PBtoWS - Procesos: Arquitectura Backend (Powerbuilder)

1.1 Proposito

Este Documento de Arquitectura de Software presenta la arquitectura del Framework **SICOF PBtoWS** a través de diferentes vistas, con el fin de ilustrar la composición del API [1)] que lo conforma.

1.2 Alcance

Este documento se centra en el desarrollo de la vista lógica del framework. Se incluyen los aspectos fundamentales del resto de las vistas y se omiten aquellas que no se consideren pertinentes como ser el caso de la vista de procesos. En cuanto a los componentes externos que se mencionen, se incluye una descripción de los mismos en el nivel que se considere apropiado y se indican las referencias donde consultar más información sobre los mismos.

1.3 Definiciones, Acrónimos y Abreviaciones

- **Powerbuilder**: PowerBuilder es una herramienta de desarrollo de clase empresarial desarrollada por la empresa PowerSoft. PowerBuilder es orientada a objetos y permite el desarrollo de diferentes tipos de aplicaciones y componentes para ejecutar arquitecturas cliente/servidor, distribuidas y Web. PowerBuilder - Wikipedia, la enciclopedia libre
- **PBL**[2)] **- Libreria Powerbuilder**: Es un archivo contenedor en el cual se agregan recursos utilizados en los proyectos powerbuilder como clase (objetos no visuales), datawindows, estructuras, funciones y objetos querys.
- **Framework**: En el desarrollo de software, un framework o entorno de trabajo es una estructura conceptual y tecnológica de asistencia definida, normalmente, con artefactos o módulos concretos de software, que puede servir de base para la organización y desarrollo de software. Típicamente, puede incluir soporte de programas, bibliotecas, y un lenguaje interpretado, entre otras herramientas, para así ayudar a desarrollar y unir los diferentes componentes de un proyecto. Representa una arquitectura de software que modela las relaciones generales de las entidades del dominio, y provee una estructura y una especial metodología de trabajo, la cual extiende o utiliza las aplicaciones del dominio. Fuente: Wikipedia
- **Clase Lanzadora**: Es la clase que permite la exposición de las operaciones SOAP de los procesos en powerbuilder, no debe contener códificación diferente a ese tipo de proceso.
- **Libreria Lanzadora**: Es el paquete que contiene el conjunto de clases que exponen las operaciones SOAP.
- **Clase Invocadora**: Es la clase que resuelve y orquesta las invocaciones de las operaciones de los servicios SOAP, no debe contener códificación diferente a ese tipo de proceso.
- **Libreria Invocadora**: Es el paquete que contiene el conjunto de clases que invocan y resuelven las operaciones SOAP con los procesos del código Powerbuilder.
- **Modelo Orientado a Eventos**: La programación dirigida por eventos es un paradigma de programación en el que tanto la estructura como la ejecución de los programas van

determinados por los sucesos que ocurran en el sistema, definidos por el usuario o que ellos mismos provoquen. Para entender la programación dirigida por eventos, podemos oponerla a lo que no es: mientras en la programación secuencial (o estructurada) es el programador el que define cuál va a ser el flujo del programa, en la programación dirigida por eventos será el propio usuario —o lo que sea que esté accionando el programa— el que dirija el flujo del programa. Aunque en la programación secuencial puede haber intervención de un agente externo al programa, estas intervenciones ocurrirán cuando el programador lo haya determinado, y no en cualquier momento como puede ser en el caso de la programación dirigida por eventos. Programación dirigida por eventos - Wikipedia, la enciclopedia libre

- **Modelo Orientado a Objetos**: La programación orientada a objetos (POO, en español; OOP, según sus siglas en inglés) es un paradigma de programación que viene a innovar la forma de obtener resultados. Los objetos manipulan los datos de entrada para la obtención de datos de salida específicos, donde cada objeto ofrece una funcionalidad especial. Muchos de los objetos prediseñados de los lenguajes de programación actuales permiten la agrupación en bibliotecas o librerías, sin embargo, muchos de estos lenguajes permiten al usuario la creación de sus propias bibliotecas. Programación orientada a objetos - Wikipedia, la enciclopedia libre
- **Core Powerbuilder**: Corresponde al código powerbuilder que representa los procesos de **SICOF ERP** que van a ser migrados del modelo orientado a eventos al módelo orientado a objetos.
- **Modelo de Red Punto a Punto**: Las redes punto a punto son aquellas que responden a un tipo de arquitectura de red en las que cada canal de datos se usa para comunicar únicamente dos nodos, en clara oposición a las redes multipunto, en las cuales cada canal de datos se puede usar para comunicarse con diversos nodos. Red punto a punto - Wikipedia, la enciclopedia libre
- **Factory Method**: En diseño de software, el patrón de diseño Factory Method consiste en utilizar una clase constructora (al estilo del Abstract Factory) abstracta con unos cuantos métodos definidos y otro(s) abstracto(s): el dedicado a la construcción de objetos de un subtipo de un tipo determinado. Es una simplificación del Abstract Factory, en la que la clase abstracta tiene métodos concretos que usan algunos de los abstractos; según usemos una u otra hija de esta clase abstracta, tendremos uno u otro comportamiento. Factory Method (patrón de diseño) - Wikipedia, la enciclopedia libre
- **JSON (acrónimo de JavaScript Object Notation, «notación de objeto de JavaScript»)**: Es un formato de texto sencillo para el intercambio de datos. Se trata de un subconjunto de la notación literal de objetos de JavaScript, aunque, debido a su amplia adopción como alternativa a XML, se considera (año 2019) un formato independiente del lenguaje. JSON - Wikipedia, la enciclopedia libre
- **SOAP (originalmente las siglas de Simple Object Access Protocol)**: es un protocolo estándar que define cómo dos objetos en diferentes procesos pueden comunicarse por medio de intercambio de datos XML. Simple Object Access Protocol - Wikipedia, la enciclopedia libre
- **SOA** - Arquitectura orientada a servicios - Wikipedia, la enciclopedia libre: La Arquitectura Orientada a Servicios (SOA, siglas del inglés Service Oriented Architecture) es un estilo de arquitectura de TI que se apoya en la orientación a servicios. La orientación a servicios es una forma de pensar en servicios, su construcción y sus resultados. Un servicio es una representación lógica de una actividad de negocio que tiene un resultado de negocio específico (ejemplo: comprobar el crédito de un cliente, obtener datos de clima, consolidar reportes de perforación)
- **Datawindow** [3]:El control DataWindow de PowerBuilder es un contenedor para objetos DataWindow en una aplicación de PowerBuilder. Puede usarlo en una ventana para presentar una visualización interactiva de datos. El usuario puede ver y cambiar datos y enviar cambios a la base de datos. Además del control DataWindow, el objeto DataStore proporciona un

contenedor no visual para aplicaciones de servidor y otras situaciones en las que no es necesario verlas en pantalla. DataWindow admite la recuperación de datos con argumentos de recuperación y actualización de datos. Puede usar estilos de edición, formatos de visualización y reglas de validación para una entrada y visualización de datos coherentes. El DataWindow proporciona muchos métodos para manipular el DataWindow, incluyendo Modificar para cambiar las propiedades del objeto DataWindow. Puede compartir un conjunto de resultados entre varios controles de DataWindow y puede sincronizar los datos entre un cliente y un servidor.

- **SICOF ERP**: Es el sistema de planificación de recursos empresarial (ERP, por sus siglas en inglés, enterprise resource planning) de la empresa ADA el cual provee los modulos Financiero, Inventarios Activos y Gestión Empresarial. Ademas de un abanico de aplicaciones que abarcan procesos de gestión de recursos hasta aplicaciones móviles.
- **SICOF PBtoWS**: Framework o marco de trabajo que expone el código migrado del modelo orientado a evento al modelo orientado a objetos de los módulos de **SICOF ERP**
- **Componente WS-SOAP SICOF ERP**: Es un componente expuesto en servicio web SOAP (.Net) que contiene un conjunto de funcionalidades que representan procesos de **SICOF ERP**

1.4 Organización del Documento

El documento se desarrolla y organiza en base a la plantilla elaborada para el artefacto **DOCUMENTO DE ARQUITECTURA SICOF.docx** del proceso de certificación CMMI Dev Nivel 3[4)], adaptada a las características particulares del tipo de proyecto en desarrollo.

La sección 2 realiza una introducción a la representación utilizada de la arquitectura de forma de asegurar una comprensión general del documento en tal sentido. Las siguientes secciones se abocan a la descripción de la arquitectura del Framework **SICOF PBtoWS**. Luego de una descripción inicial de los objetivos y restricciones influyentes, se desarrolla cada una de las vistas y se cierra con algunas consideraciones finales importantes. En las secciones finales, y sobre la base de lo desarrollado anteriormente, se incluye la descripción de la general de la arquitectura de un módulo de **SICOF ERP**[5)].

2 Representación de la Arquitectura

El modelo propuesto en el documento **DOCUMENTO DE ARQUITECTURA SICOF.docx** para representar la arquitectura del Framework **SICOF PBtoWS** utiliza el siguiente conjunto de vistas[6)]:

- **Vista Lógica:** describe las partes arquitectónicamente significativas del modelo de diseño, como ser la descomposición en capas, subsistemas o paquetes. Una vez presentadas estas unidades lógicas principales, se profundiza en ellas hasta el nivel que se considere adecuado.
- **Vista de Procesos:** describe la descomposición del sistema en threads y procesos pesados. Indica que procesos o grupos de procesos se comunican o interactúan entres sí y los modos en que estos se comunican.
- **Vista de Deployment:** describe uno o más escenarios de distribución física del sistema sobre los cuales se ejecutará y hará el deploy del mismo. Muestra la comunicación entre los diferentes nodos que componen los escenarios antes mencionados, así como el mapeo de los elementos de la Vista de Procesos en dichos nodos.
- **Vista de Implementación:** describe la estructura general del Modelo de Implementación y el mapeo de los subsistemas, paquetes y clases de la Vista Lógica a subsistemas y componentes

de implementación.
- **Vista de Datos:** describe los elementos principales del Modelo de Datos, brindando un panorama general de dicho modelo en términos de tablas, vistas, índices, etc.

3 Objetivos y Restricciones

El Framework **SICOF PBtoWS** basa el desarrollo en propiedades esenciales para la arquitectura a definir:

- diseño basado en componentes de propósito claro y concreto y con alto grado de cohesión,
- desacoplamiento entre componentes que permita el fácil reemplazo de los mismos,
- componentes altamente reutilizables,
- patron de diseño MVC[7)]

3.1 Restricciones y Limitaciones

El Framework **SICOF PBtoWS** presenta las siguientes reestricciones y limitaciones de acuerdo a la tecnólogia que lo soporta:

- No se soportan los servicios web RESTful
- El formato Json es soportado de forma no nativa por lo tanto se afecta el rendimiento y procesamiento en las respuestas de consumo especialmente en los servicios de retorno de bloques de información Ej: Servicios CRUD[8)] :READ
- La versión de la tecnología utilizada Powerbuilder [9)] 12.5 solo soporta hasta el framework .Net 4.02
- El framework **SICOF PBtoWS** solo puede exponer servicios en arquitecturas Windows 7 ó Superior y Servidores IIS 7.5 ó superior

3.2 Requerimientos Funcionales

- Soportar la gestion de transacciones por multiples empresas (multiempresa)
- Soportar la gestión integral y unificada de usuarios y roles
- Registrar la trazabilidad de acciones de los usuarios en sesión *

3.3 Requerimientos Suplementarios (No Funcionales)

- Seguridad
- Reusabilidad
- Concurrencia
- Facilidad de instalación y despliegue
- Facilidad de pruebas

3.4 Requerimientos Especiales

3.4.1 Interoperabilidad

- El framework debe soportar la capacidad de interoperar con servicios web externos (WS-SOAP) a nivel de datos y procesos.

4 Vista Lógica

4.1 Introducción

La vista lógica presenta al sistema [10)] como un todo, indicando en términos propios de la tecnología utilizada, las partes que lo forman y las relaciones principales entre ellas.

Este refinamiento consiste en descomponenter la vista en subsistemas. Los subsistemas representan cortes verticales al diseño del sistema. Cada subsistema consiste en el agrupamiento de diferentes funcionalidades relacionadas entre sí y posee la capacidad de funcionar como un sistema en sí mismo.

4.2 Descomposición en Subsistema

La descomposición utilizada en el **Framework SICOF PBtoWS** se basa en el modelo de comunicación punto a punto [11)] el cual permite organizar la arquitectura en conjuntos de funcionalidades agrupadas (en PBL) que interactúan entre si para cumplir las funciones requeridas.

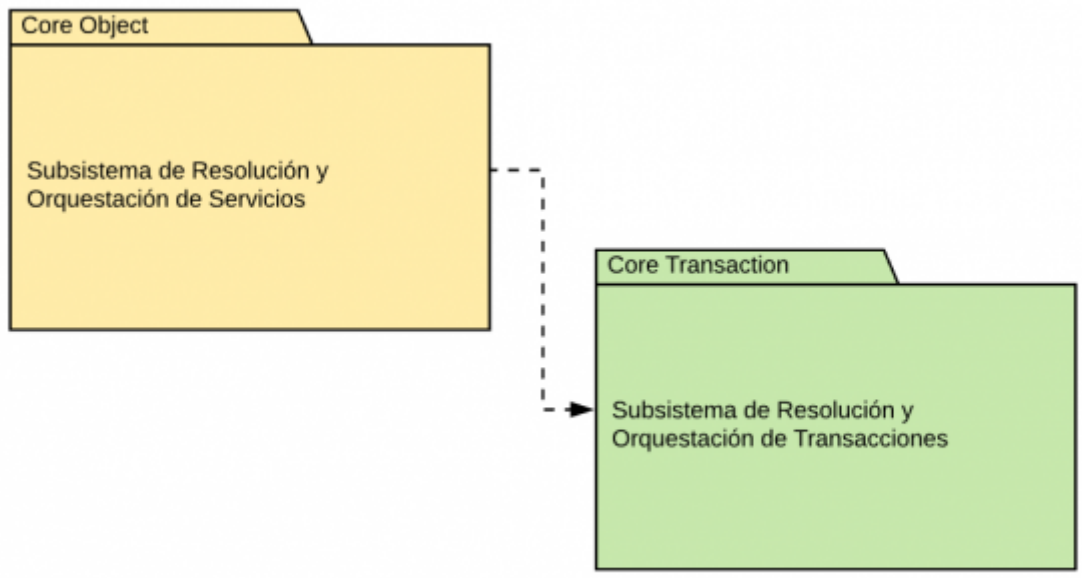

Figura 1: Principales componentes del Framework SICOF PBtoWS

4.3 Descripción de los Subsistemas

- **Resolución y Orquestación de Servicios**: Este subsistema se encarga de identificar procesos y genera invocaciones por medio de eventos internos. Para lograrlo se apoya del patron **Factory Method** que crea instancias de clases que orquestan los llamados de clases controladoras, provee la capa de seguridad de acceso al sistema y controla los accesos por medio de hash de sesión. Además se encarga de formatear los argumentos de entrada y salida de las operaciones en formato Json.
- **Resolución y Orquestación de Transacciones**: Este subsistema se encarga de resolver y entregar las transacciones[12)] de las empresas que realiza los procesos.

4.4 Diseño de Subsistemas

4.4.1 Definición de Procesos

A continuación se describen los subsistemas del **framework SICOF PBtoWS** basandose en representaciones intermedias sin profundizar en procesos del negocio. Ya que la salida de trabajo de los subsistemas serán insumos para capas externas que no hacen parte de la implementación.

4.4.2 Resolución y Orquestación de Servicios

El subsistema **Resolución y Orquestación de Servicios** tiene tres componentes como se puede observar en la siguiente imagen:

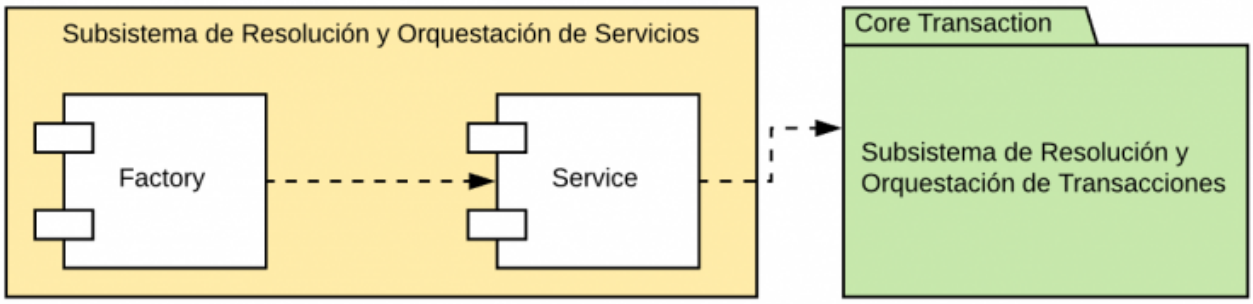

Figura 2: Subsistema de Resolución y Orquestación de Servicios

Componente Factory

Es el componente inicial del subsistema. Es el encargado de realizar las siguientes operaciones:

- Recibe y transforma la entrada de los procesos
- Genera dinámicamente la clase de invocación
- Transforma y devuelve el mensaje de retorno de la clase de invocación

Durante la ejecución del proceso, el componente realiza validaciones de instanciación y si no existen excepciones en la definición solicitada realiza una invocación interna de evento donde crea la clase de invocación y ejecuta el evento solicitado, el proceso es sincrónico ya que espera la respuesta de retorno.

Componente Service

Es el componente principal del subsistema. Es el encargado de realizar las siguientes operaciones:

- Inicializar las configuraciones generales del proceso (Mensajes, Transacción Base)
- Validar de acceso de la invocación del proceso
- Inicializar las variables de control del proceso (Transacion Cliente, Json de Configuración, Json de Datos del Consumo)
- Resolver e invocar el controlador asociado al proceso
- Transformar el objeto de retorno de la clase de invocación en una cadena string en formato Json

Durante la ejecución del proceso, el componente inicializa el API básica de procesamiento como es la transacción base para acceder a las configuraciones iniciales, la clase de mensajes y genera el Json de configuración basado en el parametro (config) luego valida el hash de conexión para determinar vigencia de acceso, permisos, identificación de empresa (para determinar la transacción del proceso) y por último orquesta el evento solicitado y espera la respuesta para transaformarla en un string en formato Json.

4.4.2.1 Diseño detallado del subsistema

Componente Factory

Está integrado por 3 componentes como se muestra en la siguiente imagen:

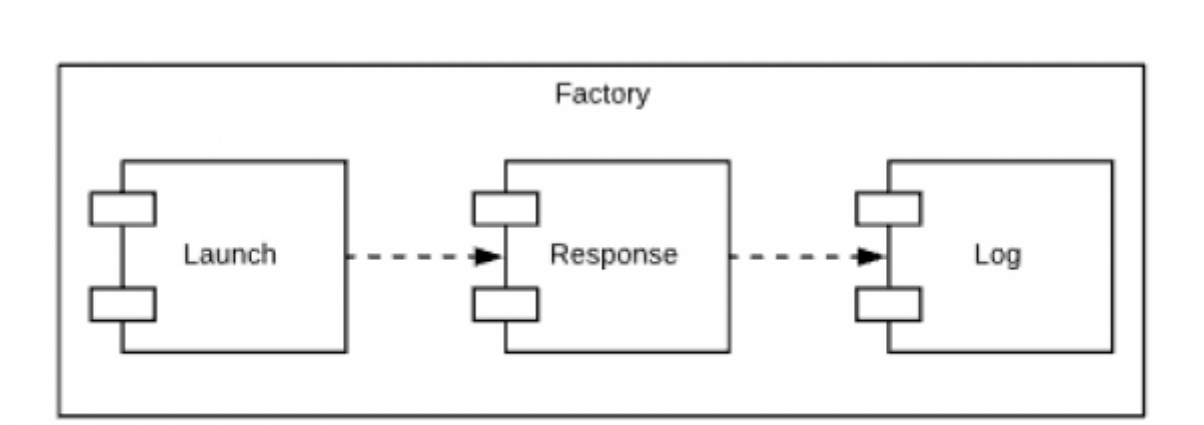

Figura 3: Componente Factory

- **Launch**: recibe la definición de la invocación (clase y evento), la representación de entrada y

genera la definición de la clase invocadora e invoca el evento de ejecución **ue_run**.

- **Response**: recibe la respuesta del proceso resultado de la clase de invocación evento **ue_run** y realiza la transformación de la salida evaluando la estructura de retorno en formato Json.
- **Log**: Almacena la sintaxis de invocación de la operación incluyendo los argumentos de entrada y el resultado de la clase de invocación evento **ue_run**

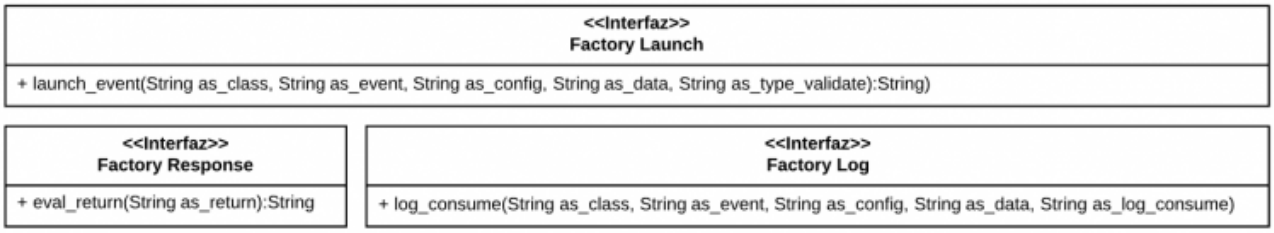

Figura 4: Interfaces - componente Factory

Componente Service

Está integrado por 3 componentes como se muestra en la siguiente imagen:

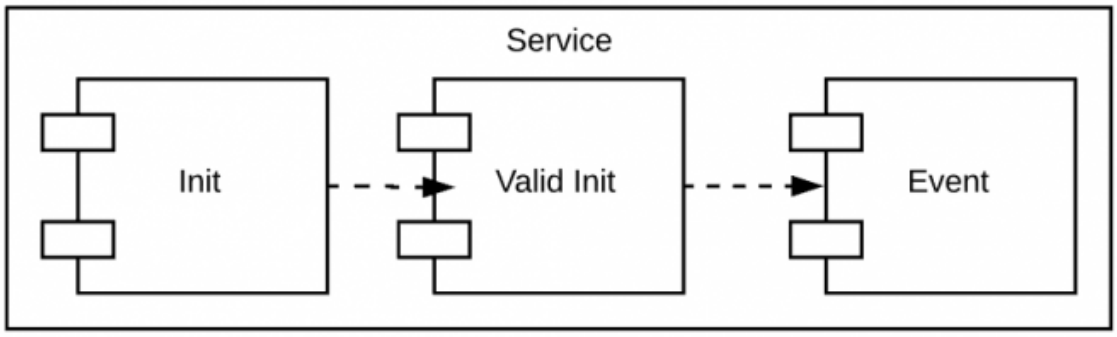

Figura 5: Componete Service

- **Init**: realizxa la inicialización de los contextos generales del proceso (Clases de configuración, Parametros del sistema y Transacción Base).
- **Valid Init**: realiza las validaciones de acceso e inicializa los parametros y variables de proceso para la empresa.
- **Event**: Invoca el evento de inicio del proceso solicitado.

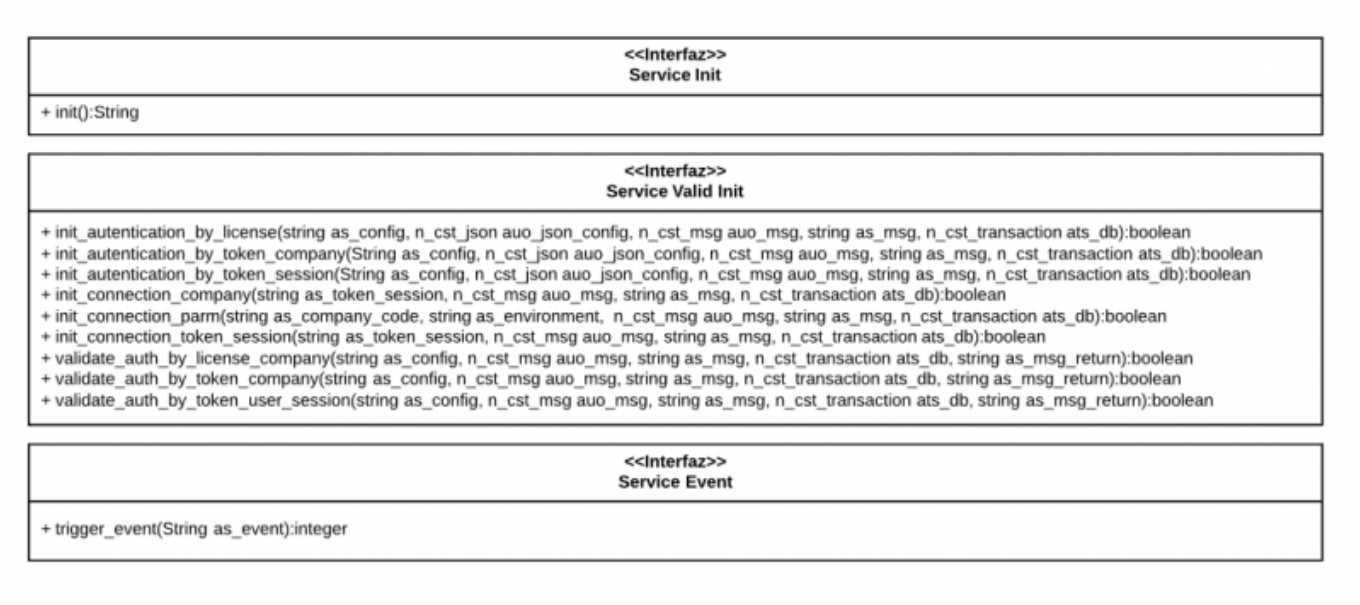

Figura 6: Interfaces - componente Service

4.4.3 Resolución y Orquestación de Transacciones

El subsistema **Resolución y Orquestación de Transacciones** tiene dos componentes como se puede observar en la siguiente imagen:

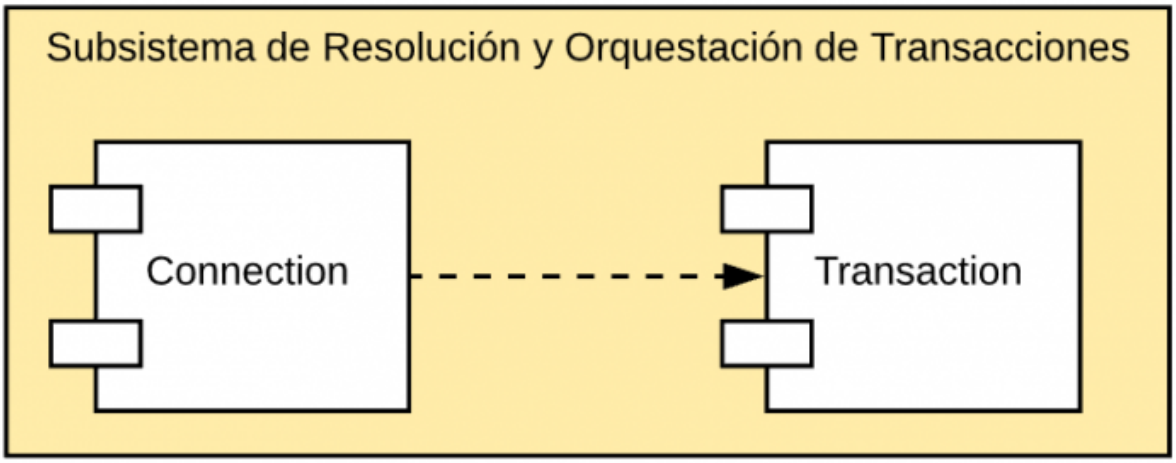

Figura 7: Subsistema de Resolución y Orquestación de Transacciones

Componente Connection

Es el componente inicial del subsistema. Es el encargado de realizar las siguientes operaciones:

- Inicializa la transacción base
- Inicializa la transacción de la empresa
- Conecta o cierra conexiones de datos

Componente Transaction

Es el componente principal del subsistema. Es el encargado de realizar las siguientes operaciones:

- Solicitar y brindar conexiones a los procesos
- Mantener el estado de los parametros de sesión
- Registrar los errores en las transacciones de datos

5 Vista de Deployment

5.1 Introducción

Está sección describe una o más configuraciones físicas sobre las cuales se realiza el deploy de los Servicios SOAP que son generados por el Framework SICOF PBtoWS, así como la infraestructura necesaria para su ejecución.

5.2 Distribución y Deployment

La siguiente imagén presenta el escenario de distribución esperado para el despliegue y ejecución de los componentes generados con el Framework SICOF PBtoWS.

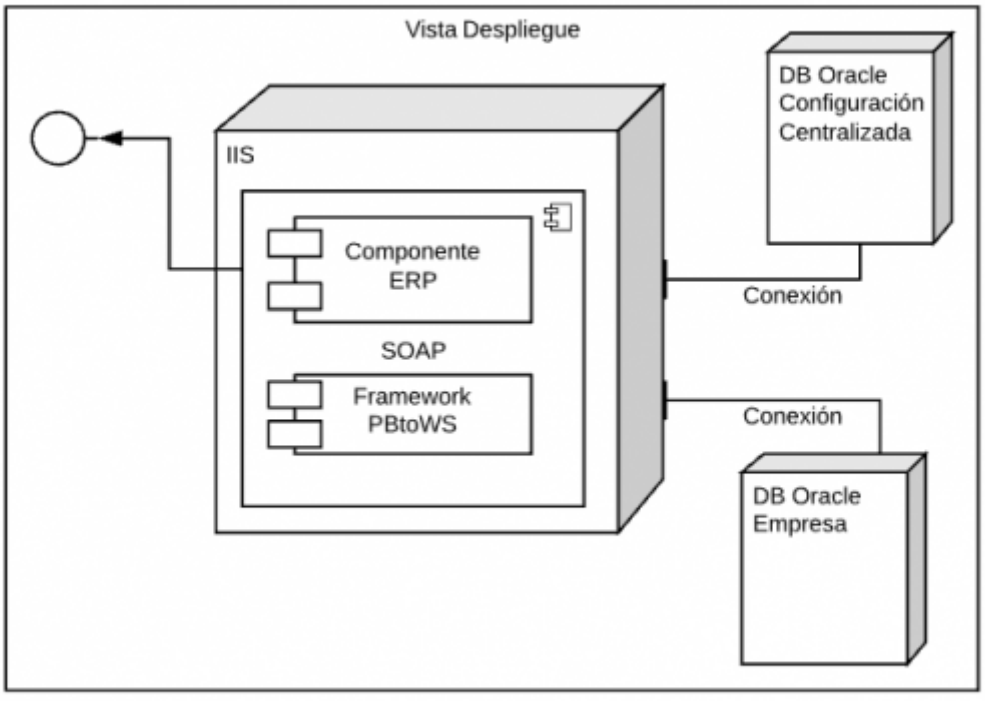

Figura 8: Vista de Despliegue del Framework SICOF PBtoWS

A continuación se describen los nodos presentes en la imagen:

- **DB Oracle Configuración Centralizada**: Servidor donde estará alojada la base de datos de la configuración de accceso y recursos unificados. El motor utilizado para la base de datos es Oracle.
- **DB Oracle Empresa**: Servidores donde estarán alojadas las base de datos con la información de la empresas (clientes). El motor utilizado para la base de datos es Oracle.
- **IIS (Internet Information Services)**: Servidor web donde estarán alojados los servicios SOAP expuesto por el framework **SICOF PBtoWS**. La versión soportada es la 7.5

6 Arquitectura General: Componente SICOF ERP

6.1 Introducción

Teniendo presente la arquitectura presentada, se describe a continuación la arquitectura general de los componentes de SICOF ERP (Powerbuilder). Se toma como prototipo y/ó prueba de concepto una definición general de componente asumiendo que todos compartirán la misma estructura de implementación y arquitectura apoyandose en la filosofía SOA (Arquitectura orientada a servicios).

La descripción se organiza repasando las vistas presentadas anteriormente, detallando en cada una de ellas como se resolvió la realización de los componentes definidos, que componentes de infraestructura, tecnologías, herramientas y lenguajes se utilizaron.

6.2 Vista Lógica

La siguiente imagen presenta una vista global de los subsistemas del framework y las herramientas externas que se utilizaron para lograr cumplir los cometidos definidos para cada subsistema.

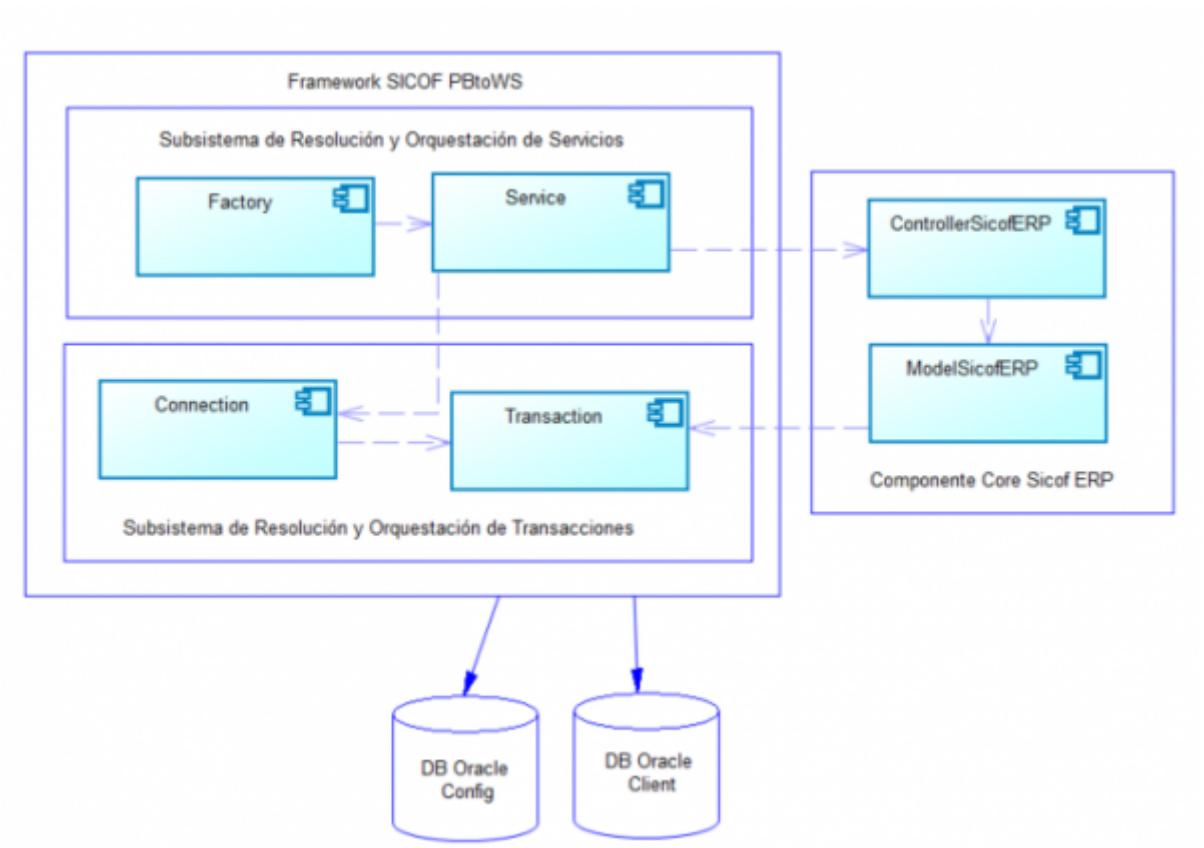

Figura 9: Vista Lógica general de los componentes SICOF ERP con el framework SICOF PBtoWS

6.2.1 Subsistemas de la arquitectura general

- Subsistema de Resolución y Orquestación de Servicios
- Subsistema de Resolución y Orquestación de Transacciones
- Subsistema: Core SICOF ERP Ver sección a continuación

6.2.1.1 Subsistema: Core SICOF ERP

Componente ControllerSicofERP

Contiene la lógica de los procesos del negocio de una funcionalidad de SICOF ERP, está representado en clases (objetos no visuales) que satisfacen funcionalidades para las cuales ha sido construido, este componente es el resultado del proceso de migración del código del Modelo Orientado a Eventos de la Versión 12.5.2.5.0.

Durante la ejecución del proceso, el componente de acuerdo a la orquestación del Subsistema de Resolución y Orquestación de Servicios consume un controlador el cual inicializa y consume métodos de las clases del paquete. Pasa los parametros de sesión y transación y devuelve el objeto de retorno que contiene la información del resultado del proceso ejecutado.

Componente ModelSicofERP

Contiene las definiciones de los modelos de datos (entidades o tablas de la base de datos) que son necesarios para la ejecución de los procesos del componente ControllerSicofERP.

Durante la ejecución del proceso el componente ControllerSicofERP va inicializando los modelos de datos (datawindow) que va necesitando y le asigna la transación identificada que corresponde a un cliente(Empresa). Posteriormente por medio de esos modelos se realiza la persistencia de las modificaciones realizadas a la base de datos.

6.3 Vista Deployment

La Figura 8 visualiza el deployment de los componente SOAP del sistema **SICOF ERP** basado en el escenario de distribución presentado en la Figura 9

6.3.1 Distribución y Deployment

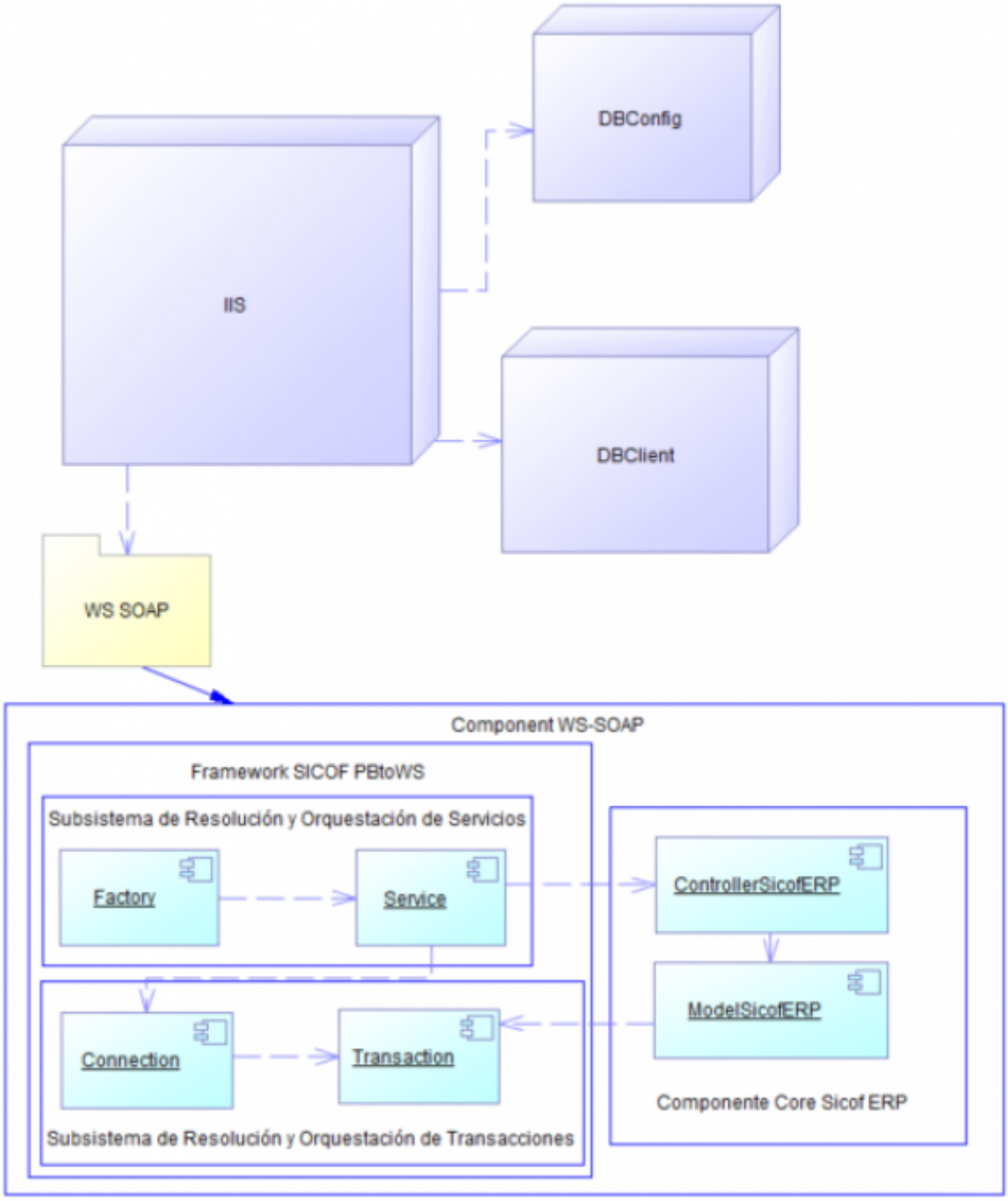

Figura 10: Deployment Componente WS-SOAP SICOF ERP

A continuación se describen los nodos presentes en la imagen:

- **DBConfig**: Servidor donde estará alojada la base de datos de la configuración de accceso y recursos unificados. El motor utilizado para la base de datos es Oracle 11g R2 o superior.
- **DBClient**: Servidores donde estarán alojadas las base de datos con la información de la empresas (clientes). El motor utilizado para la base de datos es Oracle 11g R2 o superior.
- **IIS (Internet Information Services)**: Servidor web donde estarán alojados los servicios SOAP expuesto por el framework **SICOF PBtoWS**. La versión soportada del servidor es la 7.5 y el requiere Framework .NET 4.02 o superior

6.4 Vista de Implementación

En esta sección se presentan los ejecutables y artefactos construidos para la implementación del **Componente WS-SOAP SICOF ERP**

El sistema aquí planteado es una implementación del framework **SICOF PBtoWS** descrito en las secciones anteriores que cumple con la especificación 1.1 y 1.2 de WS-SOAP.

6.4.1 Estructura del Componente WS-SOAP SICOF ERP

La implementación del framework **SICOF PBtoWS** con el componente de **SICOF ERP** se empaqueta en un archivo .MSI o .EXE el cual contiene un directorio con la estructura y contenido de los recursos necesarios para operar el WS-SOAP, el nombramiento de esos componentes queda fuera de la definición de la arquitectura pero respetará la siguiente regla ws(codigo aplicación)_(nombre del proyecto powerbuilder) Ej: ws00_login representar el componente Login de **SICOF ERP**.

La implementación del **Componente WS-SOAP SICOF ERP** se enmarca en la tecnología .Net version 4.02. cada subsistema esta implementado por un conjunto de clases que realizan tareas concretas y se comunican con otras clases de los demas subsistemas.

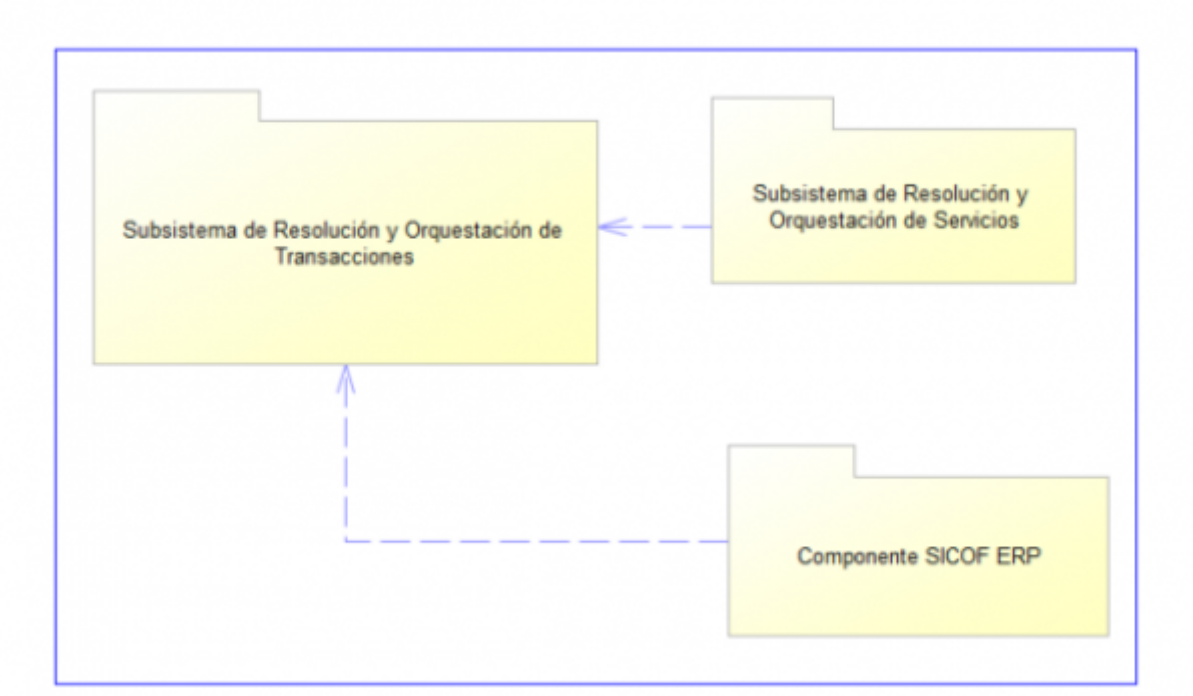

Figura 11: Mapeo de subsistemas

6.4.2 Arquitectura de Implementación

Se presenta a continuación un diagrama que muestra las dependencias entre los Componentes que contienen la definición de las interfaces y la implementación concreta del framework.

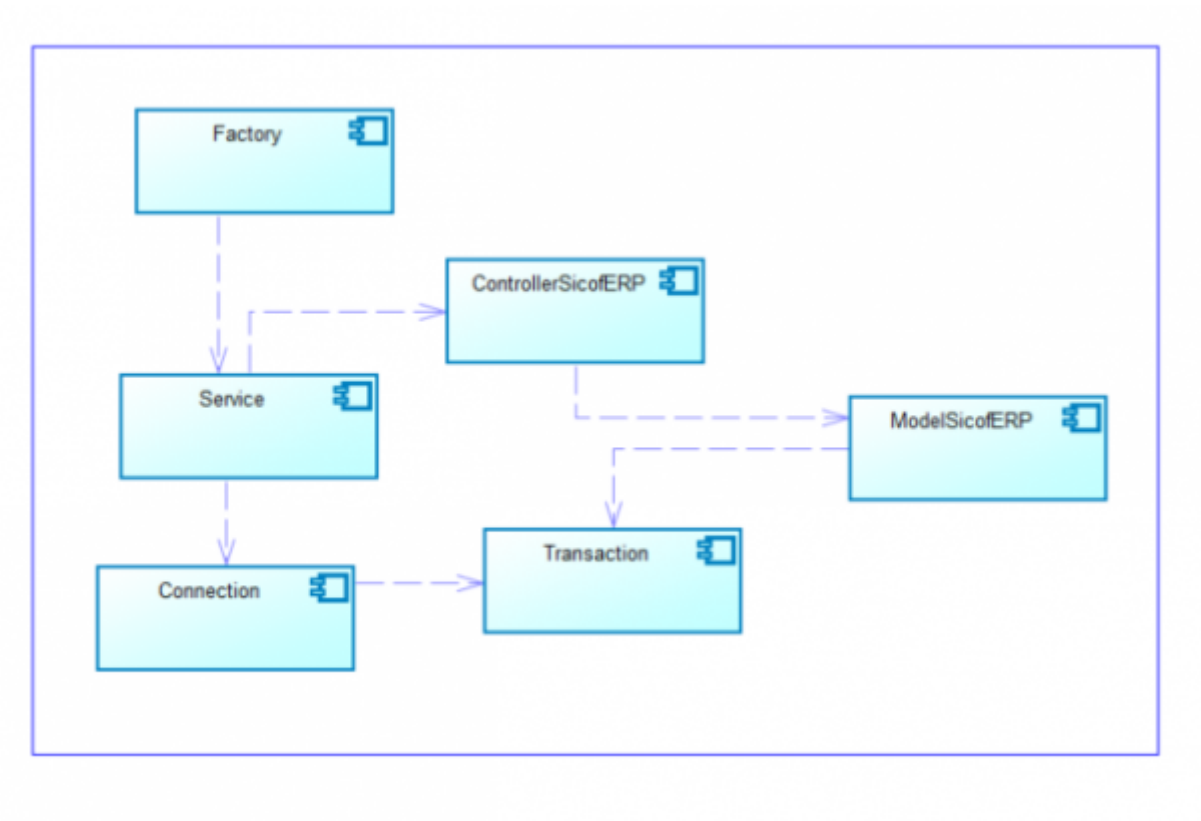

Figura 12: Dependencia de componentes del sistema

Se presenta a continuación la descripción detallada de los artefactos más importantes del subsistemas **Resolución y Orquestación de Servicios** desde el punto de vista de su implementación.

6.4.2.1 Resolución y Orquestación de Servicios

Ademas de los componentes antes abordados en la arquitectura este subsistema implementa funciones de generación XML las cuales se crean en el momento de despliegue para soportar la interfaz WSDL generada por la exposición del servicio SOAP.

Como se visualiza en la siguiente imagen la implementación del subsistema presenta una dependnecia al componente **Powerbuilder Generate WSDL** el cual es el encargado de crear las interfaces de las operaciones expuestas en el servicio SOAP. Este componente hace parte de capa interna de exposición de servicios utilizada por la tecnologia Powerbuilder y se basa en el framework .Net 4.02

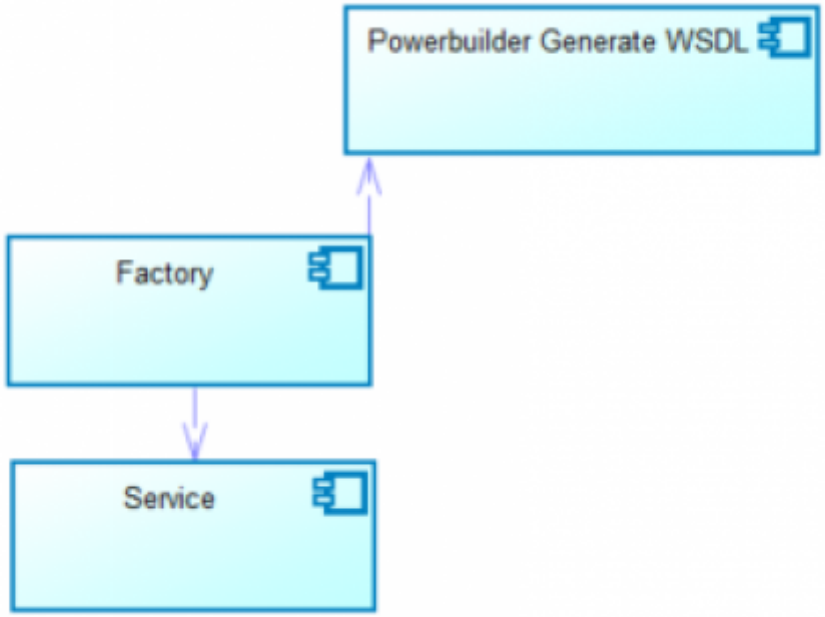

Figura 13: Dependencia del subsistema Resolución y Orquestación de Servicios

Se presenta a continuación una plantilla de un wsdl que describe la interfaz de servicio que expone el subsistema de Resolución y orquestación de Servicios.

Ejemplo: Login anónimo (Consumo)

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:tem="http://tempurl.org">
   <soapenv:Header/>
   <soapenv:Body>
      <tem:ws_anonymous>
         <!--Optional:-->
         <tem:as_config>{"company_code": "UNILLANOS", "user":"SICOF",
"password":"sicofERP", "ip":"127.0.0.1",
"environment":"TEST"}</tem:as_config>
      </tem:ws_anonymous>
   </soapenv:Body>
</soapenv:Envelope>
```

Ejemplo: Login anónimo (Respuesta)

```
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
   <soap:Body>
      <ws_anonymousResponse xmlns="http://tempurl.org">
         <ws_anonymousResult>{
    "return_code": "1",
```

```
    "return_user_message": "sesión autenticada correctamente",
    "return_technical_message": "sesión autenticada correctamente",
    "return_object": {
        "token_session":
"557055f98cd0285e319fa27192c3ab2d9f2f77583bbb72b738a693c0cf364dca"
    },
    "return_response_date": "09/07/2019 14:14:30"
}</ws_anonymousResult>
      </ws_anonymousResponse>
   </soap:Body>
</soap:Envelope>
```

1)

Una API es una interfaz de programación de aplicaciones (del inglés API: Application Programming Interface). Es un conjunto de rutinas que provee acceso a funciones de un determinado software. Web API - Wikipedia, la enciclopedia libre

2)

Powerbuilder Library

3)

http://infocenter.sybase.com/help/index.jsp?topic=/com.sybase.infocenter.dc37775.1250/html/dwprgug/CAIDADAJ.htm|Traducción de SyBooks Online

4)

Certificación obtenida en Julio de 2017 por la empresa ADA

5)

Módulo inicial Gestión Humana: Nómina

6)

Solo se toma el modelo de repsentación por vistas ya que el framework propobne una dinamica de documentación diferente

7)

Modelo Vista Controlador (MVC) es un estilo de arquitectura de software que separa los datos de una aplicación, la interfaz de usuario, y la lógica de control en tres componentes distintos.Modelo vista controlador (MVC)

8)

En informática, CRUD es el acrónimo de “Crear, Leer, Actualizar y Borrar” (del original en inglés: Create, Read, Update and Delete), que se usa para referirse a las funciones básicas en bases de datos o la capa de persistencia en un software.CRUD - Wikipedia, la enciclopedia libre

9)

PowerBuilder es una herramienta de desarrollo de clase empresarial desarrollada por la empresa PowerSoft. PowerBuilder es orientada a objetos y permite el desarrollo de diferentes tipos de aplicaciones y componentes para ejecutar arquitecturas cliente/servidor, distribuidas y Web.PowerBuilder - Wikipedia, la enciclopedia libre

10)

Framework SICOF PBtoWS

11)

Una red peer-to-peer, red de pares, red entre iguales o red entre pares es una red de ordenadores en la que todos o algunos aspectos funcionan sin clientes ni servidores fijos, sino una serie de nodos que se comportan como iguales entre sí.Peer-to-peer - Wikipedia, la enciclopedia libre

12)

Datasource de la base de datos del cliente