Revista Venezolana de Computación
ISSN: 2244-7040
http://www.svc.net.ve/revecom
Vol. 3, No. 2, pp. 13-25, Diciembre 2016
Selección de los Mejores Artículos de CoNCISa 2016

Diseño de Arquitecturas de Referencia para Líneas de Productos de Software Orientadas a Servicios: Revisión Documental Sistemática

Juan C. Herrera[1], Francisca Losavio[2], Oscar Ordaz[3]
jchr1982@gmail.com, francislosavio@gmail.com, oscarordaz55@gmail.com

[1] PFG Informática para la Gestión Social, Universidad Bolivariana de Venezuela, Caracas, Venezuela
[2] Escuela de Computación, Universidad Central de Venezuela, Caracas, Venezuela
[3] Escuela de Matemáticas, Universidad Central de Venezuela, Caracas, Venezuela

Resumen: Una Revisión Documental Sistemática (RDS) permite extraer conocimiento sobre un tema de investigación; esto se hace mediante preguntas adecuadas, a partir del gran volumen de información disponible en general en la literatura y en particular en internet. El objetivo del trabajo es realizar una RDS en el tema de Líneas de Productos de Software Orientadas a Servicios (LPSOS), el cual integra los enfoques de Líneas de Productos de Software (LPS) y de Arquitecturas Orientadas a Servicios (SOA). La intención es que a partir de la RDS se identifiquen métodos (fases, etapas, actividades, artefactos y roles) para el diseño del artefacto principal de las LPS, la Arquitectura de Referencia (AR) siguiendo un modelo de arquitectura empresarial SOA y conformada por Servicios Web. Los resultados de la RDS servirán de guía para definir un proceso sistemático para el diseño de una AR en el contexto LPSOS. Esta arquitectura contemplará el tratamiento de requisitos de calidad en las etapas tempranas del desarrollo de LPSOS para garantizar su evolución, y el modelado de la variabilidad para la derivación de productos concretos, aspectos esenciales en el contexto de producción industrial de software.

Palabras Clave: Líneas de Productos de Software; Arquitectura Orientada a Servicios (SOA); Líneas de Productos de Software Orientadas a Servicios (LPSOS); Revisión Documental Sistemática (RDS); Arquitectura de Referencia; Servicios Web.

Abstract: A Systematic Review (SR) allows to extract knowledge about a research topic; this is done by formulating adequate questions from the large volume of information available in the literature in general and particularly searching on the Internet. The goal of the work is to perform an SR on the subject of Service Oriented Software Product Lines (SOSPL), which integrates the approaches of Software Product Lines (SPL) and Service Oriented Architectures (SOA). The intention is to use SR methods (phases, stages, activities, artifacts and roles) for the design the Reference Architecture (RA), the main artifact of LPS, following the SOA enterprise architecture model conformed by Web Services. The results of the SR will provide guidance to define a systematic process to design a RA in the SOSPL context. This architecture will cover the treatment of quality requirements at early stages of SOSPL development to ensure its evolution, and the variability modeling for the derivation of concrete products, which are essential issues in the context of software industrial production.

Keywords: Software Product Lines; Service Oriented Architecture (SOA); Service Oriented Software Product Lines (SOSPL); Systematic Review (SR); Reference Architecture (RA); Web Services.

I. Introducción

El desarrollo de sistemas de software de gran complejidad y de gran escala representa un desafío para los investigadores y desarrolladores en el área de la *Ingeniería de Software (IS)*; surge entonces la tendencia al desarrollo basado en componentes reutilizables, para configurar nuevos sistemas con rapidez y de más bajo costo [1][2][3].

El enfoque de *Líneas de Productos de Software (LPS)* permite la construcción de soluciones individuales basadas en un repositorio de componentes de software reutilizables. El hecho de proporcionar soluciones individuales responde a diversas necesidades en el software con respecto a la funcionalidad, la tecnología subyacente y las propiedades no funcionales, como por ejemplo rendimiento y espacio de memoria [4]. Las LPS prometen distintos beneficios [4][5], de los cuales los más importantes son: a) adaptabilidad a las necesidades del cliente,

b) reducción de costos, c) mejora de la calidad global, d) evolución, y e) disminución del tiempo de comercialización. Este enfoque representa un gran reto para los investigadores y desarrolladores, principalmente por la tarea de construir artefactos (*activos* o del inglés *assets*) suficientemente genéricos para poder ser reutilizados en la configuración de nuevos sistemas acordes con las necesidades de los clientes [2][3]. Para el abordaje adecuado de este enfoque, la *Ingeniería de Líneas de Productos de Software (ILPS)* ofrece métodos y técnicas eficaces para un desarrollo de software basado en la reutilización, con el fin de: a) apoyar arquitecturas de software configurables o instanciables y b) permitir la personalización masiva de los sistemas de software [6][7]. Ha sido reconocida como un enfoque exitoso para la gestión de la variabilidad y de la reutilización. La ILPS consta de dos ciclos de vida principales: *Ingeniería del Dominio (ID)* e *Ingeniería de la Aplicación (IA)* [8]. La ID abarca el análisis y la identificación del ámbito de aplicación de la LPS, que incluye la captura de todo el conocimiento sobre el dominio de interés a través del modelado de partes comunes (del inglés "commonality") y de partes variables o puntos de variación (del inglés "variation points") que están presentes en la *Arquitectura de Referencia (AR)*, la cual es una arquitectura genérica basada en un esquema instanciable que dirige la derivación de productos concretos de la familia LPS.

Por otra parte, la *Arquitectura Orientada a Servicios* (del inglés *Service Oriented Architecture (SOA)*) [9][10][11] es un modelo de arquitectura empresarial distribuida, centrado en la comunicación en redes; su objetivo central es la integración y desarrollo rápido de sistemas empresariales en una organización o dominio específico. Un *servicio* se define como una unidad discreta de una funcionalidad del negocio, que está disponible a través de una *interfaz estándar* del servicio, mediante la cual el servicio se comunica con otros servicios [12]; el servicio puede ser visto como un componente de software reutilizable. Una implementación muy conocida de servicios son los Servicios Web, en el contexto de la WWW [9]. SOA especifica lineamientos para la construcción independiente de servicios alineados al negocio que pueden ser combinados a partir de procesos de negocio de alto nivel y trasladados a soluciones computacionales en el contexto empresarial. El valor real de SOA no es solo la provisión adecuada de servicios, sino más bien cuando los servicios reutilizables se combinan para "implementar" procesos de negocios ágiles y flexibles. SOA ha demostrado su rentabilidad en el desarrollo de sistemas de software interoperables, reutilizables, adaptables y de rápido desarrollo, por lo cual existen importantes ventajas mutuas en la convergencia de SOA y LPS [11][13][14], dando origen al enfoque de *Líneas de Productos de Software Orientadas a Servicios (LPSOS)*. La integración de los enfoques LPS y SOA requiere que las organizaciones de desarrollo de LPS apliquen la gestión de la variabilidad respecto a los servicios que son los componentes esenciales de SOA, y que los desarrolladores de sistemas orientados a servicios apliquen técnicas que usan los desarrolladores de LPS. Sin embargo esta tarea no es trivial; la necesidad de variación en cuanto a servicios, puede ser impulsada por las condiciones del mercado, oportunidades del negocio, las nuevas tecnologías y otros factores relacionados con la empresa. Esta necesidad puede expresarse en forma de objetivos de negocio establecidos por las organizaciones que desarrollan LPSOS [13]. Ahora bien, en un contexto de LPSOS, particularmente para la construcción de una AR, deben manejarse tres problemas principales: (1) la representación de la AR conformada por componentes y conectores [15], en donde los componentes son servicios y los conectores son mensajes entre servicios, incluyendo el traslado de servicios a componentes arquitecturales; (2) la gestión de las propiedades de calidad requeridas por los componentes que representan funcionalidades para asegurar la completitud y evolución de la AR; (3) la gestión de la variabilidad funcional y no funcional. Las soluciones a estos problemas determinarán los pasos a seguir para definir un método de diseño de una AR basada en servicios en un contexto LPS.

En los estudios anteriormente referenciados se han reflejado los problemas mencionados, aún no completamente resueltos, para algunos de los cuales este trabajo pretende proporcionar una guía para su solución. El objetivo de esta investigación, se centra principalmente en una Revisión Documental Sistemática de la literatura (RDS) [16] de trabajos recientes sobre diseño de una AR basada en servicios y configurada como una arquitectura de software [15] para LPS y no como una arquitectura empresarial, para identificar métodos sistemáticos de diseño que respondan a los tres problemas identificados anteriormente y que además contemplen:

- artefactos de entrada y de salida,
- enfoque de diseño utilizado para la AR,
- roles de quienes participan en las actividades del modelo de proceso del método.

Una RDS [16] permite extraer el conocimiento sobre el estado del arte en un tema específico de investigación; este conocimiento es extraído mediante preguntas adecuadas, a partir de un gran volumen de información disponible. En consecuencia una RDS que aborde el tema de métodos de desarrollo de LPSOS, es indispensable para determinar los alcances y limitaciones de las propuestas encontradas. Para alcanzar este objetivo, nuestra RDS se apoya en un marco de evaluación conformado por la integración de los aspectos de interés de tres propuestas de evaluación [17][18][19], que abordan las siguientes temáticas: LPS, SOA y AR-SOA. Este marco de evaluación posteriormente será utilizado para comparar los "mejores" métodos encontrados por la RDS, es decir aquellos que satisfacen más del 50% del total de las 33 propiedades descritas en las Tablas VII, VIII, IX.

Este trabajo está estructurado de la siguiente manera, además de esta introducción: la Sección II presenta algunos aspectos a ser considerados para realizar la RDS; en la Sección III se define el marco integrado de evaluación; la Sección IV presenta el análisis de la RDS realizada; la Sección V discute los resultados obtenidos. Finalmente, la Sección VI presenta las conclusiones y las perspectivas.

II. Aspectos a Ser Considerados en la RDS

El enfoque LPSOS requiere considerar los siguientes elementos: la *Gestión de Procesos de Negocios (GPN)* o del inglés *Business Process Management (BPM)*, la gestión de la variabilidad funcional y no funcional, la representación de una

Arquitectura de Referencia Orientada a Servicios y la gestión de la calidad. Estos aspectos son considerados prioritarios en una RDS de dos Santos Rocha y Fantinato [7] que presenta enfoques para la aplicación conjunta de LPS y GPN, incluyendo el soporte de SOA para GPN, y en la cual se concluye que la combinación LPS, BPM y SOA es factible y beneficiosa para la construcción de un proceso sistemático integrado para elaborar una LPSOS. A continuación se describen brevemente cada uno de estos aspectos.

A. Gestión de Procesos de Negocios

GPN en [20] define un *proceso de negocio* como un conjunto de actividades que se realizan en coordinación con un entorno organizacional y técnico. Estas actividades se desempeñan de manera conjunta para realizar los objetivos del negocio. Cada proceso de negocio es generado por una sola organización, pero puede interactuar con los procesos de negocios generados por otras organizaciones. Las partes fundamentales de GPN son su ciclo de vida y el lenguaje de modelado de procesos de negocio del inglés *Business Process Management Language (BPML)* [20]. Existen varios BPMLs concebidos en la academia y la industria que se utilizan para representar el flujo de trabajo y los artefactos y/o producidos utilizados durante la realización de las actividades en el ciclo de vida. Algunos de estos son el Diagrama de Actividades (UML 2.0) y la Notación de Procesos de Negocios [21], los cuales fueron evaluados por un framework en [22]. Esta evaluación determinó que el *Business Process Modeling Notation (BPMN)* es uno de los BPMLs más utilizado en la fase de diseño de un ciclo de vida GPN. Representa los procesos como actividades centradas en el control de flujo. El lenguaje de modelado debe proporcionar elementos notacionales y mecanismos de apoyo a aspectos tales como la especificación de las actividades, el flujo de secuencia, el flujo de datos y los eventos. En cambio, no soporta especificaciones de aspectos o requisitos no funcionales. Por otra parte, la ejecución de los procesos de negocio, además de la selección de una plataforma de ejecución, como por ejemplo SOA, puede requerir de otros lenguajes. Por ejemplo el Lenguaje *Business Process Execution Language (BPEL)* es utilizado para la ejecución de los procesos de negocio a través de la composición de servicios por orquestación [20]. No obstante, la OMG[1] con la definición de la semántica de cada elemento de la notación en BPMN 2.0, permitió la posibilidad de ejecutar los modelos sin necesidad de algún otro lenguaje de apoyo. Por ejemplo la herramienta Activiti[2] que implementa BPMN. En la actualidad, la mayoría de los enfoques de GPN enfatizan el desarrollo de un proceso de negocio, y a partir de este mismo se derivan muchas variantes, que están especializadas según diversas necesidades de la organización. Los procesos de negocio pueden tener piezas comunes para un grupo (es decir, la familia) de diferentes casos de aplicación, implicando cierta variabilidad en la selección de las actividades de un proceso de negocio. Este aspecto podría ser utilizado para LPS, donde una funcionalidad común puede ser desarrollada como un activo reutilizable para la creación de nuevas variantes de los procesos de negocio en un dominio [13][20].

[1] Object Management Group
[2] http://activiti.org

B. Gestión de la Variabilidad

La *Gestión de la Variabilidad* (del inglés *Variation Management*) [13] se aplica tanto a SOA como a LPS; los puntos de variación pueden ser implementados ya sea por un único servicio (donde una interfaz de servicios puede ofrecer parametrización por *QoS (Quality of Service)* o algún otro mecanismo) o a través de servicios similares que traten cada variación. Mohabbati [11], indica que la variabilidad es gestionada teniendo varios servicios que tienen la misma funcionalidad, pero difieren en su QoS.

C. Arquitecturas de Referencia en LPS y SOA

Otro aspecto importante corresponde a la construcción de la AR, y en particular a las *Arquitecturas de Referencia Orientadas a Servicios (AROS)* que apoyen la reutilización sistemática de los activos principales para la configuración de productos de software concretos. En la RDS [23] se indica que existen diferencias entre las *Arquitecturas de Líneas de Productos (ALP)* y las AR: ALP tienden a ser menos abstractas que las AR, pero más abstractas que las arquitecturas de software concretas, es decir, las ALP son un tipo de AR [11][23]. Con respecto a nuestra RDS, AR y ALP son consideradas similares. Las AROS se han propuesto como un tipo de AR; lo que debe estar claro es que las AROS en el contexto SOA son un modelo de arquitecturas empresariales para la integración de aplicaciones en la empresa, no son arquitecturas de software "reales" en el sentido de Garlan y Shaw [15]. En cambio en el contexto LPS, AROS son arquitecturas de software genéricas e instanciables a partir de un modelo de variabilidad, para generar sistemas concretos orientados a servicios de la familia LPS. Una AROS-LPS será entonces para nosotros una AR en un contexto de LPSOS. Varias AROS bajo SOA han sido propuestas para sistemas de gobierno, entornos de trabajo colaborativo, o sistemas de atención de la salud; generalmente no son AROS "puras" solo conformadas por servicios, sino que incluyen componentes que no son servicios. Una RDS realizada sobre las AROS bajo LPS en [7], indica que faltan directrices sobre cómo definirlas, y en particular como definir las AROS que soporten un alto grado de variabilidad [11].

D. Gestión de la Calidad en Servicios

En los actuales momentos no se concibe un sistema de software que no tenga la calidad necesaria que satisfaga las expectativas de los usuarios finales. Más aun, las propiedades de calidad para una LPSOS son fundamentales para garantizar su completitud y carácter evolutivo. Por ello, es vital abordar las características de calidad de estos sistemas desde una etapa temprana en el ciclo de vida del desarrollo de software. Particularmente esto se debe hacer en el ciclo de vida de la ID en el proceso de una *ILPS Orientada a Servicios* (ILPSOS). Los servicios exponen propiedades de calidad de bajo nivel a las que responden en tiempo de ejecución, para intercomunicar con otros servicios; los valores de los QoS se ofrecen a través de parámetros. En general, lo que ocurre con las propiedades de calidad de alto nivel a las cuales también deben responder los servicios compuestos, cuando representan componentes funcionales en el caso de AROS, no se trata en detalles. Los *Servicios Web* (en inglés *Web Services (WS))* son tipos de servicios que pueden ser descubiertos, especificados y accedidos utilizando XML y protocolos Web estándar. El

elemento central de un WS está en la definición estándar de su interfaz en WSDL[3], que contiene entre otras informaciones, los QoS. La descripción de un WS tiene dos partes: - una definición abstracta que describe el servicio como un componente, con su interfaz y operaciones; - una definición que describe los enlaces concretos de las operaciones hacia puntos de acceso, que contienen una descripción de los datos, una dirección física y la información del protocolo de comunicación [24]. En un entorno de una aplicación que utiliza WS, se distinguen tres funciones: intermediario, proveedor y consumidor de servicios. El proveedor registra un servicio en el intermediario o "broker", por ejemplo en un repositorio UDDI[4]. El consumidor descubre un servicio gestionado por el intermediario y llama al servicio del proveedor. Todas las partes utilizan protocolos tipo SOAP[5], un protocolo estándar basado en XML para el manejo de respuestas y solicitudes de WS [24]. Está claro que un WS, como componente de una aplicación, también debe responder a propiedades de calidad de alto nivel, las cuales no necesariamente son observables en ejecución, por ejemplo persistencia de la información y políticas de confidencialidad.

III. Marcos para la Evaluación de Métodos para el Desarrollo de AR-LPS, SOA, AR-SOA

A. Framework para la Evaluación de los Métodos

El marco de evaluación o *framework*, constituido por las Tablas I, II y III, es utilizado como herramienta de análisis. La Tabla I muestra los criterios para evaluar métodos de diseño AR-LPS. La Tabla II muestra criterios para evaluar métodos de desarrollo de sistemas orientados a servicios bajo SOA. La Tabla III presenta criterios para evaluar métodos de desarrollo de AR-SOA. El objetivo de este framework de evaluación no es la comparación de métodos, sino proporcionar un panorama de métodos actuales para tomar decisiones en cuanto a un método general para el diseño de una arquitectura AROS-LPS.

AROS contempla tres estrategias de desarrollo:

- *bottom-up o ascendente (reactivo o extractivo):* derivar los servicios a partir de componentes existentes, es decir de aplicaciones heredadas o de una AR ya realizada, como por ejemplo la identificada a partir de un proceso "bottom-up" que considera la refactorización de productos existentes del mercado.
- *top-down o descendente o proactivo:* derivar los servicios a partir de la especificación del modelo del dominio expresado mediante procesos de negocios.
- *middle-out o meet-in-the-middle o híbrido:* partir de la especificación de los procesos de negocio como orquestaciones de servicios y relacionarlos con los componentes de una AR ya existente.

Para la RDS de este trabajo, se buscan los métodos o enfoques existentes (estado del arte) que describan de forma sistemática las etapas, actividades, roles y artefactos necesarios para realizar una AR en un contexto LPSOS (AROS-LPS), utilizando cualquiera de los tres frameworks mencionados [17][18][19].

La aplicación del marco de evaluación proporcionará la base para la definición de categorías detalladas de elementos o aspectos presentes en un criterio. Mediante un conjunto de preguntas, este estudio intenta abordar el alcance de los métodos para identificar las diferencias y similitudes que estos presentan. Como no existe un marco específico para la evaluación de enfoques LPSOS, se van a combinar los frameworks de las Tablas I, II y III, en un marco único que incluya todos los criterios de evaluación. A continuación se describen en detalles los criterios:

Criterios del marco de evaluación para métodos de desarrollo de AR-LPS [17]:

Hay cuatro criterios esenciales para la evaluación de estos métodos:

- *Contexto del método:* situación del problema, objetivo específico, ciclo de vida abordado.
- *Usuario del método:* personas que solucionan el problema, que rol desempeña, guía proporcionada para aplicar el método.
- *Componentes del método o proceso para la resolución del problema:* 1) modelos subyacentes, 2) lenguaje, 3) definición de los pasos, su secuencia, 4) artefactos, 5) vistas arquitecturales, como el modelo "4+1 vistas" de Kruchten [25]), 6) variabilidad (nivel de abstracción donde la variabilidad es manifiesta: a nivel de proceso de negocio, a nivel de los componentes), 7) herramientas que facilitan la realización del método.
- *Gestión de la calidad (nuevo criterio):* ¿Cómo el método soporta la especificación de las propiedades de calidad y/o requisitos no funcionales?; el aseguramiento de la calidad del producto es una actividad crucial para el éxito de la industria del software, pero es más importante cuando se trata del desarrollo de la LPS, dado que la reutilización masiva de activos de software hace que sus atributos de calidad (o propiedades medibles de un artefacto de software) impacten en la calidad de los productos de la LPS [26].
- *Validación del método:* casos de estudios donde se compruebe la efectividad del método, el estilo arquitectural elegido satisface las expectativas de los usuarios finales.

Criterios del marco de evaluación para métodos de desarrollo de sistemas bajo el modelo SOA [18]:

Hay cinco criterios esenciales para la evaluación de un método para desarrollo de sistemas o aplicaciones aisladas bajo el modelo SOA de arquitectura empresarial, que no involucran AR sino una arquitectura concreta de un producto:

- *Concepto de servicio:* para analizar si los conceptos de servicio, servicios de negocios y software, son soportados; un servicio de negocio es un conjunto específico de acciones que son llevadas a cabo por una organización, un servicio de software expone las funcionalidades de la

[3] Web Services Definition Language
[4] Universal Description, Discovery and Integration
[5] Simple Object Access Protocol

aplicación que pueden ser reutilizados y compuestos basados en las necesidades u objetivos del negocio, por lo tanto un servicio de software soporta la ejecución de un servicio de negocio.

- *Estrategia de realización o desarrollo:* top-down, bottom-up, middle-out o meet-in-the-middle que combina ambos; esta última estrategia es la más recomendada para implementar SOA.
- *Cobertura del ciclo de vida* o fases principales para el desarrollo bajo SOA; según SOMA [27] son: identificación, especificación y realización.
- *Grado de granularidad del análisis:* el nivel de detalle utilizado para describir el método, que va desde un nivel bastante descriptivo, con gran cantidad de detalles, hasta un nivel más abstracto, para describir un proceso más general, flexible y adaptable.
- *Accesibilidad y validez:* Indica que el método debe proveer documentación accesible y poder ser validado.

Criterios del marco de evaluación para métodos de desarrollo de AR-SOA [19]:

Las arquitecturas de software [15] juegan un papel importante en la determinación de la calidad del sistema, ya que forman la columna vertebral de cualquier sistema intensivo de software exitoso; el estilo arquitectónico garantiza por sí solo el cumplimiento de propiedades de calidad globales del dominio. En el contexto LPS, las AR reúnen el conocimiento del dominio y son los instrumentos que proveen esquemas reutilizables. Entre los principales beneficios de AR, se encuentran una mejor comprensión del dominio, el establecimiento de un vocabulario común, la reutilización de activos, la consistencia en la representación del sistema, y un menor tiempo de comercialización. SOA, como modelo de *arquitectura de referencia empresarial (AR-SOA)* ha tomado la atención en los últimos años. Facilita la integración mediante la interoperabilidad de servicios, la escalabilidad y la reutilización, ya que los sistemas basados en SOA son independientes respecto al lenguaje de programación y a la plataforma de ejecución utilizada. El estilo arquitectónico de sistemas de software empresariales orientados a servicios "puros", que no involucran otro tipo de sistemas no orientados a servicios, es basado en eventos [15], siguiendo un modelo cliente–servidor para la distribución y la comunicación en redes.

El marco de evaluación AR-SOA surge de responder la pregunta: ¿Cuáles características de SOA han sido consideradas durante el diseño y desarrollo de la AR-SOA? Se presentan tres criterios:

- *Publicación del servicio:* La AR-SOA debe permitir la publicación de la descripción de los servicios directamente a los consumidores, a través de intermediarios, por ejemplo UDDI.
- *Calidad del servicio (QoS):* La AR-SOA debe permitir utilizar mecanismos para capturar, controlar, registrar y señalar el incumplimiento de requisitos no funcionales descritos en los acuerdos de servicio por su interfaz. Estos acuerdos especifican las características de calidad que proporciona el servicio en ejecución, como el tiempo de respuesta, rendimiento, disponibilidad, entre otros.
- *Composición de los servicios:* es el proceso de construir un nuevo servicio a partir de servicios existentes [9]. Todos los servicios se comunican por paso de mensajes, a través de protocolos. La AR-SOA debe permitir la composición de los servicios: a través de los procesos de negocio, mediante una orquestación coordinada de servicios que enfatiza el paso de mensajes, o mediante la coreografía, más orientada hacia los aspectos funcionales que el servicio debe proporcionar.

B. Criterios y Elementos del Marco Único de Evaluación basado en los Tres Framework AR-LPS, SOA y AR-SOA

El objetivo del *Marco Único de Evaluación (MUE)* no es solo proporcionar una visión general de los actuales métodos de ingeniería para la LPSOS, sino además indagar cómo los métodos difieren en cuanto a criterios de diseño**.** Esto es especificado en la sección del informe de la RDS, donde se muestran los resultados de nuestra evaluación. La aplicación de MUE proporciona la base para la definición detallada de los elementos presentes en los criterios. Mediante preguntas, este marco intenta abordar por ejemplo, madurez, sentido práctico y alcance de los métodos para encontrar las similitudes y diferencias, además, de cómo se identifican los servicios a partir del modelo de procesos de negocio, y cuáles son los componentes que constituirán una AROS-LPS y que implementarán los servicios, incluyendo la relación entre estos componentes y el modelo de variabilidad.

A continuación, las Tablas I, II, III describen criterios y elementos que conforman el MUE, con sus respectivas preguntas de análisis. Estas preguntas ayudan a la captura de información relevante para cada uno de los criterios de evaluación. El MUE se deriva integrando todos los criterios de evaluación y será utilizado en la RDS para efectuar el análisis de los artículos seleccionados como *estudios primarios*: son aquellas investigaciones relevantes que quedan para el análisis, luego de haberse aplicado los criterios RDS de inclusión, exclusión y calidad:

Tabla I: Criterios y Elementos de Evaluación para Métodos de Desarrollo de AR-LPS, Adaptado de [17]

Criterios	Elementos	Preguntas
Dominio de la LPS	Ámbito	¿Cuál es/son el/los dominio(s) de la LPS en la que se enfoca el método? Ejemplo: dominio de los sistemas de información.
Contexto del método	Objetivo específico	¿Cuál es el objetivo específico del método?
	Fase(s) de línea de productos	¿Qué fases de la LPS cubre el método? Por ejemplo, el método abarca solamente las fases de análisis y diseño en el ciclo de vida de la ID.
	Entradas del método	¿Cuál es (son) la (s) entrada (s) del método?
	Salidas del método	¿Cuáles son la (s) salida (s) del método?
Usuario del	Grupo	¿Quiénes son los actores que

método	destinatario	intervienen en el método?
	Motivación	¿Cuáles son los beneficios que percibe el usuario al utilizar el método?
	Habilidades necesarias	¿Qué habilidades necesita el usuario para cumplir las tareas requeridas por el método?
	Orientación	¿Cómo el método guía al usuario mientras se aplica el método?
Componentes del método	Estructura del método	¿Cuáles son los pasos, fases y/o actividades del proceso de diseño que se utilizan para llevar a cabo un objetivo específico del método?
	Artefactos	¿Cuáles son los artefactos creados y gestionados por el método?
	Vistas arquitecturales	¿Cuáles son las vistas arquitecturales que se consideran en el método?
	Lenguaje/ notación	¿El método define un lenguaje o notación para representar los modelos, diagramas y otros artefactos que produce?
	Variabilidad	¿Cómo el método soporta la expresión de la variabilidad funcional y no funcional?
	Herramientas	¿Cuáles son las herramientas y/o técnicas que apoyan el método?
Gestión de la calidad en el método	Propiedades de calidad	¿Cómo el método soporta la especificación de las propiedades de calidad y/o requisitos no funcionales?
Validación del método	Madurez	¿Se ha validado el método en casos de estudios prácticos industriales?
	Calidad de la arquitectura	¿Cómo el método valida la calidad del diseño de la AR-LPS?

Tabla II: Criterios y Elementos de Evaluación para Métodos de Desarrollo de Sistemas bajo el Modelo SOA, Adaptado de [18]

Criterios	Elementos	Preguntas
Concepto de servicio	Servicios de negocios	¿Cuál es el concepto de servicio soportado?
	Servicios de software o WS	
	Ambos	
Estrategia de realización o desarrollo	Top-down	¿Cuál es el tipo de estrategia utilizada para el desarrollo bajo SOA?
	Bottom-up	
	Middle-out	
Cobertura del ciclo de vida	Identificación	¿Cuál es el grado de cobertura con relación al ciclo de vida SOA?
	Especificación	
	Realización	
Grado de granularidad del análisis	Bajo	¿Cuál es el grado de granularidad del nivel de abstracción del método?
	Moderado	
	Alto	
Accesibilidad y validez	Documentación no accesible	¿La documentación del método es apropiada y sirve de guía para su uso en la práctica?
	Parcialmente documentado	
	Bien documentado (Caso de estudio)	

Tabla III: Criterios y Elementos de Evaluación para Métodos de Desarrollo de AR-SOA, Adaptado de [19]

Criterios	Elementos	Preguntas
Publicación del servicio	Directamente a los consumidores	¿La AR-SOA permite la publicación de la descripción de los servicios?
	A través de intermediarios	
Composición de los servicios	Orquestación de servicios	¿La AR-SOA permite que el desarrollo de software pueda ser construido por medio de la composición de servicios?
	Coreografía de servicios	
Calidad del Servicio	Cumplimiento de los requisitos no funcionales de alto nivel	¿La AR-SOA permite el uso de mecanismos para validar que se cumplan los requisitos no funcionales de alto nivel requeridos por la funcionalidad que un servicio desempeña?
	Cumplimiento de los requisitos no funcionales de bajo nivel (QoS)	¿La AR-SOA permite el uso de mecanismos para validar que se satisfagan los requisitos no funcionales de bajo nivel (QoS) descritos en los acuerdos de su interfaz?

IV. Revisión Documental Sistemática (RDS)

Kitchenham [16] indica que la gestión del proceso de revisión sistemática se fundamenta en tres fases principales: Planificación, Realización e Informe de la revisión. En la Tabla IV se describen las fases, las etapas de cada fase y los artefactos generados en el proceso RDS.

Tabla IV: Proceso de Revisión Documental Sistemática

Fases	Etapas	Artefactos
Planificación de la revisión	Identificación de la necesidad de una revisión.	Protocolo
	Especificación de la(s) interrogante(s) de investigación.	
	Desarrollo del protocolo de revisión.	
Realización de la revisión	Identificación de estudios relacionados.	Estudios aceptados
	Selección de estudios primarios.	
	Evaluación de la calidad de los estudios.	
	Extracción de los datos.	Modelo de datos (checklist)
	Síntesis de los datos.	
Informe de la revisión	Especificación de mecanismos de difusión.	Framework de evaluación
	Presentación de resultados	

A. Planificación de la Revisión

Esta fase define los objetivos de la investigación y el protocolo en que la revisión se ejecutará.

1) Identificación de la Necesidad: Revisar los procesos de desarrollo actuales con el fin de proporcionar una guia para el diseño de una AR para LPSOS, en nuestro caso. Ademas, de identificar problemas pendientes por resolver y posibles áreas para la mejora de los procesos de desarrollo.

2) Desarrollo del Protocolo: El protocolo es el plan o conjunto de pasos a seguir en un estudio, constituido por las preguntas de investigación, las estrategias de búsqueda, los criterios y procesos de selección, los criterios de evaluación de la calidad, el modelo y proceso de extracción de datos, y el plan de análisis y sintesis de los datos.

Las siguientes *Preguntas de Investigación (PI)* fueron consideradas en inglés; aquí las colocamos en castellano:

- PI1. ¿Cuáles métodos que siguen enfoques “bottom-up” y/o “top-down” existen (estado del arte) para pasar de (conversión) una arquitectura basada en componentes a una arquitectura basada en WS o como llevar a cabo esta correspondencia?

- PI2. ¿cómo se expresan los requisitos de calidad de las funcionalidades respecto a los QoS de los WS?

Con relación a la estrategia de búsqueda, las fuentes de búsqueda seleccionadas para encontrar los estudios primarios son los siguientes: Google Academic y ResearchGate.

Se identificaron las palabras claves de búsqueda (descriptores): Software Product Line, Service-Oriented Architecture, Methods and Tools Design, Reference Architecture, Web Services, Business Process, Non-Functional Properties, Quality of Service; combinándolas con operadores lógicos AND, OR; para capturar los estudios primarios.

Los *Criterios de Inclusión (CI)* utilizados para incluir los estudios relevantes en esta RDS son:

- CI1: Estudios publicados entre 2008 y 2015
- CI2: Que aborde la arquitectura para el desarrollo de Líneas de Productos de Software para Familias de Productos Orientados a Servicios utilizando SOA.
- CI3: Que en su contenido incluya al menos el ciclo de vida de la ID para LPSOS.
- CI4: Que en su contenido describa un método de desarrollo para LPSOS.

Los *Criterios de Exclusión (CE)* son:

- CE1: Que no aborde la integración de LPS y SOA como alternativa para el desarrollo de productos de software.
- CE2: Que esté escrito en un lenguaje diferente al inglés o castellano y que no esté disponible en formato PDF; o que tenga como contenido solo una presentación en un congreso o workshop realizado por los autores.
- CE3: Si existen trabajos repetidos, se elige uno de ellos.

En la selección inicial de las publicaciones entre los resultados de búsqueda, se considera la lectura de los títulos, palabras claves y resúmenes de las publicaciones encontradas. Las publicaciones serán almacenadas en carpetas independientes de acuerdo a la expresión de búsqueda utilizada, y de esta forma facilitar la ubicación de los estudios para realizar el análisis con mayor fluidez y comodidad.

Con respecto a la extracción de datos y método de síntesis, se prevé la construcción de tablas para la obtención de datos relacionados con las preguntas de investigación.

Para estandarizar la forma en que la información estará representada, se definió un formato (modelo de datos) para recopilación de los datos en los estudios seleccionados. Las Tablas VII, VIII, IX muestran la descripción de las propiedades de extracción o "checklist" que serán utilizadas a lo largo de la RDS. Las Tablas X y XI muestran los hallazgos encontrados de la extracción de los datos relevantes en los estudios.

El modelo de datos será utilizado en la fase de Realización de la revisión, para seleccionar los mejores estudios, y luego en la fase de Informe de la revisión, estos estudios serán analizados a través del MUE que se describe en la Sección III.

B. Realización de la Revisión

En esta fase se sigue el protocolo previamente establecido. Como resultado de la búsqueda, 331 estudios fueron localizados, de los cuales solo 21 fueron aceptados inicialmente para su análisis a través de tres "checklists", las Tablas VII, VIII y IX respectivamente. La Tabla V a continuación, muestra un resumen de los resultados indicando la cantidad de estudios localizados, aceptados y rechazados.

Tabla V: Relación de Estudios Localizados y Aceptados según la Fuente de Datos

Fuentes de datos	**Artículos**		
	Aceptados	***Rechazados***	***Localizados***
Google Academic	12	283	295
ResearchGate	9	27	36
Total	**21**	309	331

De los estudios localizados se descartaron aquellos artículos cuyos contenidos eran idénticos, aun cuando sus identificadores eran diferentes. También, se identificaron estudios similares, que abordaban el mismo proceso y en los cuales participaban los mismos autores, aunque el titulo del estudio era distinto; para estos casos, se tomó la decisión de elegir el estudio más completo y actualizado. La Tabla VI muestra los estudios primarios aceptados según el tipo de evento y año de publicación.

Tabla VI: Relación de Estudios Primarios Relativos a Procesos de Desarrollo para LPSOS según Tipo y Año de Publicación

Año	**Tipo de evento**			**Total**
	Conferencia	***Revista***	***Tesis***	
2008	2	1		3
2009	2			2
2010	2	2	1	5
2011	5	1		6
2012	1			1
2013			1	1
2014	1	1		2
2015		1		1
Total	**13**	**6**	**2**	**21**

Los 21 estudios primarios describen un proceso para el desarrollo de una LPSOS. A continuación, un análisis más detallado se llevó a cabo en cada uno de ellos. Este análisis se realizó mediante la aplicación de dos "checklists" (Tablas X y XI) respecto a las propiedades que nos interesan evidenciar en los estudios (Tablas VII, VIII y IX). Esto permite caracterizarlos con relación a las propiedades presentes en sus respectivos contenidos. Se utiliza la nomenclatura (++), (+), (-), que significa que la calidad de la documentación fue buena, regular con un valor en ambos casos de 1 punto c/u o que carece de esa propiedad con un valor de 0 punto, respectivamente. Este aspecto también influyó en la decisión sobre cuales estudios fueron seleccionados para ser evaluados por el MUE.

Tabla VII: Descripción de las Propiedades: Métodos para AR-LPS

Propiedad	Descripción
ID	**Ciclo de Vida de Ingeniería del Dominio (ID)**
An	Fase de Análisis
Di	Fase de Diseño
Re	Fase de Realización
IA	**Ciclo de Vida de Ingeniería de la Aplicación (IA)**
Estrategia Desarrollo	**Estrategia utilizada para abordar el método [28]:**
Td	Top-down (proactivo)
Bu	Bottom-up (extractivo/reactivo)
Hi	Middle out o Meet-in-the-middle (híbrido)
Método	**El método describe un proceso sistemático:**
Act	Actividades a seguir
E/S	Entradas y salidas
Art	Artefactos producidos/utilizados
Rol	Roles de participación en el proceso
Arq	Vistas arquitecturales
CasoEst	Caso de estudio que valida el proceso
ModVariab	**Modelado de la variabilidad funcional y no funcional (calidad) [29]**
Car	En el modelo de características [46]
Arq	En el modelo arquitectural
PN	En el modelo de procesos de negocio
WS	En servicios: un servicio proporciona variantes mediante la parametrización; o varios servicios ofrecen la misma funcionalidad, pero con diferentes QoS (implementaciones)
Estd	Uso de un Modelo de Calidad estándar para especificar las propiedades de calidad [30]: ISO-9126, ISO-25010, McCall, Boehm, FUR PS, Dromey, etc.
EspArq	**El método considera la especificación de la AR**
Tech	**Enfoques y Tecnologías:** SCA[6]; MDA[7]; MDE[8]; SOC[9]
Ont	**Ontologías:** expresan el conocimiento del dominio; expresan la AR como medio para relacionar el dominio del problema con el dominio de la solución con una terminología común.
Herr	**Herramientas:** grado en que el método es automatizado; si una herramienta es propuesta o implementada.

Tabla VIII: Descripción de las Propiedades: Métodos para SOA

Propiedad	Descripción
GPN	**incluye el ciclo de vida de GPN [20]:**
NotacPN	Notación (BPMN, etc.) para el modelado de los procesos de negocios
LengEj	Utiliza el lenguaje BPEL para implementar (ejecutar) los procesos de negocios en la etapa de implementación del ciclo de vida de GPN
EstArq	**El estilo (modelo) arquitectural es SOA**
WS	**WS como tecnología para realizar SOA**
Métodos de conversión	**Método para identificar, especificar y realizar los servicios bajo un enfoque SOA**
Car/S	Actividades de como traducir del Modelo de Características a Servicios (“top-down”)
PN/S	Actividades de como traducir de un Modelo de Procesos de Negocios a Servicios (“top-down”)
WS/Arq	Actividades de como traducir los Servicios a Componentes arquitecturales (“top-down”)
Arq/WS	Actividades de como traducir una AR de software (componentes, conectores) a una AR cuyo componentes son Servicios (“bottom-up”)
Métodos de desarrollo	**Métodos de desarrollo SOA para las fases de Análisis y Diseño [28]:** IBM RUP/SOMA; SOAF; SOUP; SOMA; Erl methodology; Papazoglou methodology
ModRef/ ModConcept	**Modelos de Referencia/Modelos conceptuales SOA [31]:** SOA-RM[10]; SOA-RFA[11]; SOA Ontology[12]; SOMF[13]; PIM4SOA[14]; SoaMl[15]

[6] Service Component Architecture
[7] Model Driven Architecture
[8] Model Driven Engineering
[9] Service Oriented Computing
[10] OASIS Reference Model for SOA
[11] OASIS Architecture Foundation for SOA
[12] Open Group SOA Ontology
[13] Service-oriented Modeling Framework
[14] Platform-independent Model for SOA
[15] SOA Modeling Language

Tabla IX: Descripción de las Propiedades: Métodos para AR-SOA

Propiedad	Descripción
Pub	Permite la publicación de la descripción de los servicios directamente a los consumidores, a través de intermediarios
Comp	Permite que el software pueda ser construido por medio de la composición de los servicios
RNF	Cumplimiento de los requisitos no funcionales de alto nivel y/o de bajo nivel

Tabla X: Checklist de las Propiedades para Métodos AR-LPS Relevantes Identificadas en los Estudios Primarios sobre LPSOS

Estudios Primarios	AR-LPS																					Total
	ID			*IA*	*Estrategia Desarrollo*			*Método*						*ModVariab*				*Estd*	*Esp Arq*	*Tech*	*Ont*	*Herr*	
	An	Di	Re		Td	Bu	Hi	Act	E/S	Art	Rol	Arq	Caso Est	Car	Arq	PN	WS						
ZGCAL [32]	++	++	++	-	++	-	-	++	++	++	-	++	++	++	-	-	-	-	-	-	++	**11**
GB [24]	++	+	+	-	-	++	-	++	++	-	-	-	++	++	-	-	-	-	-	-	-	**8**
Istoan [2]	++	++	++	++	++	-	-	++	++	++	-	-	++	++	-	++	-	-	-	MDE[16]	-	**12**
RMALAAG [29]	+	-	++	-	++	-	-	++	++	++	-	-	-	-	-	-	++	-	-	-	-	**7**
CK [13]	++	-	-	-	-	++	-	++	-	-	-	-	++	-	-	-	++	-	-	-	-	**5**
GE [33]	++	-	-	-	++	-	-	++	-	-	-	-	-	-	-	-	-	-	-	-	-	**3**
BGFF [34]	++	-	-	-	++	-	-	++	-	-	-	-	-	-	-	-	-	-	-	-	-	**3**
EMA [35]	++	+	-	-	++	-	-	++	++	++	-	++	++	++	-	-	-	-	++	-	-	**10**
Med [3]	++	++	++	-	++	-	-	++	++	++	++	++	++	-	++	-	-	-	++	-	-	**12**
BMKARGBH [36]	++	++	++	++	++	-	-	++	++	++	++	-	++	++	-	-	-	-	-	-	++	**13**
DRHU [37]	++	-	-	-	++	-	-	++	-	-	-	-	-	-	-	-	-	-	-	-	-	**3**
PJ [38]	++	++	++	++	-	++	-	++	++	++	++	++	++	++	-	-	-	-	-	-	++	**13**
SS [39]	++	-	-	-	++	-	-	++	-	-	-	-	-	-	-	-	-	-	-	-	-	**3**
AG [40]	++	++	++	++	++	-	-	++	++	++	++	++	++	++	-	++	++	-	-	-	-	**14**
BCCMV [41]	++	-	-	-	++	-	-	++	-	-	-	-	-	-	-	-	-	-	-	-	-	**3**
MAGL [6]	++	++	++	++	++	-	-	++	++	++	-	++	++	++	-	++	++	-	++	-	-	**14**
LMN [42]	++	++	++	++	++	-	-	++	++	++	-	++	++	++	-	++	-	-	-	-	-	**12**
Galster [43]	++	+	-	-	-	++	-	++	++	++	++	-	++	-	-	-	-	-	-	-	-	**8**
AMKGBH [20]	++	++	++	++	++	-	-	++	++	++	++	++	++	++	-	++	++	-	++	MDE	-	**17**
NBG [44]	++	-	-	-	++	-	-	++	-	-	-	-	-	-	-	-	-	-	-	-	-	**3**
ALHG [45]	++	++	++	++	-	++	-	++	++	++	-	++	++	++	++	-	++	-	++	-	-	**14**
21	**21**	**13**	**12**	**8**	**16**	**5**	**0**	**21**	**14**	**13**	**6**	**9**	**14**	**11**	**2**	**5**	**6**	**0**	**5**	**2**	**1**	**4**	

Tabla XI: Checklist de las Propiedades para Métodos bajo SOA y AR-SOA Relevantes Identificadas en los Estudios Primarios sobre LPSOS

Estudios Primarios	SOA										AR-SOA			Total
	GPN		*EstArq*	*WS*	*Métodos de conversión a Servicios*				*Métodos de Desarrollo*	*ModRef/ModConcept*	*Pub*	*Comp*	*RNF*	
	NotacPN	LengEj			Car/S	PN/S	WS/Arq	Arq/WS						
ZGCAL [32]	++	++	++	++	++	++	++	-	-	-	-	++	-	**8**
GB [24]	-	-	++	-	++	-	-	-	-	-	-	-	-	**2**
Istoan [2]	++	-	++	-	++	++	-	-	-	-	-	++	-	**5**
RMALAAG [29]	-	-	++	-	-	-	-	-	-	-	-	-	-	**1**
CK [13]	-	-	++	-	-	-	-	-	-	-	-	-	-	**1**
GE [33]	-	-	++	-	-	-	-	-	-	-	-	-	-	**1**
BGFF [34]	-	-	++	-	-	-	-	-	-	-	-	-	-	**1**
EMA [35]	++	-	++	-	-	-	-	-	-	-	-	-	-	**2**
Med [3]	-	-	++	++	++	-	-	-	RUP/SOMA	-	+	++	++	**7**
BMKARGBH [36]	-	-	++	++	++	-	-	-	-	-	-	++	-	**4**
DRHU [37]	-	-	++	-	-	-	-	-	-	-	-	-	-	**1**
PJ [38]	-	-	++	++	++	-	++	-	-	-	-	++	-	**5**
SS [39]	-	-	++	-	-	-	-	-	-	-	-	-	-	**1**
AG [40]	++	-	++	-	++	++	-	-	-	SoaML[17]	-	++	-	**6**
BCCMV [41]	-	-	++	-	-	-	-	-	-	-	-	-	-	**1**
MAGL [6]	++	-	++	-	-	++	-	-	-	-	-	++	++	**5**
LMN [42]	++	-	++	-	++	++	++	-	-	-	-	++	-	**6**
Galster [43]	-	-	++	-	-	-	-	-	-	-	-	-	-	**1**
AMKGBH [20]	++	-	++	-	++	++	-	-	-	-	-	++	-	**5**
NBG [44]	-	-	++	-	-	-	-	-	-	-	-	-	-	**1**
ALHG [45]	-	-	++	++	++	++	++	-	-	-	-	++	-	**6**
21	**7**	**1**	**21**	**5**	**10**	**7**	**4**	**0**	**1**	**1**	**1**	**10**	**2**	

[16] Model Driven Engineering
[17] http://www.omg.org/spec/SoaML

Luego de la captura de los datos, reflejados a través de las propiedades identificadas en las Tablas X, XI, aquellos estudios que obtuvieron el mayor número de propiedades consideradas, fueron seleccionados para ser evaluados utilizando las categorías, elementos y preguntas del MUE descrito en la sección III (Tablas I, II, III).

V. DISCUSIÓN

A continuación se presenta el informe de los resultados analíticos de la RDS, considerando cada pregunta de investigación y analizados a través de la aplicación del MUE, expresado a través de sus criterios y elementos de análisis. La Tabla XII identifica los 10 estudios más representativos relacionados con procesos de desarrollo de LPSOS, de los 21 estudios primarios inicialmente aceptados y que luego de ser analizados, son los que satisfacen el mayor número de propiedades encontradas en los métodos, este valor es obtenido de la sumatoria de los totales de las Tablas X, XI. MUE se aplicó a estos 10 estudios, que tratan explícitamente procesos de desarrollo para LPSOS.

Tabla XII: Estudios más Representativos entre los Primarios para LPSOS

Estudios	Propiedades
ZGCAL [32]	19
Istoan [2]	17
Med [3]	19
BMKARGBH [36]	17
PJ [38]	18
AG [40]	20
MAGL [6]	19
LMN [42]	18
AMKGBH [20]	22
ALHG [45]	20

A. Con Respecto a Métodos para AR-LPS (ver Tabla I)

1) Criterio Dominio: Los estudios tratan los dominios: gobierno electrónico o "e-government", comercio electrónico (e-business), y reservaciones (e-travel).

2) Criterio Contexto del Método: objetivos, fases, entradas, salidas

Objetivo: Se destacan dos propuestas para el desarrollo de LPSOS [2][20] que utilizan el paradigma de la ingeniería dirigida por modelos, en inglés *Model Driven Engineering (MDE)*. La principal diferencia entre ambos, es que aplican el enfoque de forma diferente: [20] lo aplica para desarrollar el ciclo de vida de ID para LPS completo; y [2] lo utiliza en la etapa final del ciclo de vida de la IA, para la generación de los productos de la línea.

Fases: Todas las propuestas incluyen el ciclo de vida de la ID para LPS.

Entradas: Todas las propuestas parten de que el dominio del problema es conocido.

Salidas: Modelo de características [4][20][32][38][42][46], modelo de variabilidad [8], modelo de procesos de negocio [6][7][20][40][41], modelo de servicios [2], modelo de componentes [3][42] y AR [11][19][20][45].

3) Criterio Usuarios del Método: grupo, motivación habilidades, orientación

Grupo: La mayoría de las propuestas señalan como participantes o actores a ingenieros del dominio, arquitectos e ingenieros de software y analistas SOA. Los *roles más representativos* son: *Business Analyst (Analista de Negocio)* que es el responsable de verificar que los candidatos a orquestación de servicios representen de forma precisa la lógica del negocio; *Domain Architect (Arquitecto del Dominio)* para la identificación de los componentes que proveen la implementación de las operaciones expuestas por los servicios; *SOA Architect (Arquitecto SOA)* para la identificación de los servicios candidatos; *Domain Designer (Diseñador del Domino)* and *Service Designer (Diseñador de Servicios)*. Para [3] los más importantes son: Analista de Negocio, el Arquitecto SOA y el Arquitecto del Dominio.

Orientación: Todas las propuestas indican al usuario como realizar las actividades en cada una de las fases del método.

Motivación y habilidades: Ninguna de las propuestas describe las habilidades requeridas por los usuarios de los métodos para poder realizar de forma efectiva las actividades presentadas.

4) Criterio Componentes del Método: estructura, artefactos, vistas, lenguajes, variabilidad y herramientas

Estructura: En cuanto a la *definición de los pasos y su secuencia*, existe consenso en determinar el alcance de la LPS mediante el análisis de sus características expresado a través del modelo de características, esto es desde el punto de vista estructural, y desde el punto de vista de su comportamiento, se realiza un análisis de los procesos de negocios utilizando por lo general BPMN para expresarlos, ambos conceptos son muy utilizados en la especificación de la variabilidad en la LPSOS.

En [20], para lograr la integración LPS-MDE-SOA, se comienza por la definición de los procesos de negocio y por la especificación del conjunto de actividades coordinadas para el cumplimiento de los objetivos organizacionales. Luego, los elementos de software necesarios para lograr estos objetivos (los servicios) son identificados, y para cada servicio, se identifican y desarrollan el conjunto de interfaces realizando la especificación del servicio, así como las correspondientes implementaciones de servicio.

Artefactos: el modelo de características [46] es el más utilizado, seguido del modelo de procesos de negocios [6][40], el modelo de variabilidad [32][38][42], y la AR [45].

Variabilidad: [3] indica que la gestión de la variabilidad es una actividad clave en ILPSOS, en la cual los servicios proporcionan la flexibilidad de las LPS y de sus productos, diferenciándolos en el momento de establecer el enlace de los servicios: en tiempo de diseño (estáticamente) y en tiempo en ejecución (dinámicamente). La variabilidad en la mayoría de los estudios analizados se realiza en el modelado de características, en algunos estudios ésta es realizada a través de los procesos de negocios [20], y en un solo estudio [45] es realizada a nivel de los servicios y de los componentes, expresados mediante una AR.

También en [20] se indica que la mayoría de los enfoques de gestión de procesos de negocio enfatizan el desarrollo de un proceso de negocio, y a partir del mismo, se derivan muchas variantes, especializadas según diversas necesidades de la organización. Los procesos de negocio pueden tener algunas piezas comunes para un grupo de diferentes casos de aplicación, donde implica cierta variabilidad en la selección de las actividades de un proceso de negocio. Una funcionalidad común puede ser desarrollada como un activo reutilizable para la creación de nuevas variantes de los procesos de negocio en

un dominio. Además, los puntos de variación, derivados de las propiedades no funcionales y tecnológicas de las actividades, proporcionan la funcionalidad del negocio con diferentes propiedades no funcionales y tecnológicas. Tal variabilidad facilita la configuración flexible del grupo de procesos de negocio dirigida por la calidad [20].

Vistas: Con relación a las *vistas arquitecturales*, la especificación de la arquitectura es la actividad, en la cual la arquitectura es documentada utilizando diferentes vistas y notaciones para representar los intereses de los diferentes actores involucrados. Existen cuatro estudios [3][32][40][42] que proponen la utilización de vistas arquitecturales similar al enfoque del 4+1 vistas [25].

En [40] se especifican 5 vistas: Vista de Características, Vista de Contrato de Servicio, Vista de Procesos de Negocio, Vista de Interface de Servicios y la Vista de Coordinación de Servicios. En [3] las 4 vistas arquitecturales consideradas son: Vista de Capas; Vista de Integración; Vista de Componente y Vista de Interacción. En [32] se especifican 3 vistas arquitecturales: Vista de Coordinación, Vista de Aplicación, Vista de Datos. En [42] se propone un estilo arquitectural heterogéneo, con tres niveles de descomposición asociados a tres estilos arquitecturales representados mediante 3 vistas: Vista Arquitectural de Servicios, Vista Arquitectural de Comunicación entre Procesos, Vista Arquitectural (en el ADL[18] C2). Solo los estudios [3][42] consideran la Vista de Componentes, donde existe una correspondencia entre las interfaces de los servicios y los componentes que las implementan.

Lenguajes: Los lenguajes utilizados por el método para el modelado arquitectural, [3][32][40][42] utilizan UML para expresar algunas de las vistas arquitecturales, especialmente [45] la utiliza para especificar la AR; para modelar los procesos de negocios, en [20][36] utilizan BPMN y en [30][40] el diagrama de actividades se expresa en UML.

Herramientas: En cuanto a las herramientas utilizadas solo cuatro trabajos mencionan alguna [20][32][36][38], que abordan parcialmente alguna parte del método propuesto; en [20] se utiliza FeatureMapper[19] para especificar el modelo de características y relacionar las características con los artefactos de software que las implementan; en [36] utilizan un enfoque basado en ontologías, donde utilizan el modelo de características [46] para la generación automática de composiciones de servicios ejecutables, para ello usan un lenguaje semántico de WS y utilizan un framework que soporta la composición de WS a nivel semántico, denominado *Web Service Modelling Ontology (WSMO)*, donde las composiciones de servicios son codificadas en el lenguaje WSML, este framework tiene una herramienta que ejecuta estas composiciones, denomina *Web Service Modelling eXecution environment (WSMX)*[20], que permite generar y ejecutar la orquestación y coreografía de los servicios. También utilizan *ATLAS*[21] *Transformation Language (ATL)* para traducir del modelo de características al WSML. En [32] *Feature Plug-in*[22] y [38] *FeatureIDE*[23] son utilizadas para especificar el modelo de características.

5) Criterio Gestión de la Calidad en el Método: propiedades de calidad

Propiedades de calidad: La calidad en general es muy poco abordada en la LPSOS y tratada tarde en la etapa de diseño, solo dos estudios [3][6] trataron este aspecto de forma más precisa. En [6] se indica que en la fase de diseño del dominio se debe incluir la especificación de las propiedades no funcionales, debido a que los NFRs están entrelazados y relacionados con los requisitos funcionales, y que puede haber varios servicios que proporcionan la misma funcionalidad, pero que difieren de la implementación de su QoS. Para ello, las características en el modelo de características tipo FODA [46] son etiquetadas con los rangos de calidad que serán soportados por la arquitectura de la línea de productos, ayudando al ingeniero de servicios y de software a evaluar el impacto de las características variantes seleccionadas acorde a las características de calidad que los servicios proveen. Por su parte [3] presenta una técnica para identificar servicios a partir de los atributos de calidad, analizando escenarios de atributos de calidad para identificar los servicios que ayuden al cumplimiento de los atributos de calidad arquitectural, para ello, dispone de un conjunto de patrones de diseño SOA que permiten satisfacer algunos atributos de calidad. Ninguna propuesta utiliza un estándar para especificar la calidad.

6) Criterio Validación del Método: madurez, calidad de la arquitectura

Madurez: Todas las propuestas analizadas presentaban un caso de estudio, donde se evaluaba su factibilidad.

Calidad de la arquitectura: Solo dos propuestas evalúan la calidad arquitectural: [6] lo hace desde el punto de vista de los usuarios finales del método, mediante la evaluación del desempeño y escalabilidad del método; [3] presenta métricas para medir la cohesión y el acoplamiento entre los componentes a nivel arquitectural.

B. Con Respecto a Métodos para SOA (ver Tabla II)

1) Criterio Concepto de Servicio: negocio, WS

En términos generales la mayoría de los métodos abordan ambas definiciones: servicios de negocios y WS, primero identifican cuales son los servicios necesarios para dar soporte a las funcionalidades del negocio y luego identifican cuales WS son requeridos.

2) Criterio Estrategia de realización o desarrollo: top-down, bottom-up, middle-out

"Top-down" fue la más utilizada, con un 76,2% del total de estudios analizados, seguido del "bottom-up" con un 23,8%; en ninguno de los métodos se abordó un enfoque hibrido.

3) Criterio Cobertura del ciclo de vida: Identificación, Especialización, Realización

Solo un estudio [45] abarcó en su análisis el ciclo de vida completo SOA, el resto de los estudios solo se enfocan en la presentación de propuestas para la fase de identificación de los servicios.

[18] Architecture Description Language
[19] http://featuremapper.org
[20] http://www.wsmx.org
[21] https://eclipse.org/atl
[22] http://gsd.uwaterloo.ca/fmp
[23] http://wwwiti.cs.uni-magdeburg.de/iti_db/research/featureide

4) Criterio Grado de granularidad del análisis: Bajo, Moderado, Alto

El nivel de abstracción en general es alto, la identificación de los servicios es la actividad más realizada en los estudios [3][20][36][38][40], la especificación es muy poco abordada y la realización casi inexistente. Sin embargo, dos estudios ameritan mencionarse: en [20], en el paradigma SOA, las actividades en un grupo de procesos de negocio pueden ser implementadas por servicios o delegadas a los usuarios. Cada actividad, puede tener uno o más servicios que pueden realizarla. Los servicios pueden ser comunes y reutilizados entre diversos procesos de negocios. Diferentes servicios pueden proporcionar funcionalidad del negocio para soportar distintos requisitos tecnológicos, como la comunicación a través de diversos medios de transporte. Además, pueden proporcionar funcionalidad del negocio con diferentes propiedades no funcionales, como seguridad y rendimiento, expuestos por diferentes implementaciones de interfaces de servicios y composición de servicios. En [45], la variabilidad de una LPS es realizada a nivel de los servicios y componentes, expresados en una AR. Ninguno de los trabajo revisados aborda tener una AR y luego adaptarla a servicios.

5) Criterio Accesibilidad y validez: Documentación no accesible, parciamente accessible, bien accessible

Todos los métodos analizados presentan una buena documentación y presentan un caso de estudio, donde se detallaba las actividades a seguir y se evaluaba su validez.

C. Con Respecto a Métodos para AR-SOA (ver Tabla III)

1) Criterio Publicación del servicio: directamente a consumidores, por intermediarios

Este criterio no fue abordado en los métodos analizados.

2) Criterio Composición de los servicios: orquestación, coreografía

La mayoría de las propuestas presentan como alternativa para la composición de los servicios la orquestación, implementada a través de los procesos de negocios, quienes coordinan el llamado a los servicios.

3) Criterio Calidad del Servicio: cumplimiento de requisitos no funcionales de alto nivel y de bajo nivel

Ninguno de los métodos evalúa la calidad de servicio a un alto nivel de abstracción en las arquitecturas presentadas, como por ejemplo desempeño, escalabilidad, interoperabilidad.

VI. Conclusiones

La principal contribución de este trabajo fue realizar una RDS para presentar una perspectiva detallada sobre las fases, actividades, artefactos y roles que participan en un proceso de desarrollo para LPSOS; en la literatura no se encontraron RDS similares. Aunque los beneficios de la orientación a servicios son frecuentes en la literatura, la revisión, análisis y evaluación de los 21 métodos presentados en esta RDS ha demostrado que falta un enfoque integral que incluya en las fases de un ciclo de vida ID de LPSOS, la consideración de las propiedades de calidad, para la identificación, especificación y realización de los servicios como componentes arquitecturales representados mediante una AROS-LPS, que muestre la clara trazabilidad entre los componentes de servicios funcionales y no funcionales. La gestión de calidad en los métodos fue muy poco abordado, y ningún estándar para especificar las propiedades de calidad fue considerado. Por su parte, la gestión de la variabilidad es clave en un contexto: SOA y LPS. En ambos casos, los puntos de variación deben ser implementados ya sea por un único servicio (donde una interfaz de servicios puede ofrecer parametrización) o a través de servicios similares que traten cada variación; estas variaciones pueden estar influenciadas por las tecnologías seleccionadas para satisfacer propiedades de calidad requeridas para alcanzar los objetivos empresariales, representados como componentes de servicios que resuelven funcionalidades. Los métodos que aplican la gestión de procesos de negocio enfatizan el desarrollo de un proceso de negocio, y a partir de él, se derivan las variantes, especializadas según las necesidades de la organización. Los procesos de negocio pueden tener algunas partes comunes para un grupo de diferentes casos de aplicación, donde implica cierta variabilidad en la selección de las actividades de un proceso de negocio. La RDS presentada provee a la comunidad de desarrolladores de LPSOS la visión general de métodos actuales para tomar decisiones en cuanto a la definición de un método general para el diseño de una AROS-LPS. La principal limitación del estudio realizado es que la RDS no es una comparación para determinar quién de los métodos de desarrollo es el mejor, la intención es de describir como los autores abordan el proceso de desarrollo de una AROS-LPS. Una perspectiva a ser considerada sería evaluar el alcance de la AROS-LPS respecto a la calidad y número de productos que se pueden derivar a partir de ella, para una familia de LPSOS determinada.

Agradecimientos

Al CDCH-UCV (Consejo de Desarrollo Científico y Humanístico), proyecto DARGRAF PG 03-8730-2013-2.

Referencias

[1] P. Istoan, G. Nain, G. Perrouin, and J. Jezequel, *Dynamic Software Product Lines for Service-Based Systems*, 9th IEEE International Conference on Computer and Information Technology (CIT´09), Xiamen, China, October 2009.

[2] P. Istoan, J. M. Jézéquel, and G. Perrouin, *Software Product Lines for Creating Service-Oriented Applications*, Ph.D. thesis, Irisa Rennes Research Institute, Rennes, France, June 2009.

[3] F. M. Medeiros, *SOPLE-DE: an Approach to Design Service-Oriented Product Line Architectures*, Master. thesis, Programa de Pós-Graduação em Ciência da Computação, Universidade Federal de Pernambuco, Recife, Brasil, 2010.

[4] S. Apel, D. Batory, C. Kästner, and G. Saake, *Feature-Oriented Software Product Lines*, Springer, 2013.

[5] Q. Munir and M. Shahid, *Software Product Line: Survey of Tools*, Master. thesis, Department of Computer and Information Science, Linköping University, Linköping, Sweden, 2010.

[6] B. Mohabbati, M. Asadi, D. Gašević, and J. Lee, *Software Product Line Engineering to Develop Variant-Rich Web Services*, In Web Services Foundations, Springer, New York, 2014.

[7] R. Dos Santos and M. Fantinato, *The Use of Software Product Lines for Business Process Management: A Systematic Literature Review*, Information and Software Technology vol. 55, no. 8, pp. 1355-1373, 2013.

[8] K. Pohl, G. Bockle, and F. van der Linden, *Software Product Line Engineering: Foundations, Principles, and Techniques*, Springer, 2005.

[9] World Wide Web Consortium, *Web Services Architecture Requirements*, W3C Working Group Note, February 2004.

[10] J. Nicolai, *SOA in Practice: the Art of Distributed System Design*, OReilly: Sebastopol, CA, 2007.

[11] M. Galster, P. Avgeriou, and D. Tofan, *Constraints for the Design of Variability-Intensive Service-Oriented Reference Architectures – An*

Industrial Case Study, Information and Software Technology, vol. 55, no. 2, pp. 428-441, 2013.

[12] M. Rosen, B. Lublinsky, K. Smith, and M. Balcer, *Applied SOA: Service-Oriented Architecture and Design Strategies*, Wiley & Sons, 2008.

[13] S. Cohen and R. Krut, *Managing Variation in Services in a Software Product Line Context* (No. CMU/SEI-2010-TN-007), Carnegie-Mellon University, Software Engineering Institute, Pittsburgh, PA, USA, 2010.

[14] G. Kotonya, J. Lee, and D. Robinson, *A Consumer-Centred Approach for Service-Oriented Product Line Development*, In proceedings of Working IEEE/IFIP Conference on Software Architecture & European Conference on Software Architecture (WICSA/ECSA 2009), pp. 211-220, Cambridge, UK, September 2009.

[15] M. Shaw and D. Garlan, *Software Architecture: Perspectives on an Emerging Discipline*, Prentice Hall, 1996.

[16] B. Kitchenham, *Guidelines for Performing Systematic Literature Reviews in Software Engineering, Version 2.3*, EBSE Technical Report, Software Engineering Group, School of Computer Science and Mathematics, Keele University, UK and Department of Computer Science, University of Durham, UK, 2007.

[17] M. Matinlassi, *Comparison of Software Product Line Architecture Design Methods: COPA, FAST, FORM, KobrA and QADA*, In Proceedings of the 26th International Conference on Software Engineering (ICSE'04), pp. 127-136, Edinburgh, UK, May 2004.

[18] T. Kohlborn, A. Korthaus, T. Chan, and M. Rosemann, *Identification and Analysis of Business and Software Services - a Consolidated Approach*, Services Computing, IEEE Transactions, vol. 2, no. 1, pp. 50-64, 2009.

[19] L. B. R. de Oliveira, K. R. Felizardo, D. Feitosa, and E. Y. Nakagawa, *Reference Models and Reference Architectures Based on Service-Oriented Architecture: a Systematic Review*, In Software Architecture, pp. 360-367, Springer Berlin Heidelberg, 2010.

[20] M. Asadi, B. Mohabbati, N. Kaviani, D. Gaševic, M. Boškovic, and M. Hatala, *Model-driven Development of Families of Service-Oriented Architectures*, In Proceedings the 1th International Workshop on Feature-Oriented Software Development (FOSD´09), pp. 95-102, Denver, Colorado, USA, October 2009.

[21] Object Management Group (OMG). *Business Process Model and Notation (BPMN), Version 2.0*, 2011.

[22] B. List and B. Korherr, *An Evaluation of Conceptual Business Process Modelling Languages*, In Proceedings of the 2006 ACM symposium on Applied computing (SAC'06), pp. 1532-1539, Dijon, France, April 2006.

[23] J. Herrera, F. Losavio, and A. Matteo, *RDS de Enfoques y Técnicas para la Construcción de Arquitecturas en un Contexto de Líneas de Productos de Software*, Revista Venezolana de Ciencias de la Computación ReVeCom, vol. 1, no. 1, pp. 17-25, Junio 2014.

[24] S. Günther and T. Berger, *Service-Oriented Product Lines: Towards a Development Process and Feature Management Model for Web Services*, In SPLC, vol. 2, pp. 131-136, October 2008.

[25] P. Kruchten, *Architectural blueprints—The "4+1" View Model of Software Architecture*, In proceedings of the Conference on TRI-Ada'95, Anaheim, CA, USA, November 1995.

[26] H. J. González, *Integration of Quality Attributes in Software Product Line Development*, Tesina de Máster en Ingeniería del Software, Métodos Formales y Sistemas de Información (ISMFSI), 2012.

[27] A. Arsanjani, *Service-Oriented Modeling and Architecture: How to Identify, Specify, and Realize Services for your SOA*, IBM Software Group, November 2004.

[28] S. Svanidzaitė, *A Comparison of SOA Methodologies Analysis & Design Phases*, In Tenth International Baltic Conference on Databases and Information Systems (Baltic DB & IS 2012), Vilnius, Lithuania, July 2012.

[29] H. G. B. Ribeiro, S. R. de Lemos Meira, E. S. de Almeida, D. Lucredio, A. Alvaro, V. Alves, and V. Garcia, *An Assessment on Technologies for Implementing Core Assets in Service-Oriented Product Lines*, In 4th Brazilian Symposium on Software Components, Architectures and Reuse (SBCARS´10), pp. 90-99, Bahia, Brazil, September 2010.

[30] J. P. Miguel, D. Mauricio, and G. Rodríguez, *A Review of Software Quality Models for the Evaluation of Software Products*, International Journal of Software Engineering & Applications (IJSEA), vol. 5, no. 6, November 2014.

[31] M. Mohammadi and M. Mukhtar, *A Review of SOA Modeling Approaches for Enterprise Information Systems*, Procedia Technology, vol. 11, pp. 794-800, 2013.

[32] F. Zaupa, I. M. de Souza Gimenes, D. D. Cowan, P. S. Alencar, and C. J. P. de Lucena, *A Service-Oriented Process to Develop Web Applications*, Journal UCS, vol. 14, no. 8, pp. 1368-1387, 2008.

[33] M. Galster and A. Eberlein, *Identifying Potential Core Assets in Service-Based Systems to Support the Transition to Service-Oriented Product Lines*, In 18th IEEE International Conference and Workshops on Engineering of Computer-Based SystemsIn Engineering of Computer Based Systems (ECBS), pp. 179-186, Las Vegas, Nevada, USA, April 2011.

[34] M. H. ter Beek, S. Gnesi, A. Fantechi, and J. L. Fiadeiro, *Variability and Rigour in Service Computing Engineering*, In 35th Annual IEEE Software Engineering Workshop (SEW), pp. 122-127, Limerick, Ireland, June 2011.

[35] A. Ezenwoke, S. Misra, and M. O. Adigun, *An Approach for E-Commerce On-Demand Service-Oriented Product Line Development*, Acta Polytechnica Hungarica, vol. 10, no. 2, pp. 69-87, 2013.

[36] M. Bošković, D. Gašević, B. Mohabbati, M. Asadi, M. Hatala, N. Kaviani, J. Rusk, and E. Bagheri, *Developing Families of Software Services: a Semantic Web Approach*, Journal of Research & Practice in Information Technology, vol. 43, no. 3, 2011.

[37] N. C. Das, S. Ripon, O. Hossain, and M. S. Uddin, *Requirement Analysis of Product Line Based Semantic Web Services*, Lecture Notes on Software Engineering, vol. 2, no. 3, p. 210, 2014.

[38] C. Parra and D. Joya, *SPLIT: an Automated Approach for Enterprise Product Line Adoption Through SOA*, Journal of Internet Services and Information Security (JISIS), vol. 5, no. 1, pp. 29-52, 2015.

[39] H. Serajzadeh and F. Shams, *An Approach for Discovering Services for a Service-Oriented Product Line*, Global Journal on Technology, vol. 1, 2012.

[40] M. Abu-Matar and H. Gomaa, *Feature Based Variability for Service Oriented Architectures*, In WICSA'11 Proceedings of the 2011 Ninth Working IEEE/IFIP Conference on Software Architecture, pp. 302-309, Boulder, Colorado, USA, June 2011.

[41] N. Boffoli, D. Caivano, D. Castelluccia, F. M. Maggi, and G. Visaggio, *Business Process Lines to Develop Service-Oriented Architectures Through the Software Product Lines Paradigm*, In SPLC, vol. 2, pp. 143-147, 2008.

[42] J. Lee, D. Muthig, and M. Naab, *A Feature-Oriented Approach for Developing Reusable Product Line Assets of Service-Based Systems*, Journal of Systems and Software, vol. 83, no. 7, pp. 1123-1136, 2010.

[43] M. Galster, *Describing Variability in Service-Oriented Software Product Lines*, In Proceedings of the 4th European Conference on Software Architecture (ECSA'10), pp. 344-350, Copenhagen, Denmark, August 2010.

[44] M. Njima, M. H. Ter Beek, and S. Gnesi, *Product Line Architectures for SOA*, In Proceedings of the 11th Conference on Software Engineering Research and Practice (SERP´11), Las Vegas, Nevada, USA, July 2011.

[45] I. Achour, L. Labed, R. Helali, and H. B. Ghazela, *A Service Oriented Product Line Architecture for E-Government*, The 2011 International Conference on e-Learning, e-Business, Enterprise Information Systems, and e-Government (EEE'11), Las Vegas, Nevada, USA, July 2011.

[46] K. Lee, K. Kang, and J. Lee, *Concepts and Guidelines of Feature Modeling for Product Line Software Engineering*, Lecture Notes in Computer Science, vol. 2319, pp. 62-77, 2002.