto ha sido un total éxito, puesto que no solo se han obtenido los resultados propuestos, sino que se han añadido varias posibles soluciones e incluso añadiendo una página web de apoyo para la cual ha sido necesario un estudio y una serie de test de usuario, para ofrecer una composición y una experiencia amigables.

A pesar de que podemos decir que el proyecto está concluido basándonos en los requisitos propuestos inicialmente, siempre existe un marco de posibles mejoras. Considerando el estado actual del proyecto, una lista de posibles mejoras futuras sería la siguiente:

- **Creación de nuevos perfiles:** En los cuales cada usuario no sea capaz de acceder a los mismos filtros
- **Añadir nuevos métodos:** Para ofrecer una mayor versatilidad del servicio creado
- **Interacción con otros datos del sistema:** Ofreciendo así una mayor posibilidad de interoperabilidad entre los datos
- **Desarrollo de un panel o dashboard:** Para obtener una representación gráfica de la evolución de los datos

Capítulo 5: Summary and Conclusions

During the development of the project, a study has been carried out, both of the diversity of web services that exist, and the new technologies applied to their development, being that by indications of the project it was already known which tools were to be used. During the process there have been a variety of conflicts that have resulted in a great variety of changes, this being a source of inspiration for the improvement of the originally proposed functionalities. Having said this, the development of the project has been carried out in two very distinguished sections:

In the first place, the configuration of the database, where a simulation of the real content that the University of La Laguna can offer has been recreated, replicating the necessary databases and filling these with simulated data for later use. Communication with this is done through a JCA or a DBAdapter, and in the latter the tables to be accessed, the parameters to be used and the queries themselves are configured.

Once the connection with the database has been established, the service itself has been developed, which has been carried out mostly from the JDeveloper tool, since it was one of the initial requirements, thus using the WSDL documents to describe the services and the XML files for their definition. On the other hand, several XQuery files have been generated to establish the relationship of the data obtained from the database, with the previously defined expected response, allowing the possibility of discarding unwanted elements. To finish, two endpoints have been defined for the application, thus resulting in an API in SOAP and another in REST.

Finally, and in order to establish a more visual representation of the service, a web page has been developed from which you can comfortably test how the application of the final result would be.

Based on the initially proposed objectives we can conclude that the project has been a total success, since not only have the proposed results been obtained, but several possible solutions have been added and even adding a support web page for which it has been necessary a study and a series of user tests, to offer a friendly composition and experience.

Although we can say that the project is concluded based on the initially proposed requirements, there is always a framework of possible improvements. Considering the current state of the project, a list of possible future improvements would be as follows:

- **Creation of new profiles:** In which each user is not able to access the same filters
- **Add new methods:** To offer greater versatility of the service created
- **Interaction with other system data:** thus offering a greater possibility of interoperability between the data
- **Development of a panel or dashboard:** To obtain a graphical representation of the evolution of the data

Capítulo 6: Presupuesto

Dadas las características del presente proyecto, se expone el presupuesto adaptado a la duración del mismo con una estimación de tres meses, desarrollado por un único trabajador.

A continuación se presenta el desglose del presupuesto detallando, tanto la labor personal del desarrollo, así como los elementos tanto físicos, como digitales necesarios para el correcto desempeño del mismo.

6.1 Mobiliario

Este apartado se encuentra destinado a definir el material físico necesario para un adecuado puesto de trabajo para el desarrollador. Aquí se encuentra el material de oficina básico para la definición de un puesto de desarrollo estándar.

Tipo	Unidades	Precio(Unidad)	Total
Escritorio	1	360€	360€
Silla	1	140€	140€
Ordenador	1	800€	800€
Monitor	1	200€	200€
Periféricos	1	100€	100€
Total			1600€

Tabla 6.1: Presupuesto del mobiliario

6.2 Herramientas

Este apartado se encuentra destinado a definir el material digital necesario para un adecuado puesto de trabajo para el desarrollador. Aquí se encuentran los programas necesarios.

Tipo	Licencia	Total
Eclipse	GRATUITA	0,00€
SQLDeveloper	GRATUITA	0,00€
Oracle Database	Oracle Database Enterprise Edition	3139,73€
Jdeveloper	GRATUITA	0,00€
Total		3139,73€

Tabla 6.2: Presupuesto de las herramientas

6.3 Suministro y similares

Este apartado se encuentra destinado a definir los suministros necesarios para el correcto desarrollo del proyecto.

Tipo	Duración(Meses)	Precio(Mes)	Total
Alquiler	3	500€	1500,00€
Electricidad	3	50€	150,00€
Internet	3	100€	300,00€
Hosting	3	0€	0,00€
Total			1950,00€

Tabla 6.3: Presupuesto de los suministros

6.4 Salario

Este apartado se encuentra destinado a definir los costes de contratación del desarrollador, teniendo en cuenta que este será un programador junior, con poca experiencia laboral.

Tipo	Horas	Precio(Hora)	Total
Análisis	130	42€	5460,00€
Desarrollo	350	32€	11200,00€
Total			16660,00€

Tabla 6.4: Presupuesto del salario

6.5 Presupuesto final

Presupuesto final, obtenido de la suma de los sub-apartados anteriormente expuestos, necesarios para el correcto desarrollo del servicio.

Descripción	Total
Mobiliario	1600,00€
Herramientas	3139,73€
Suministros	1950,00€
Salario	16660,00€
Total	23349,73€

Tabla 6.5: Presupuesto final

Apéndice A

Código fuente

A.1 Adaptador de la base de datos

```
/*****************************************************************************
*
* DBReference_db.jca
*
*****************************************************************************
*
* AUTOR: Sergio Manuel Rodríguez Vega
*
*
* FECHA: 06/03/2021
*
*
* DESCRIPCIÓN: Código fuente del adaptador de la base de datos
*
*
*****************************************************************************/

<adapter-config name="DBReference" adapter="db"
wsdlLocation="DBReference.wsdl"
xmlns="http://platform.integration.oracle/blocks/adapter/fw/metadata">

  <connection-factory UIConnectionName="TFMdb" location="eis/db/tfm"/>
  <endpoint-interaction portType="DBReference_ptt"
operation="DBReferenceSelect">
    <interaction-spec className="oracle.tip.adapter.db.DBReadInteractionSpec">
      <property name="DescriptorName" value="DBReference.Egresados"/>
      <property name="QueryName" value="DBReferenceSelect"/>
      <property name="MappingsMetaDataURL"
value="DBReference-or-mappings.xml"/>
      <property name="ReturnSingleResultSet" value="false"/>
      <property name="GetActiveUnitOfWork" value="false"/>
    </interaction-spec>
  </endpoint-interaction>

</adapter-config>
```

A.2 Query de la petición

```
/**************************************************************************************
*
* alumnoRequest.xqy
*
**************************************************************************************
*
* AUTOR: Sergio Manuel Rodríguez Vega
*
*
* FECHA: 01/04/2021
*
*
* DESCRIPCIÓN: Código de la petición
*
*
**************************************************************************************/

xquery version "1.0" encoding "utf-8";

(:: OracleAnnotationVersion "1.0" ::)

declare namespace ns1="http://www.example.org";
(:: import schema at "../Resources/dataTypes.xsd" ::)
declare namespace
ns2="http://xmlns.oracle.com/pcbpel/adapter/db/top/DBReference";
(:: import schema at "../Business/DBReference_table.xsd" ::)

declare variable $xq_in as element() (:: schema-element(ns1:Request) ::)
external;

declare function local:func($xq_in as element() (:: schema-element(ns1:Request)
::)) as element() (::
schema-element(ns2:DBReferenceSelect_titulacion_curso_dni_sexoInputParame
ters) ::) {
   <ns2:DBReferenceSelect_titulacion_curso_dni_sexoInputParameters>
      <ns2:titulacion>{fn:data($xq_in/ns1:titulacion)}</ns2:titulacion>
      <ns2:curso>{fn:data($xq_in/ns1:curso)}</ns2:curso>
      <ns2:dni>{fn:data($xq_in/ns1:dni)}</ns2:dni>
      <ns2:sexo>{fn:data($xq_in/ns1:sexo)}</ns2:sexo>
   </ns2:DBReferenceSelect_titulacion_curso_dni_sexoInputParameters>
};
```

```
local:func($xq_in)
```

A.3 Query de la respuesta

```
/**************************************************************************************
*
* alumnoResponse.xqy
*
**************************************************************************************
*
* AUTOR: Sergio Manuel Rodríguez Vega
*
*
* FECHA: 01/04/2021
*
*
* DESCRIPCIÓN: Código de la respuesta
*
*
**************************************************************************************/

xquery version "1.0" encoding "utf-8";

(:: OracleAnnotationVersion "1.0" ::)

declare namespace ns1="http://www.example.org";
(:: import schema at "../Resources/dataTypes.xsd" ::)
declare namespace
ns2="http://xmlns.oracle.com/pcbpel/adapter/db/top/DBReference";
(:: import schema at "../Business/DBReference_table.xsd" ::)

declare variable $xq_out as element() (::
schema-element(ns2:EgresadosCollection) ::) external;

declare function local:func($xq_out as element() (::
schema-element(ns2:EgresadosCollection) ::)) as element() (:: element(*,
ns1:Egresados) ::) {
   <ns1:Egresados>
      {
         if ($xq_out/ns2:Egresados/ns2:cursoacadRef)
         then
<ns1:cursoacadRef>{fn:data($xq_out/ns2:Egresados/ns2:cursoacadRef)}</ns1:cu
rsoacadRef>
```

```
            else ()
        }
        {
            if ($xq_out/ns2:Egresados/ns2:tipoDocIdentidad)
            then
<ns1:tipoDocIdentidad>{fn:data($xq_out/ns2:Egresados/ns2:tipoDocIdentidad)}</
ns1:tipoDocIdentidad>
            else ()
        }

<ns1:numeroDocumento>{fn:data($xq_out/ns2:Egresados/ns2:numeroDocument
o)}</ns1:numeroDocumento>

<ns1:letraDocumento>{fn:data($xq_out/ns2:Egresados/ns2:letraDocumento)}</ns
1:letraDocumento>
        {
            if ($xq_out/ns2:Egresados/ns2:apellido1)
            then
<ns1:apellido1>{fn:data($xq_out/ns2:Egresados/ns2:apellido1)}</ns1:apellido1>
            else ()
        }
        {
            if ($xq_out/ns2:Egresados/ns2:apellido2)
            then
<ns1:apellido2>{fn:data($xq_out/ns2:Egresados/ns2:apellido2)}</ns1:apellido2>
            else ()
        }
        {
            if ($xq_out/ns2:Egresados/ns2:nombre)
            then
<ns1:nombre>{fn:data($xq_out/ns2:Egresados/ns2:nombre)}</ns1:nombre>
            else ()
        }
        {
            if ($xq_out/ns2:Egresados/ns2:sexo)
            then <ns1:sexo>{fn:data($xq_out/ns2:Egresados/ns2:sexo)}</ns1:sexo>
            else ()
        }
        {
            if ($xq_out/ns2:Egresados/ns2:nacionalidad)
            then
<ns1:nacionalidad>{fn:data($xq_out/ns2:Egresados/ns2:nacionalidad)}</ns1:naci
onalidad>
            else ()
        }
```

```
    {
        if ($xq_out/ns2:Egresados/ns2:paisNacimiento)
        then
<ns1:paisNacimiento>{fn:data($xq_out/ns2:Egresados/ns2:paisNacimiento)}</ns1
:paisNacimiento>
        else ()
    }
    {
        if ($xq_out/ns2:Egresados/ns2:fechaNacimiento)
        then
<ns1:fechaNacimiento>{fn:data($xq_out/ns2:Egresados/ns2:fechaNacimiento)}</
ns1:fechaNacimiento>
        else ()
    }
    {
        if ($xq_out/ns2:Egresados/ns2:universidad)
        then
<ns1:universidad>{fn:data($xq_out/ns2:Egresados/ns2:universidad)}</ns1:univers
idad>
        else ()
    }

<ns1:codTitulacion>{fn:data($xq_out/ns2:Egresados/ns2:codTitulacion)}</ns1:cod
Titulacion>
    {
        if ($xq_out/ns2:Egresados/ns2:fechaccesosue)
        then
<ns1:fechaccesosue>{fn:data($xq_out/ns2:Egresados/ns2:fechaccesosue)}</ns1:
fechaccesosue>
        else ()
    }
    {
        if ($xq_out/ns2:Egresados/ns2:fechaccesoest)
        then
<ns1:fechaccesoest>{fn:data($xq_out/ns2:Egresados/ns2:fechaccesoest)}</ns1:f
echaccesoest>
        else ()
    }
    {
        if ($xq_out/ns2:Egresados/ns2:modaccesoest)
        then
<ns1:modaccesoest>{fn:data($xq_out/ns2:Egresados/ns2:modaccesoest)}</ns1:
modaccesoest>
        else ()
    }
```

```
{
    if ($xq_out/ns2:Egresados/ns2:notaingresoest)
    then
<ns1:notaingresoest>{fn:data($xq_out/ns2:Egresados/ns2:notaingresoest)}</ns1:
notaingresoest>
    else ()
}
{
    if ($xq_out/ns2:Egresados/ns2:fechaegreso)
    then
<ns1:fechaegreso>{fn:data($xq_out/ns2:Egresados/ns2:fechaegreso)}</ns1:fecha
egreso>
    else ()
}
{
    if ($xq_out/ns2:Egresados/ns2:notaegresoest4)
    then
<ns1:notaegresoest4>{fn:data($xq_out/ns2:Egresados/ns2:notaegresoest4)}</ns
1:notaegresoest4>
    else ()
}
{
    if ($xq_out/ns2:Egresados/ns2:notaegresoest10)
    then
<ns1:notaegresoest10>{fn:data($xq_out/ns2:Egresados/ns2:notaegresoest10)}</
ns1:notaegresoest10>
    else ()
}
{
    if ($xq_out/ns2:Egresados/ns2:estupadre)
    then
<ns1:estupadre>{fn:data($xq_out/ns2:Egresados/ns2:estupadre)}</ns1:estupadre
>
    else ()
}
{
    if ($xq_out/ns2:Egresados/ns2:estumadre)
    then
<ns1:estumadre>{fn:data($xq_out/ns2:Egresados/ns2:estumadre)}</ns1:estumad
re>
    else ()
}
{
    if ($xq_out/ns2:Egresados/ns2:ocupadre)
    then
```

```
<ns1:ocupadre>{fn:data($xq_out/ns2:Egresados/ns2:ocupadre)}</ns1:ocupadre>
            else ()
        }
        {
            if ($xq_out/ns2:Egresados/ns2:ocumadre)
            then
<ns1:ocumadre>{fn:data($xq_out/ns2:Egresados/ns2:ocumadre)}</ns1:ocumadre
>
            else ()
        }
        {
            if ($xq_out/ns2:Egresados/ns2:trabajoalum)
            then
<ns1:trabajoalum>{fn:data($xq_out/ns2:Egresados/ns2:trabajoalum)}</ns1:trabaj
oalum>
            else ()
        }
        {
            if ($xq_out/ns2:Egresados/ns2:munifam)
            then
<ns1:munifam>{fn:data($xq_out/ns2:Egresados/ns2:munifam)}</ns1:munifam>
            else ()
        }
        {
            if ($xq_out/ns2:Egresados/ns2:codpostfam)
            then
<ns1:codpostfam>{fn:data($xq_out/ns2:Egresados/ns2:codpostfam)}</ns1:codpo
stfam>
            else ()
        }
        {
            if ($xq_out/ns2:Egresados/ns2:paisfam)
            then
<ns1:paisfam>{fn:data($xq_out/ns2:Egresados/ns2:paisfam)}</ns1:paisfam>
            else ()
        }
    </ns1:Egresados>
};

local:func($xq_out)
```

A.4 Adaptador REST

```
/*************************************************************************************
*
* getStudentsPipeline.wdsl
*
*************************************************************************************
*
* AUTOR: Sergio Manuel Rodríguez Vega
*
*
* FECHA: 10/05/2021
*
*
* DESCRIPCIÓN: Código del pipeline de un adaptador REST
*
*
*************************************************************************************/

   <wsdl:types>
      <xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema">
         <xsd:import
namespace="http://xmlns.oracle.com/pcbpel/adapter/db/DBstudentsTotal"
schemaLocation="../Resources/DBstudentsTotal.xsd"/>
      </xsd:schema>
   </wsdl:types>
   <wsdl:message name="requestMessage">
      <wsdl:part name="in" element="inp1:DBstudentsTotalInput"/>
   </wsdl:message>
   <wsdl:message name="replyMessage">
      <wsdl:part name="out" element="inp1:DBstudentsTotalOutputCollection"/>
   </wsdl:message>
   <wsdl:portType name="execute_ptt">
      <wsdl:operation name="execute">
         <wsdl:input message="tns:requestMessage"/>
         <wsdl:output message="tns:replyMessage"/>
      </wsdl:operation>
   </wsdl:portType>
   <wsdl:binding name="execute_bind" type="tns:execute_ptt">
      <soap:binding transport="http://schemas.xmlsoap.org/soap/http"/>
      <wsdl:operation name="execute">
         <soap:operation style="document" soapAction="execute"/>
         <wsdl:input>
            <soap:body use="literal"
namespace="http://xmlns.oracle.com/TFM-SB/ull-alumno/getStudentsTotalPi"/>
```

Bibliografía

[1] Charles J. Petrie. Web Service Composition - enlace [06/03/2021]
[2] Michel, Jason Paul. Web Service APIs and Libraries - enlace [06/03/2021]
[3] D. Box, D. Ehnebuske, G. Kakivaya, A. Layman, N. Mendelsohn, H. Nielsen, S. Thatte, and D. Winer, "Simple Object Access Protocol (SOAP) 1.1, W3C Note 08 May 2000," - enlace [06/03/2021]
[4] E. T. Ray, Learning XML: creating self-describing data - enlace [06/03/2021]
[5] D. Box, "Inside SOAP" - enlace [06/03/2021]
[6] A. Ryman, "Understanding web services" - enlace [06/03/2021]
[7] Wikipedia. "Representational State Transfer" - enlace [06/03/2021]
[8] R. Fielding, "Architectural Styles and the Design of Network-based Software Architectures" - enlace [06/03/2021]
[9] IETF, "RFC 1945" - enlace [06/03/2021]
[10] Eclipse, Página oficial - enlace [09/03/2021]
[11] Wikipedia. "Eclipse (Software)" - enlace [09/03/2021]
[12] Oracle. "¿Qué es SQL Developer?" - enlace [18/03/2021]
[13] W3C. "Página oficial" - enlace [18/03/2021]
[14] Datta, Sudip. Building and Managing a Cloud Using Oracle Enterprise Manager 12c - enlace [19/03/2021]
[15] Edwin Biemond ; Eric Elzinga ; Mischa Kölliker ; Guido Schmutz ; Jan van Zoggel, "Oracle Service Bus 11g Development Cookbook" - enlace [30/03/2021]
[16] Oracle, "Understanding Technology Adapters" - enlace [31/03/2021]
[17] IBM doc, "WDSL" - enlace [12/04/2021]
[18] Oracle, "Securing Oracle Service Bus with Oracle Web Services Manager" - enlace [12/04/2021]
[19] Oracle, "Protección de Oracle Service Bus con Oracle Web Services Manager" - enlace [12/04/2021]
[20] Wikipedia, "WSDL" - enlace [20/04/2021]