Lámpsakos | N°. 14 | pp. 40-50 | julio-diciembre | 2015 | ISSN: 2145-4086 | Medellín–Colombia

Capacidad de Orquestación de Servicios Web en las Herramientas MULE ESB y Oracle Service Bus

Web Service Orchestration Capacity in MULE ESB and Oracle Service Bus tools

Claudia Ivette Castro-Zamora, Ing.
Instituto Superior Politécnico José Antonio Echeverría.
La Habana, Cuba.
ccastro@ceis.cujae.edu.cu

Eva Flores-Valdés, Ing.
Instituto Superior Politécnico José Antonio Echeverría.
La Habana, Cuba
eflores@ceis.cujae.edu.cu

(Recibido el 10-04-2015. Aprobado el 20-06-2015)

Citación de artículo, estilo IEEE:
C. I. Castro-Zamora, E. Flores-Valdés, "Capacidad de Orquestación de Servicios Web en las Herramientas ESB MULE y Oracle Service Bus", Lámpsakos, N° 14, pp. 40-50, 2015
DOI: http://dx.doi.org/10.21501/21454086.1640

Resumen

La gestión de la información se ha convertido en un factor fundamental para el desempeño óptimo del sector empresarial. Es muy frecuente que los sistemas implantados deban evolucionar en conjunto con la organización en la que se encuentran funcionando, con el objetivo de satisfacer los nuevos requerimientos que surgen en la misma. En este proceso de crecimiento se ve la necesidad de que estos sistemas compartan información y se trabaje con esta de manera automatizada para dar soporte a los disímiles procesos de negocio definidos. La adopción de una Arquitectura Orientada a Servicios (SOA) posibilita la creación de sistemas dinámicos y escalables, además, facilita el proceso de interacción entre estos y otras aplicaciones. Entre los facilitadores para la implantación de una SOA se encuentra la tecnología Bus de Servicios Empresariales que se basa en una infraestructura de software que funciona como capa intermedia, proporcionando servicios de integración de las distintas aplicaciones a través de mensajería basada en estándares y servicios de sincronización. Esta se basa en un conjunto de capacidades para dar solución a cualquier escenario de integración. En este trabajo se evaluará el comportamiento de dos herramientas que implementan la tecnología ESB con respecto a la capacidad de orquestación de servicios que no es más que un proceso de colaboración entre distintos servicios bajo un patrón determinado.

Palabras clave: Orquestación de Servicios; Arquitectura Orientada a Servicios; Bus de Servicios Empresariales.

Abstract. Information Management has become a key factor for enterprise optimal development. It is usual that legacy systems implanted evolve along with the organization in which they function, in order to fulfill new upcoming requirements. Growth processes are essential in those systems that share information while supporting defined business processes. The adoption of a Service Oriented Architecture (SOA) enhances the creation of both dynamic and scalable systems. Furthermore, it facilitates the interaction among those systems and other applications. The technology Enterprise Service Bus (ESB) arises as a facilitator of an SOA implantation. An ESB is bases upon a software infrastructure that acts as a middleware that provides integration services between different applications by means of messaging based standards. It also provides a set of capacities to support the making of feasible solutions to any integration scenario. In this work, authors analyze some of the macro components provided by referred ESB´s tools that support service orchestration capacity as a service collaboration process under a defined pattern.

Keywords: Service Orchestration; Service Oriented Architecture; Enterprise Service Bus.

1. INTRODUCCIÓN

En la actualidad, la problemática de integración de aplicaciones es un tema común debido a que las empresas requieren desarrollar, implementar e integrar nuevas aplicaciones en sus entornos. Esta necesidad creciente de aplicaciones dinámicas constituye uno de los motivos principales por el que muchas organizaciones han adoptado o están por adoptar una Arquitectura Orientada a Servicios (SOA, por sus siglas en inglés) como la nueva base de sus aplicaciones. La idea de SOA es que, mediante la utilización de servicios, se pueda dar soporte a los requisitos del negocio, proporcionando la creación de aplicaciones altamente escalables y que puedan interactuar entre diferentes sistemas propios de la organización o de terceros [1].

Entre los facilitadores tecnológicos que permiten la implantación de una SOA, se encuentra el modelo Bus de Servicio Empresarial (ESB, por sus siglas en inglés). Lo esencial de este modelo es que se encuentra diseñado para actuar como una capa intermedia de comunicación entre los distintos sistemas sin que importe la tecnología en que estén desarrollados y su ubicación en la red [2].

Según una encuesta aplicada por la compañía *Forrester Research* en el año 2011 de un total de 167 encuestados entre desarrolladores de aplicaciones y arquitectos del área de integración pertenecientes a empresas ubicadas en Europa, Asia y Norteamérica, el 58% se encontraban utilizando un ESB, un 32% de los encuestados estaban considerando el uso de este modelo y solo un 7% indicaron que no estaban interesados [3]. Este resultado revela un alto nivel de interés por los ESB, ya que las amplias capacidades que proporcionan han hecho una opción popular para resolver las necesidades de integración e implementar una SOA.

Entre las ventajas de un ESB está que proporciona ubicación centralizada para la administración y gestión de los sistemas integrados, brindando una serie de funcionalidades predefinidas que ahorran en tiempo y esfuerzo a los desarrolladores a través de una mayor configuración en vez de tener que codificar toda la integración [4]. Además provee la capacidad de combinar varios servicios existentes para que cumplan con las necesidades requeridas por el cliente, puesto que con un solo servicio no se satisfacen; esta capacidad es conocida como orquestación de servicios [4]. La gestión y monitoreo de los servicios para tener conocimiento y control de la calidad de los servicios (QoS, por sus siglas en inglés), así como la transformación y enrutamiento de los mensajes que viajan de una aplicación a otra, son otras de las capacidades con las que cuenta un ESB [5]. Sin embargo la implantación de un ESB en una organización exige de una administración constante y se precisa de personal capacitado. También presenta mayor latencia, causada por los mensajes que viajan dentro del ESB, especialmente si se compara con las comunicaciones punto a punto.

En el Instituto Superior Politécnico José Antonio Echevarría (CUJAE) existen disímiles sistemas que están especializados en la gestión de procesos en los que se involucran trabajadores docentes, no docentes y estudiantes. Entre estos se tienen, el sistema SIGENU, encargado de la gestión de los procesos estudiantiles, el Sistema del pago del estipendio estudiantil en el departamento de Economía de la CUJAE, el Sistema Doctus para la gestión de préstamos de libros en la biblioteca o el sistema Copérnico, encargado de la gestión de los proyectos de investigación en el que participan profesores y especialistas de la universidad. Además, existen, para cada Facultad, sistemas de Intranet en la que se gestionan algunos procesos, tenidos en cuenta ya en aplicaciones mencionadas. Esta situación da paso al problema de la existencia de un entorno automatizado, donde los sistemas actúan como islas y la información que manejan se encuentra dispersa y replicada en muchos casos. Debido a esto se han adoptado medidas, en principio, para dotar a las aplicaciones existentes de mecanismos de comunicación, tales como los servicios web y se han documentado estos en sitios web con el objetivo de brindar algunas opciones de registro; esta última solución, los autores la clasifican como deficiente debido a la no utilización de registros especializados, como los basados en el estándar UDDI.

En estos momentos se están investigando iniciativas para re-implementar los procesos actuales utilizando estrategias de integración para la construcción de una visión unificada de la información y los procesos de la universidad. Una de estas estrategias es el estudio del modelo Bus de Servicios Empresariales, sus capacidades y factibilidad de uso para solucionar ciertos entornos de integración. En este artículo se analiza la capacidad orquestación de servicios en dos herramientas ESB para lo que se realiza un estudio de componentes de las herramientas MuleSoft y Oracle Service Bus utilizados en esta capacidad,

además se valora su utilización en la implementación de un caso de estudio de integración donde se involucran los sistemas SIGENU y la Intranet de la Facultad Ingeniería Informática.

2. SERVICIOS WEB

En las últimas décadas los sistemas informáticos han evolucionado desde ser totalmente aislados y centralizados hacia su concepción dinámica y distribuida [6] aprovechando las posibilidades que han traído los avances en las comunicaciones sobre la red. Este hecho se evidencia a través del surgimiento de la computación orientada a servicios (SOC, por sus siglas en inglés) la que provee un enfoque de gobierno para la visión de la automatización de procesos o lógica de negocio como sistemas distribuidos [6]. SOC tuvo sus antecedentes en los modelos orientados a objetos y posteriormente en el modelo de desarrollo basado en componentes, el cual incorporó beneficios como la auto-descripción, encapsulación, descubrimiento y la carga dinámica que la orientación a servicios sigue adoptando.

La Arquitectura Orientada a Servicios es un paradigma para la concepción de SOC y proporciona una guía para el diseño de servicios accesibles y reusables con el objetivo de obtener como resultado una arquitectura flexible. Los servicios constituyen el componente principal de una SOA y de forma general exponen una función de negocio que debe ser consumida por una aplicación cliente [1]. La comunicación entre un proveedor de servicios y su consumidor se puede implementar por diferentes vías [7]:

Comunicación dentro de la misma plataforma utilizando mecanismos como *Remote Method Invocation* (RMI, por sus siglas en inglés) o *Java Message Service* (JMS, por sus siglas en inglés) todos dentro de la plataforma Java. Este tipo de comunicación tiene la desventaja de que el servicio y sus consumidores deben compartir la misma tecnología lo que no favorece la interoperabilidad.

Capa de comunicación de objetos distribuidos como el uso de una Arquitectura Común de Intermediarios en Peticiones a Objetos (CORBA, por sus siglas en inglés) o a través la utilización del Modelo de Objetos de Componentes Distribuidos (DCOM, por sus siglas en inglés). Este mecanismo aunque aportaba interoperabilidad a las soluciones, decreció en popularidad debido al auge de tecnologías sobre la Web y consecuentemente, de soluciones mejor alienadas a esta [1].

Protocolo de comunicación basado en texto, donde el cliente y su consumidor se comunican a través de mensajes de texto. Los servicios web están basados en este enfoque y se logra la interoperabilidad ya que los mensajes que se intercambian utilizan el estándar XML para su representación.

Según la W3C los servicios web son sistemas de software diseñados para el soporte de la interacción máquina a máquina sobre la red que contiene una interfaz descrita en un formato procesable [8]. Desde el punto de vista tecnológico los servicios web se pueden implementar por una parte, utilizando el estándar Simple Object Access Protocol (SOAP, por sus siglas en inglés) para identificar el formato de mensajes que se va a intercambiar; Web Service Description Language (WSDL, por sus siglas en inglés) para la descripción de los datos, operaciones e información de comunicación. Por otra parte, los servicios web REST se caracterizan por la definición de un conjunto de principios de arquitectura que basa su funcionamiento en las capacidades que brinda el protocolo HTTP donde todos los recursos del servicio se identifican a través de una URI, se manipulan utilizando un conjunto fijo de operaciones PUT, GET, POST y DELETE además, los recursos no están acoplados con su representación por lo que su contenido puede ser accedido en otros formatos como HTML, XML, texto plano, entre otros [9].

La selección de algunos de estos estilos de arquitectura para la implementación de servicios web depende totalmente de entorno en el que vayan a funcionar. Los servicios basados en REST son usualmente más sencillos de desarrollar y más ligeros, por lo que se pueden adecuar mejor a escenarios de aplicaciones móviles por ejemplo; mientras que, si se desea exponer lógica y no datos o se requieren las capacidades de los protocolos estándares de WS-*, entonces la opción del estilo SOAP es la más adecuada [6].

3. ORQUESTACIÓN DE SERVICIOS

Como se ha mencionado anteriormente, la esencia de SOA es la creación de servicios que deben tener, entre otras, la característica de ser reusables para sistemas que los utilicen directamente o para

implementar nuevas funcionalidades, producto de la composición de un conjunto de estos. Según [10] la composición de servicios es una agregación coordinada de servicios cuyo alcance está asociado generalmente a la automatización de procesos de negocio. Los servicios web se pueden combinar de dos formas, mediante una orquestación o una coreografía [11].

La coreografía no depende de un orquestador central, cada servicio que interviene en el proceso tiene que conocer cuándo entrar en acción y cuándo estar inactivo [12]. Desde el punto de vista de la ejecución de un proceso la orquestación se centra en el funcionamiento interno, mientras que la coreografía lo hace en la perspectiva externa, enfocándose en la interacción de este proceso [11]. Por otra parte, la orquestación se refiere a un proceso de negocio compuesto por servicios web para completar su objetivo. El proceso es controlado por un agente en el sistema y es descrito en términos de intercambio de mensajes y órdenes de ejecución [11, 12]. Los servicios web invocados no son conscientes de que están siendo invocados para la definición de un proceso de negocio [11]. Esto hace que la orquestación sea un proceso más eficiente para la composición de servicios, aspecto que es demostrado por el apoyo que ha recibido por la industria [11].

En la última década se han desarrollado varios estándares para la descripción formal de ambos procesos, específicamente para la coreografía se concibieron, entre otros, Web Service Choreography Description Language (WS-CDL) [13] y Web Service Choreography Interface (WSCI) [14]. Por otra parte, BPEL es una especificación de lenguaje basado en XML para la concepción de procesos de negocio a través de la interacción de servicios web y es utilizado para modelar el comportamiento tanto de procesos de negocio ejecutables como abstractos [15].

La definición de un proceso BPEL puede ser ejecutada por un motor de orquestación, que es el coordinador central. El motor lee el documento BPEL e invoca los servicios web necesarios en el orden especificado, ya sea secuencialmente o en paralelo. El proceso en sí puede ser ofrecido como un nuevo servicio web, ser invocado de la misma forma y puede expresar comportamiento condicional. Además, se pueden definir lazos, tratamiento de errores, copiar y asignar valores, entro otros, por lo que la definición de un proceso BPEL puede realizarse de forma algorítmica.

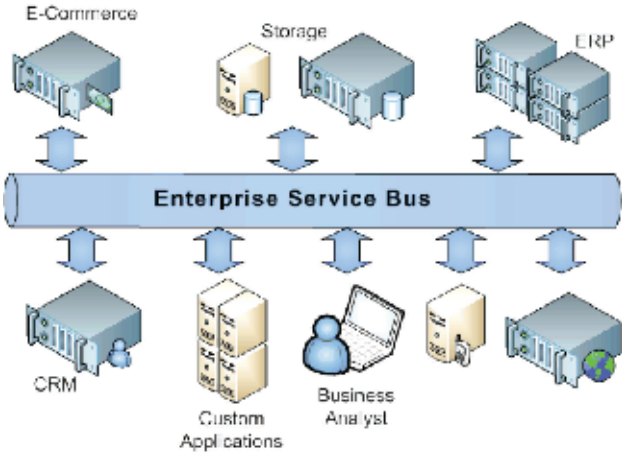

Fig. 1. Modelo ESB [17]

4. ESB Y SUS CAPACIDADES

En una Arquitectura Orientada a Servicios, como se muestra en la Fig. 1 un ESB representa el elemento de software que media entre las aplicaciones y permite la comunicación entre ellas, sin importar la heterogeneidad de protocolos y tecnologías. Idealmente ESB es una oferta de producto que proporciona al desarrollador un conjunto de funcionalidades o capacidades de integración y un entorno de gestión válidos para satisfacer los requisitos de cualquier escenario de integración [16].

Transformación de mensajes: cuando se habla de integración de aplicaciones, no es un error pensar que estas soporten diferentes formatos de información, por lo que, si se quiere lograr establecer una comunicación entre estas, a través de un ESB, es necesario que este tenga soporte, para trabajar en los formatos que utilicen los sistemas y tenga funcionalidades para transformarlos de unos a otros [18].

Balanceo de carga: con la implementación de esta capacidad, el ESB puede registrar la dirección de varios servidores en los que está desplegado un servicio web, con el objetivo de prevenir que fallos en uno de estos, afecte su disponibilidad. Esta capacidad se puede implementar, además, basada en el contenido de un mensaje cuando, en dependencia de parámetros definidos para este, se envía a colas de mensaje u otros

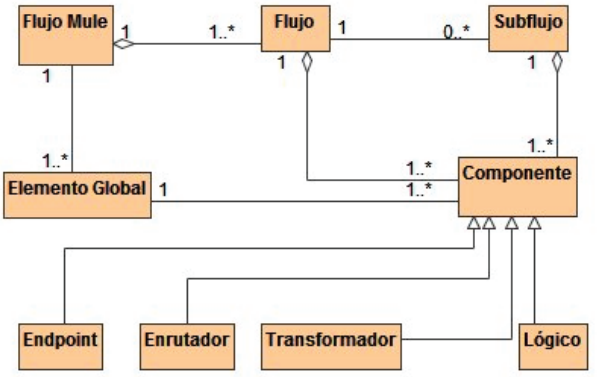

Fig. 2. Conceptos que se relacionan en un flujo de Mule ESB. Fuente: Elaboración propia

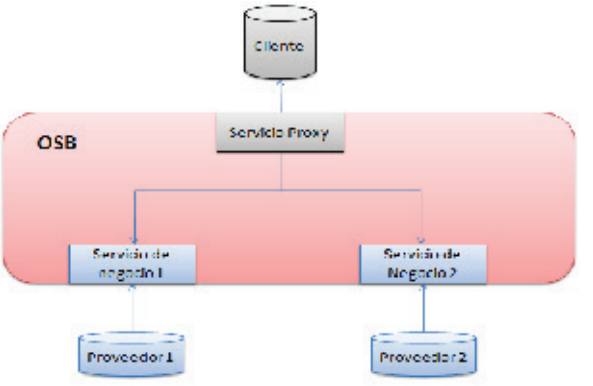

Fig. 3. Servicios de negocio y servicios proxy en Oracle Service Bus [25].

4.1. Herramienta MULE ESB.

Mule ESB es un proyecto de software libre desarrollado por Ross Mason perteneciente a la compañía MuleSoft, actualmente se encuentra en su versión 3.4. Mule ESB es un *framework* de mensajería ligera basada en Java que permite la rápida conexión e intercambio de datos entre aplicaciones [19]. Mule permite a los desarrolladores integrar sus aplicaciones e intercambiar datos sin tener en cuenta las tecnologías y maneja todas las interacciones entre aplicaciones y componentes de manera transparente sin importar los protocolos de transporte utilizados (JMS, Web Services, JDBC, HTTP, FTP, SMTP). Permite la publicación y la invocación de servicios web de tipos REST y SOAP, haciendo uso del *framework* CXF [20]. Adicionalmente posibilita hacer transformaciones de mensajes incluyendo XSLT. Se integra con proyectos como Maven y Spring que garantizan una correcta estructuración de los proyectos y configuración de objetos en los flujos ESB respectivamente.

Mule provee herramientas de apoyo a los desarrolladores para la creación y configuración del fichero XML que contiene la definición de los flujos Mule. Este es el caso de la herramienta *Eclipse*, que a través de *plugins* se pueden desarrollar proyectos de tipo ESB y de la herramienta *MuleStudio* que está basada también en Eclipse y brinda un conjunto de componentes gráficos que hace más fácil el diseño de los flujos.

Tal y como se muestra en la Fig. 2 para el desarrollo de flujos Mule, se hace necesario definir un flujo principal, en el cual se definen flujos y estos a su vez pueden tener asociados un conjunto de subflujo, esto con el objetivo de lograr una mejor organización del flujo principal. Además cada flujo y/o sub flujo está compuesto por componentes enrutadores, transformadores, de tipo *endpoint*, y lógicos. Además se pueden declarar elementos globales que son también componentes, pero se pueden utilizar a nivel del flujo Mule y no en cada flujo o subflujo que se defina.

Mule ESB en su versión Enterprise cuenta con una consola de administración (MMC) que actúa como servidor ESB, que brinda la posibilidad de monitorear y gestionar los distintos flujos de las aplicaciones que son desplegados en el ESB. Este producto permite visualizar los recursos de las aplicaciones implementadas, dígase flujos, componentes y mensajes que se intercambian entre componentes, además del estado del servidor en cuanto a consumo de recursos físicos de donde ha sido desplegado.

4.2. Herramienta OSB

OSB es un programa desarrollado por la compañía BEA *System*, y adquirido, posteriormente, por la compañía Oracle. Este permite conectar cualquier servicio utilizando estándares de servicios web, protocolos de mensajes tradicionales y transportes personalizados adaptables a un entorno empresarial determinado [21].

```
Parámetros de entrada: año,grupo
lista de ids = invocar getListStudentIdentificationByLoadFilter con año y grupo
para cada id en lista de ids
  id = lista de ids en índice actual
  datosPersonales = invocar getStudentFilePersonalData con id
  promedio = invocar getStudentEvaluationAverage con id
  datosEstudiante = datosPersonales y promedio
  listaDatos en índice actual = datosEstudiante
devolver listaDatos
```

Fig. 4. Seudocódigo con la lógica de orquestación de los servicios web analizados.

Fuente: Elaboración propia

OSB tiene la capacidad de soportar diversos protocolos para servicios web como son HTTP/SOAP, WS-I, WS-Security, WS-Policy, WS-Addressing, EJB/RMI. En WebSphere, SOAP v1.1 y v1.2. Este también acepta mensajes en SOAP v1.1 y los convierte en SOAP v1.2 cuando el servicio destino lo requiere. Para la integración de aplicaciones incorpora conectores para MQ Series, CICS, .NET, C y C++ de forma nativa [21]. Provee un entorno para el modelado del flujo de mensajes donde permite configurar los servicios, modelarlos, sincronizarlos a través del registro de servicios y validar los mensajes. También soporta transformación de mensajes y llamadas a servicios de proveedores externos.

OSB se integra mediante un *plug-in* a la herramienta eclipse, el cual habilita un entorno gráfico para el desarrollo del flujo de mensaje. Una de sus capacidades consiste en que este entorno gráfico es donde se configura la lógica del enrutamiento de los mensajes basándose en su contenido y se validan los mismos [22, 23].

Una de las ventajas de OSB es que implementa por defecto un servicio para la gestión, el cual permite monitorear el envío de todos los mensajes. La consola de administración de la herramienta provee una vista unificada donde se muestra el estado de todas las aplicaciones que se encuentran desplegadas y el monitoreo de los servicios [24, 25].

Para la implementación de sus capacidades, la herramienta se basa en dos conceptos principales, los servicios de negocio y los servicios proxy. Los primeros son la representación de los servicios de las aplicaciones desplegadas con las cuales se desea intercambiar información. Este recurso permite realizar configuraciones como el transporte a utilizar con la aplicación destino, políticas de seguridad, entre otras [24]. Por otra parte, el servicio proxy es uno de los principales elementos dentro de la arquitectura de OSB. Estos se convierten en la interfaz que utilizan los consumidores para conectarse con los proveedores de servicios. Una de las ventajas que posee, es que permite definir flujos de mensajes, en el cual se puede implementar lógica para encaminar los mensajes a múltiples servicios de negocios [24].

Oracle Service Bus soporta estándares abiertos con el objetivo de asegurar la integridad y privacidad de las comunicaciones y que solo los usuarios autorizados puedan acceder a los recursos en un dominio OSB[21]. La herramienta utiliza el *framework* de seguridad de su servidor de aplicaciones Weblogic server como base para sus servicios de seguridad. El modelo de seguridad de Oracle Service Bus incluye la Seguridad inbound (de entrada), outbound (de salida) y administrativa. Mediante la configuración de la seguridad inbound se asegura que los servicios proxy gestionen solo las peticiones que provienen de usuarios autorizados. Esta se configura cuando los servicios que se exponen en OSB para los clientes; deben ser asegurados debido a las características de la información que se gestiona. La seguridad outbound, por otra parte, se aplica cuando los servicios que proveen las aplicaciones proveedoras tienen implementados requisitos de seguridad. Por último la configuración de seguridad administrativa permite restringir el acceso a funciones dentro de Oracle Service Bus mediante el establecimiento de roles con privilegios de seguridad predefinidos [21].

Por su rol como intermediario, la herramienta OSB define dos conceptos fundamentales que intervienen verticalmente en todas las capas de su arquitectura funcional.

Servicio de Negocio: son la representación de los servicios de las aplicaciones desplegadas con las cuales se desea intercambiar información, en OSB. Este recurso permite realizar configuraciones como el transporte a utilizar con la aplicación destino, políticas de seguridad, entre otras [21, 26].

Servicios Proxy: constituye uno de los principales elementos dentro de la arquitectura de OSB. Estos se convierten en la interfaz que utilizan los consumidores para conectarse con los proveedores de servicios. Una de las ventajas que posee, es que per-

mite definir flujos de mensajes, en el cual se puede implementar lógica para encaminar los mensajes a múltiples servicios de negocios [21, 26].

De manera general los dos tipos de servicio colaboran entre sí en la implementación de un escenario de integración determinado; los primeros representando a los servicios de las aplicaciones que tienen el rol de proveedor y los segundos, construyendo, a través de la configuración de un flujo de mensaje donde se invocan a los servicios de negocio, las funcionalidades que requieren las aplicaciones con el rol de clientes. La Fig. 2 ilustra esta relación.

4.3. Implementación de un escenario de integración con las herramientas Mule ESB y OSB

El sitio web de la Intranet de la Facultad de Ingeniería Informática de la CUJAE está creado con el objetivo de gestionar la información asociada a los procesos de gestión de la Facultad. La versión actual tiene implementados un mecanismo de actualización que se basa en una filosofía de colaboración y participación al otorgar roles y permisos asociados a usuarios desde estudiantes a trabajadores docentes y no docentes con el objetivo de modificar la información y garantizar así la constante renovación y actualidad de la misma [27]. Este mecanismo tiene que ir aparejado con una campaña de motivación de estas personas a cooperar, lo que no se ha realizado con el nivel de profundidad requerido. Por estas razones hemos identificado al sitio web como fuente de buenos escenarios de utilización de servicios web tanto simples como compuestos. Uno de los requisitos funcionales identificados relativo a la información de cada estudiante es que, dado un grupo de la Facultad, se obtengan los datos personales con el promedio de sus estudiantes.

Por otra parte el Sistema de Gestión Universitaria (SIGENU) es el sistema informático para la gestión del proceso docente que se utiliza en la Cujae la cual expone sus funcionalidades a través de servicios web. Un estudio exploratorio de estos servicios web arrojó que no existe ninguno que directamente satisfaga el requisito funcional del portal de la Intranet de la Facultad de Ingeniería Informática. Aun así se identificaron algunos que combinados (orquestados) pueden obtener el resultado deseado. La tabla siguiente muestra la descripción de estos servicios.

Tabla 1. Descripción de los servicios web de sigenu identificados (fuente: elaboración propia)

Nombre del servicio	Nombre de la operación	Descripción
ListStudentId ByFilter	getListStudent Identification ByLoadFilter	Devuelve un listado con los CI de los estudiantes dado un filtro
FileStudent Service	getStudentFile PersonalData	Devuelve información personal del expediente de un estudiante dado su carnet de identidad (CI).
Evaluation Student Service	getStudent Evaluation Average	Devuelve el promedio general de un estudiante dado su CI.

Los servicios web anteriores están implementados utilizando la tecnología JAX-WS; requieren autenticación básica y dos de ellos, además, utilizan certificados de seguridad.

La Fig. 4 muestra un seudocódigo que define la lógica de orquestación para los servicios web descritos anteriormente y que soluciona el requisito funcional analizado en este escenario de integración.

4.4. Orquestación con la herramienta Mule ESB

Para la orquestación de los servicios en la herramienta Mule ESB se diseñó un flujo principal encargado de hacer transformaciones de datos y coordinar las llamadas a tres subflujos responsables de invocar los servicios web a orquestar.

Como punto de entrada al flujo principal se utilizó el componente HTTP al igual que para acceder a los servicios del proveedor SIGENU. Esto implica la definición de HTTP como protocolo de comunicación a utilizar.

Durante el proceso de diseño de orquestación se hizo necesario utilizar algunas transformaciones al mensaje en el flujo, para lo que se utilizaron los componentes:

Body to Parameter Map con el objetivo de transformar la solicitud HTTP a un mapa de parámetros y así obtener los valores de entrada a la orquestación; el componente *Groovy* para elaborar el objeto complejo que requiere uno de los servicios a consumir en el componente Java para confeccionar el objeto de respuesta de la orquestación y el componente *Object to XML* para dar como respuesta de la orquestación un XML con los datos requeridos.

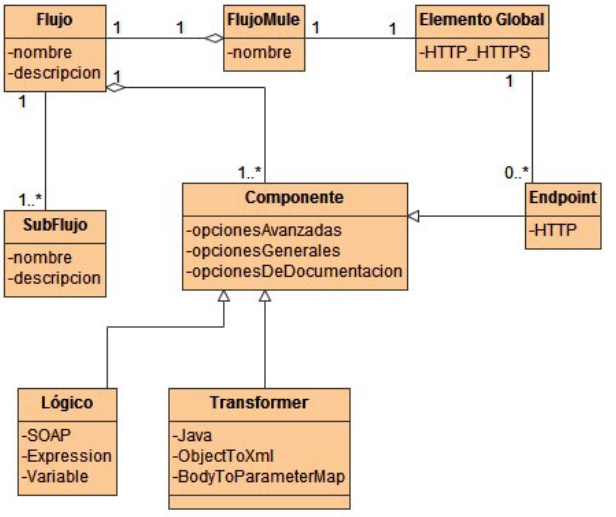

Fig. 5. Relación entre los componentes utilizados para la orquestación en Mule ESB. Fuente: Elaboración propia.

Para indicar las operaciones a consumir de las interfaces de servicios se recurrió al componente SOAP ya que este es el protocolo de encapsulamiento de los mensajes. Además se utilizó el componente *ForEach* para recorrer la lista de objetos resultantes de la invocación de uno de los servicios web.

Debido a que los tres servicios a consumir de SIGENU requieren autenticación básica y dos de ellos certificados de seguridad se tuvo en cuenta crear un elemento global en el flujo del tipo HTTP_HTTPS, donde se especifica la dirección física del certificado, así como las credenciales tanto para el certificado como para la autenticación básica.

El diagrama de la Fig. 5 muestra la relación entre los componentes utilizados en la orquestación.

4.5. Orquestación con la herramienta OSB

Como resultado de la implementación del escenario de integración se obtendrá un servicio cuyo resultado va a ser la combinación de los resultados de tres servicios web. Para esto se crearon tres servicios de negocio correspondientes a los servicios que se quieren orquestar; para cada uno de estos se configuró la seguridad outbound de acuerdo con sus requisitos. Para el servicio que tiene autenticación básica se configuró un componente *Service Account* con las credenciales para su acceso; y se configuró el componente *Service Key Provider* en los servicios de negocio que utilizan certificados digitales como mecanismo de seguridad. Una vez configurados los servicios de negocio, se crea el servicio proxy, que va a utilizar el protocolo SOAP para representar los mensajes de petición y respuesta del mismo. En la configuración de este servicio es donde se define el flujo del mensaje, que indica las acciones a realizar cuando este se invoca. Estas acciones comprenden, de forma general, el llamado a los servicios de negocio creados y la unión de los valores obtenidos para conformar el resultado inicial. En esta orquestación se pueden invocar dos servicios de negocio de forma paralela por lo que se utilizó el componente *Split-Join* para realizar este proceso ya que contiene el nodo *Parallel* que permite realizar llamadas concurrentes a servicios. Además se utilizaron las acciones *Asign* para asignar los valores de los parámetros de entrada del servicio proxy y las peticiones SOAP a los servicios, a variables dentro del contexto del mensaje. Para realizar el llamado a los servicios de negocio se utilizó el nodo *Service Invoque* al cual se le configura el servicio de negocio y la operación que se va a invocar; y con el objetivo de iterar sobre los valores de retorno de estos se utilizó el nodo *For Each***.** Por último se realizó una transformación de los resultados obtenidos del proceso de invocar los servicios de negocio hacia el resultado final, utilizando el componente *XQuery Mapper* basado en lenguaje de consulta XQuery. La Fig. 6 muestra un diagrama de clases donde se relacionan los componentes utilizados en la solución descrita

5. TRABAJOS RELACIONADOS

Por la naturaleza multiprotocolo de la tecnología ESB su utilización se ha evaluado en disímiles sectores donde ha sido necesario integrar información. Específicamente en el área de gestión de procesos universitarios, se encontró que en [1] se propone una estrategia para la interoperabilidad entre aplicaciones utilizando esta tecnología, la que fue evaluada, con buenos resultados, tomando como caso de estudio la integración de diferentes sistemas que interactúan con el entorno docente del Instituto Superior Politécnico José Antonio Echevarría (CUJAE). En [28] se propone la utilización de una herramienta ESB para la integración de sistemas IT en la Universidad Comenius en Bratislava, al igual que en UCLA [29] y en la Universidad de Birmingham; esta última persiguiendo el objetivo de contribuir con el funcionamiento de la existente Arquitectura Orientada a Servicios que a su vez sirve de apoyo para la concepción de edificios inteligentes en el campus universitario [30].

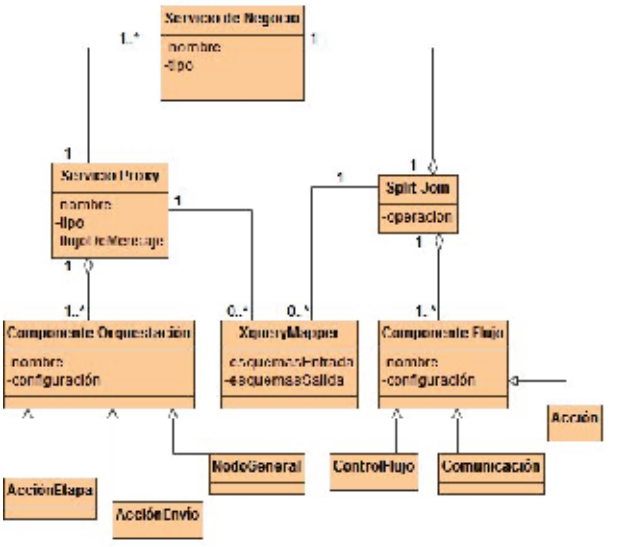

Fig. 6. Relación entre los componentes utilizados para la orquestación en OSB. Fuente: Elaboración propia.

Otra área donde la aplicación de un Bus de Servicios Empresariales se ha estudiado es en la salud, donde aparecen trabajos como el de [31], que ha propuesto una arquitectura basada en SOA con ESB como facilitador tecnológico para resolver los problemas de centralización y aislamiento de los sistemas de salud basados en el paciente en Sudáfrica. Además, una revisión sistemática de la literatura acerca de SOA para el soporte de toma de decisiones clínicas se realizó en [32] donde se refiere la utilización de las capacidades de la tecnología ESB para identificar patrones de distribución de pacientes y servicios de salud en regiones particulares utilizando el sistema Infoway [33]; también para el apoyo a los procesos de la unidad de cuidados intensivos a través de los sistemas HEARTFAID [34], COSARA [35] y para la prevención de enfermedades crónicas con el sistema SOCBeS [36].

En la web existen muchos ejemplos de aplicaciones de la tecnología ESB donde mayormente se explotan sus capacidades de orquestar servicios, soporte de múltiples protocolos, seguridad, transformación de datos y monitoreo. Además que es un concepto bastante apoyado por la industria ya que existe una gran variedad de sistemas ESB que ofrecen diferentes niveles de funcionalidades; en este artículo se han analizado dos de estos.

6. TRABAJOS FUTUROS

Con este trabajo se ha tenido el objetivo de proponer un conjunto de componentes de orquestación de servicios que brindan las herramientas del estudio para solucionar un escenario de integración determinado. Para trabajos futuros quedan, dentro de la capacidad referida, investigar otros componentes y su aplicación en otros tipos de escenarios de integración, extendiéndose la investigación a otras capacidades de la tecnología ESB.

7. CONCLUSIONES

El trabajo con Mule ESB, no es tan engorroso debido a que cuenta con una herramienta de apoyo visual para el diseño del flujo ESB, denominada Mule Studio. Los componentes con que cuenta Mule Studio abarcan en su gran mayoría las principales acciones que se quieren realizar dentro del flujo, siendo el caso de recorrer el mensaje que viaja entre componentes, la posibilidad de definir variables dentro del flujo y poder utilizarlas en cualquier punto del mismo. Al contener un componente como el DataMapper nos da la oportunidad de realizar transformaciones tales como de un POJO a XML, de XML a JSON, entre otras tantas.

En cuanto al tratamiento de la seguridad no existen grandes inconvenientes para el consumo de servicios que hacen uso de la autenticación básica y de certificados de seguridad. El despliegue es sencillo y demanda pocos recursos de hardware y software.

La herramienta Oracle Service Bus cuenta con un conjunto de componentes para configurar la lógica de procesamiento de mensajes, el cual ocurre a través de consultas en los lenguajes XQuery y XPath sobre una estructura de mensaje definida por la herramienta. Los componentes abarcan un conjunto de funcionalidades como insertar, eliminar, validar y reescribir partes del mensaje, invocar servicios de forma secuencial o de forma paralela, crear variables para trabajar con partes del mensaje, recorrer colecciones de objetos, hacer transformaciones de tipos de datos, entro otras.

Por otra parte, el tratamiento de la seguridad en OSB es un proceso complejo que requiere de especialistas adiestrados en el tema, porque hay que tener en cuenta que la herramienta basa su esquema de se-

guridad en el del servidor de aplicaciones *weblogic* y que además puede importar políticas de seguridad realizadas en otras herramientas.

La utilización de la tecnología ESB demanda amplios conocimientos de las características técnicas del entorno en que se aplica y requiere de un equipo multidisciplinario adiestrado en varios temas como desarrollo de servicios web en varias tecnologías, tratamiento con lenguajes estándares basados en XML, cuestiones de seguridad, entre otros.

REFERENCIAS

[1] M. Díaz Rosales, "Estrategia para la interoperabilidad entre aplicaciones a través de un Bus de Servicios Empresariales" in *MSc.dissertation, directed by D.J.M. Prieto.* Instituto Superior Politécnico José Antonio Echevarría, La Habana. 2012.

[2] Microsoft, MSDN. *Introduction to the Microsoft ESB Guidance*. Jun 2015. URL: https://msdn.microsoft.com/en-us/library/ff648282.aspx.

[3] K. Vollmer, "The Forrester Wave: Enterprise Service Bus, Q2 2011", 2011. Disponible en: http://semanticommunity.info/@api/deki/files/17783/2011_forrester-esb-wave.pdf

[4] J. Dirksen, T. Rademakers & . "Open-Source ESBs in Action Example Implementations in Mule and Service Mix".New York: Manning Publications Co, 2008. ISBN: 1933988215

[5] Dalila Kindelan Castro, Deborah Oliva Alfonso. "Informe Técnico sobre la Integración de Información". *Complejo de Investigaciones de Tecnologías Integradas,* 2011.

[6] Geebelen, Kristof. "Adaptive Workflow Composition in Service-Based Systems". *Doctorate. dissertation, directed by Dr. ir. W. Joosen & Dr. E. Truyen.* Katholieke Universiteit Leuven. Belgium. 2012.

[7] Barai, Vincenzo Caselli, Binildas C. A. & Malhar. "Web Services and SOA". Service Oriented Architecture with Java. Birmigham: Packt Publishing, 2008.

[8] W3C. *Web Service Architecture*. 2004. URL: http://www.w3.org/TR/ws-arch/.

[9] Mehta, Bhakti. "RESTful Java Patterns and Best Practices".Birmigham: Packt Publishing, 2014. isbn: 978-1-78328-797-0

[10] Erl, Thomas. "SOA Principles of Service Design". Upper Saddle River, NJ,USA: Prentice Hall PTR, 2007. isbn: 0132344823

[11] Trcek, Damjan Kovac; Denis."A Survey of Web service Orchestration and Choreography with Formal Models", 2011.

[12] B.Juric, Matjaz. *A Hands-on Introduction to BPEL*. 2009. URL: http://www.oracle.com/technetwork/articles/matjaz-bpel1-090575.html.

[13] W3C, Web Services Choreography Description Language Version 1.0. 2005, W3C.

[14] W3C, Web Service Choreography Interface (WSCI), 2002, W3C.

[15] OASIS. "Business Process Execution Language for Web Services Version 2.0", 2007. Disponible en: http://docs.oasis-open.org/wsbpel/2.0/wsbpel-v2.0.pdf

[16] Chappell, David. "Enterprise Service Bus". Sebastopol: O´Reilly, 2004. isbn: 978-0-596-55650-1

[17] Centeractive. *Centeractive. Enterprise Service Bus*, 2012. URL: http://www.centeractive.com/content/enterprise-service-bus.

[18] Microsoft. "Guidelines for Applicaction Integration". Microsoft Press, 2004. isbn:

[19] Mulesoft, *Introducing Mule Concepts*, 2012.

[20] D. Dossot, J. D´Emic & V. Romero. "Mule in action, Second Edition". New York: Manning Publications Co, 2014. isbn: 9781617290824

[21] Floyd Jones, Legacy authors, "Concepts and Architecture for Oracle Service Bus". 2010, ORACLE.

[22] Rieber, Jeff Davies; David Schorow; Samrat Ray & David, *The Definitive Guide to SOA: Oracle Service Bus, Second Edition*, S. Anglin, Editor. Apress: Berkeley, California, 2008, p. 524.

[23] Herrera, Arianna Tomé Ulloa; Remeberto. "Evaluación Crítica de Oracle Service Bus". *Ing.dissertation, directed by I.D.O. Alfonso.* Instituto Superior Politécnico José Antonio Echevarría, 2010.

[24] Authors, Floyd Jones & Legacy, "Oracle Fusion Middleware Developer´s Guide for Oracle Service Bus", ORACLE, 2010.

[25] Valdés, Eva Flores, "Implementación de lenguajes de contrato electrónico en Oracle Service Bus," in *Revista Cubana de Ciencias Informáticas,* 2015, UCI: La Habana p. 63-77.

[26] Jeff Davies, David Schorow, Samrat Ray, David Rieber. "The Definitive Guide to SOA: Oracle Service Bus, Second Edition". New York: Apress, 2008. isbn:

[27] Castillo, Isela Mendoza del. "Particularidades de la Implementación del Back-End para un Sistema de Información en Ambiente Web. Caso de Estudio: Intranet de Ingeniería Informática". *Ing. dissertation, directed by D.F.O.F. Peña.*Instituto Superior Politécnico José Antonio Echavarría. La Habana. 2010.

[28] Pálos, Pavol Mederly; Gustav, "Enterprise Service Bus at Comenius University in Bratislava, in Advances in Software Engineering Techniques," *4th IFIP TC2 Central and East European Conference on software Engineering Techniques*, CEE-SET 2009 2008, Springer: Krakow, Poland.

[29] UCLA IT Services, "UCLA Enterprise Service Bus". 2013. URL: http://www.csg.ucla.edu/documents/2010/esbproposalandarchitecture2013-05-28.pdf

[30] S. Hipwell, "Developing smart campuses–A working model," in *2014 International Conference on Intelligent Green Building and Smart Grid* (IGBSG), 2014.

[31] H. D. Masethe, O. O. Olugbara, S. O. Ojo & A. O.Adewuni,."Integration of Health Data using Enterprise Service Bus," In *Proceedings of the World Congress on Engineering and Computer Science San Francisco*, USA, 2013.

[32] Huser, S. Rodríguez Loya, Kensaku Kawamoto; C. Chatwing, Vojtech."Service Oriented Architecture for Clinical Decision Support: A Systematic Review and Future Directions ". *Journal of Medical Systems,* vol. 38, no.12, 2014.

[33] Sartipi K, Yarmand MH, Down D. G., "Mined-Knowledge and Decision Support Services in Electronic Health in International Workshop on Systems in SOA Environments SDSOA '07: ICSE Workshops 2007". IEEE: Minneapolis. p. 10, 2007.

[34] Calemis, Achilles Kameas; Ioannis. "Pervasive Systems in Health Care," *Handbook of Ambient Intelligence and Smart Environment*, Springer US, pp. 315-346, 2010. doi: 10.1007/978-0-387-93808-0_12

[35] Turck, Bruno van de Bossche; Sofie van Hoecke; Chris Danneels, J. Decruyenare, B. Dhoedt, Filip de."Design of a Jain SLEE/ESB- based platform for routing medical data in the ICU". *Computer Methods and Programs in biomedicine,* vol.91, pp.265-277, 2008.

[36] Serhani, Abdelghani Benharret; Mohamed Adel, "Novel cloud and SOA- Based framework for E-health monitoring using wireless biosensors". *IEEE Journal of Biomedical and Health Informatics,* vol.18, no.1, pp.46-55, 2014.