Evaluación de una Arquitectura de Software

Evaluation of a Software Architecture

Óscar Agudelo Varela[1], Fernando Riveros-Sanabria[2], Santiago Valbuena Rodríguez[3]

[1] MSc, Grupo de Investigación Horizonte Mediático, * Universidad de los Llanos, Villavicencio, Colombia.
ORCID: https://orcid.org/0000-0002-1725-0490. Email: oscar.agudelo@unillanos.edu.co
[2] MSc, Grupo de Investigación MACRYPT, * Universidad de los Llanos, Villavicencio, Colombia.
ORCID: https://orcid.org/0000-0001-8948-6956. Email:friveros@unillanos.edu.co
[3] MSc, Grupo de Investigación Horizonte Mediático, * Universidad de los Llanos, Villavicencio, Colombia.
ORCID:https://orcid.org/0000-0002-9034-6640. Email:svalbuena@unillanos.edu.co

Recibido: 02/02/2021
Aceptado: 19/05/2021

Cite this article as: O. Agudelo, F. Riveros-Sanabria y S. Valbuena "Evaluación de una Arquitectura de Software", Prospectiva, Vol 19, N° 2, 2021.

https://doi.org/10.15665/rp.v19i2.2636

RESUMEN

Una arquitectura de software correctamente diseñada es clave para que las organizaciones puedan avanzar y concentrarse en su función misional. Por otro lado, la Resolución 4505 de 2012 del Ministerio de Salud y Protección Social de Colombia, establece el proceso de validación, control y seguimiento de los informes generados por las Instituciones Prestadoras de Salud de la población vulnerable no asegurada. Para cumplir con lo estipulado en la Resolución, se plantea el desarrollo de una aplicación informática, que mediante su arquitectura de software permita dar respuesta a los constantes cambios exigidos desde el Ministerio.
La secretaria de Salud de Villavicencio requiere un software para dar cumplimiento a la Resolución 4505, para lo cual se desarrolló un ejercicio de selección de una arquitectura de software. Se realizaron evaluaciones tempranas a la arquitectura, en el Sprint Cero. mediante el Método de Análisis de Arquitecturas de Software SAAM, analizando el atributo de calidad modificabilidad, dado los constantes cambios en la Resolución, lo cual permitió la elección de una arquitectura, favorecer la comunicación con los stakeholders y una mejor interpretación de las historias de usuario.

Palabras Clave: Arquitectura de Software, Escenarios, MVC, evaluación de arquitecturas de software, SAAM.

ABSTRACT

A properly designed software architecture is key for organizations to move forward and focus on their missionary role. On the other hand, Resolution 4505 of 2012 of the Ministry of Salud y Protección Social of Colombia, establishes the process of validation, control and monitoring of the reports generated by the Health Provider Institutions of the vulnerable uninsured population. To comply with what is stipulated in the Resolution, the development of a computer application is proposed, which its software architecture allows to respond to the constant changes demanded by the Ministry. The Villavicencio Health Secretariat requires software to comply with Resolution 4505, for which a software architecture selection exercise was developed. Early evaluations of architecture are carried out in Sprint Zero. Through the SAAM Software Architectures Method, analyzing the modifiable quality attribute, given the constant changes in the Resolution, which chose the choice of an architecture, favoring communication with stakeholders and a better interpretation of user stories.

Keywords: *Architecture Software, Scenarios, MVC, evaluation of software architectures, SAAM.*

I. INTRODUCCIÓN

A. PLAN NACIONAL DE DESARROLLO PROSPERIDAD PARA TODOS

El Plan adopta una ruta para solucionar las necesidades de la nación, el cual establece pilares como desarrollo regional, igualdad de oportunidades, consolidación de la paz, innovación y buen gobierno entre otros [1]. Es así, como surge el sistema de evaluación y calificación de las Direcciones Territoriales de Salud, de las Entidades Promotoras de Salud (EPS) e Instituciones Prestadoras de Salud (IPS).

La Resolución 4505 de 2012 de Colombia, determina la estructura del reporte relacionado con el registro de las actividades de Protección Específica, Detección Temprana y enfermedades de interés en salud pública de obligatorio cumplimiento [2], la estructura alimenta el sistema de evaluación y calificación de las Direcciones Territoriales de Salud. Con el fin de dar cumplimiento al artículo número 5, que establece las responsabilidades de las direcciones municipales de salud [3].

La secretaria Municipal de Salud de Villavicencio [SMSV] periódicamente recibe los reportes de las IPS con la información de la población no asegurada [4]. Los reportes son recibidos en archivos planos, como lo indica la Resolución, el número de registros aproximado que se reciben es de 20.000 y cada registro tiene 118 campos. Debido al volumen de información, se presenta un alto número de errores en los registros.

La SMSV realiza la verificación de la información de modo manual, es así que, solo se garantiza una revisión estructural del archivo, permitiendo el paso de registros inconsistentes, razón por la cual no se garantiza la integridad y la veracidad de la información.

Como resultado de lo anterior, se presentan errores en la toma de decisiones en cuanto a las acciones que se le deben realizar sobre un grupo poblacional de interés. Adicionalmente los informes que se presentan a la comunidad y a las dependencias de nivel departamental y nacional contienen datos erróneos.

La SMSV preocupada por orientar adecuadamente sus recursos a la población indicada y cumplir las exigencias del Ministerio de Salud, requiere el desarrollo de un aplicativo que implemente una serie de herramientas que permitan validar e informar de los errores en los archivos planos, de manera detallada, indicando si es el error es de tipo estructural o lógico. De esta manera se permite que los informes que se realicen posteriormente a esta validación conserven características de integridad y veracidad, adicionalmente, se requiere generar alarmas de seguimiento a las citas y los procedimientos establecidos para 5 enfermedades de alto impacto en la población del municipio de Villavicencio.

B. ARQUITECTURA DE SOFTWARE

Dijkstra en 1968 propone generar un artefacto adicional para el entendimiento del software y su construcción [5], permitiendo con éste, la adaptación a nuevos requerimientos y mantenimientos, de manera más eficiente. Este artefacto o artefactos han evolucionado hasta ser llamados Arquitectura de Software (AS), según [6], se cuenta con múltiples definiciones de esta disciplina; a continuación, se enuncian algunas:

- La arquitectura de edificios ofrece ideas a la arquitectura de software, requiere de varios planos para establecer múltiples puntos de vista, la arquitectura de software se compone de 3 aspectos, los cuales son: elementos, forma y justificación [6]
- La estructura de los componentes de un sistema, sus relaciones, principios y directrices que rigen su diseño y evolución a lo largo del tiempo [7].
- La AS es el conjunto de estructuras necesarias para razonar sobre el sistema, que comprende elementos de software, relaciones entre ellos y propiedades de ambos [8].

- Conceptos fundamentales de un sistema en su entorno encarnado en sus elementos, relaciones y en los principios de su diseño y evolución [9].
- Conjunto de elementos estáticos, propios del diseño intelectual del sistema, que definen y dan forma tanto al código fuente, como al comportamiento del software en tiempo de ejecución. Naturalmente este diseño arquitectónico ha de ajustarse a las necesidades y requisitos del proyecto [10].
- La AS de un sistema de computación son las estructuras, que contienen componentes de software, las propiedades externas visibles de dichos componentes y las relaciones entre ellos [11].

Dado lo anterior el estándar ISO/IEC/IEEE 42010:2011, revisado y confirmado en 2017 [9], establece como describir una AS, y lo establece mediante una notación, terminología y mejores prácticas de diseño y descripción, para lograr lo anterior, se describen para ello conceptos básicos, algunos de estos son:
descripción de la arquitectura, describe la arquitectura de cualquier sistema en el que se esté interesado [12], ***vista de arquitectura*** es la representación concreta desde una perspectiva de preocupaciones específicas del sistema [13], "Una preocupación puede ser enmarcada por más de un punto de vista" [13], ***punto de vista de la arquitectura***, define las convenciones para representar un conjunto de preocupación relativo a una arquitectura, incluye un vocabulario apropiado para los elementos de diseño del sistema y reglas para su composición [12] y ***Lenguajes de descripción de arquitectura*** es cualquier forma de expresión utilizada para la descripción de la arquitectura, tales como: Aesop, Rapide, AutoSAR, Wright, xADL, SysML, AADL, Jacal, ArchiMate y los idiomas de punto de vista de RM-ODP [12].

La arquitectura de software brinda soluciones especializadas en algunos dominios, lo cual permite simplificar el proceso de construcción de nuevos sistemas, a través de la reutilización de la infraestructura existente, reduciendo costos y facilitando el mantenimiento de los sistemas [14].
Adicional al debate de contar con una definición de AS aceptada, tenemos el concepto de estilos de AS o patrones arquitectónicos, diferente a patrones de diseño. Estos últimos proporcionan soluciones puntuales o de menos escala en un desarrollo de software.

Un estilo o patrón arquitectónico define una familia de sistemas en términos del esquema de organización estructural [15], otros autores, lo establecen como un concepto descriptivo que define una forma de articulación u organización arquitectónica [16], además, Allen [17] establece que son un conjunto de reglas de diseño que identifica los tipos de componentes y conectores que se pueden utilizar para estructurar el sistema o subsistema, adicionalmente se puede definir como una entidad, consistente en cuatro elementos [18]:

1. Un vocabulario de elementos de diseño: componentes y conectores tales como tuberías, filtros, clientes, servidores, traductores, bases de datos, etcétera.
2. Reglas de diseño o restricciones que determinan las composiciones permitidas de esos elementos.
3. Una interpretación semántica que proporciona significados precisos a las composiciones.
4. Análisis susceptibles de practicarse sobre los sistemas construidos en un estilo, por ejemplo, análisis de disponibilidad para estilos basados en procesamiento en tiempo real, o detección de abrazos mortales para modelos cliente-servidor.

Existen diversos estilos de arquitectura ampliamente difundidos [tuberías y filtros, capas, repositorios, etc.] y otros específicos para dominios particulares. Las arquitecturas para dominios específicos de software proveen una estructura organizacional hecha a medida para una familia de aplicaciones. El conocimiento de diseño en un dominio permite definir un estilo arquitectural para una colección de sistemas relacionados, lo cual incluye un vocabulario apropiado para los elementos de diseño del sistema y reglas para su composición.

A continuación, se describe una pequeña clasificación de estilos tomada de [14] y [18], estas clasificaciones pueden variar dependiendo del autor y sus puntos de vista:

Estilos de Flujo de Datos

Este estilo enfatiza la reutilización y la flexibilidad. Es apropiada para sistemas que implementan transformaciones de datos en pasos sucesivos.

- Tubería y filtros
- Proceso secuencial en lote

Estilos Llamada y Retorno

Este estilo también enfatiza la flexibilidad y la escalabilidad. Son los estilos más generalizados en sistemas en gran escala.

- Arquitecturas en Capas
- Model-View-Controller [MVC]
- Arquitecturas Orientadas a Objetos
- Arquitecturas Basadas en Componentes

Estilos Centrados en Datos

Se enfatiza la integridad de los datos. Se estima apropiada para sistemas que se fundamentan en acceso y actualización de datos en estructuras de almacenamiento.

- Repositorios y pizarra.
- Basadas en hipertextos

Estilos de Código Móvil

Esta clase de estilos enfatiza la portabilidad.

- Sistemas basados en reglas
- Arquitectura de Máquinas Virtuales

C. EVALUACIÓN DE AS

La evaluación de la AS se considera una buena práctica de ingeniería de software, ya que permite la toma de decisiones frente a la calidad y los requerimientos del proyecto, seguidamente se describen algunas de estas.

Software Architecture Analysis Method [SAAM]

Primer método de análisis y evaluación temprana de la AS basado en escenarios, útil para evaluar atributos de calidad de manera rápida, tales como, modificabilidad y funcionalidad, además de su fácil comprensión por parte todos los interesados, cuenta con buena documentación, y un óptimo balance entre esfuerzo y costo de los cambios, los cuales se estiman por anticipado [19]. Estos factores lo hacen propició para este proyecto.

SAAM realiza su evaluación basada en escenarios, los cuales describen interacciones que ocurren o pueden ocurrir entre el sistema y los actores, la creación o formulación de los escenarios estimula a los interesados a considerar futuros usos o cambios del sistema [19].

Architecture Trade-off Analysis Method [ATAM]

ATAM es un método desarrollado por Carnegie Mello, ATAM ayuda a los stakeholders a realizar las preguntas correctas para encontrar decisiones arquitectónicas potencialmente problemáticas y realizar actividades de mitigación en los riesgos descubiertos [20].

Método basado en escenarios de arquitectura para la evaluación de atributos de calidad como modificabilidad, portabilidad, extensibilidad e integrabilidad [19]

.

La aplicación de ATAM se realiza en nueve pasos enunciados en la tabla 1.

Tabla 1. Pasos ATAM

1. Presentación de	2. Factores clave para el

ATAM	negocio
3. Presentación de la AS	4. Identificar enfoques arquitectónicos
5. Generar árbol de utilidad de atributos de calidad	6. Analizar enfoques arquitectónicos
7. Lluvia de ideas y priorizar escenarios	8. Analizar enfoques arquitectónicos
9. Presentar resultados	

Architecture-Level Modifiability Analysis [ALMA]

Dos grupos de investigadores que contaban con métodos para analizar arquitecturas se unieron y consolidaron un método basado en escenarios centrado en el análisis de modificabilidad a nivel de la arquitectura, el cual distingue múltiples objetivos de análisis y establece supuestos explícitos. ALMA consta de cinco pasos [21]:

1. Selección de objetivos
2. Descripción de la arquitectura del software
3. Obtención del escenario de cambio
4. Evaluación del escenario de cambio
5. Interpretación del escenario de cambio

II. METODOLOGÍA

Para brindar una solución a la problemática de validar y generar los reportes de las IPS, solicitados por el Ministerio y definidos por la Resolución 4505, la SMSV se decide adelantar un proyecto de desarrollo de software. bajo una metodología ágil, se eligió Scrum ya que permite afrontar cambios de última hora, facilita la gestión y desarrollo del producto y permite contar con funcionalidades en corto tiempo [22], además debido a los constantes cambios emanados del Ministerio, se requiere hacer una evaluación de la AS que permita su fácil adaptación a los nuevos requerimientos.

Dentro de la metodología Scrum utilizamos Joint Application Development [JAD] para definir el Product Backlog y dado los cambios continuas del anexo técnico de la Resolución, se desarrolló en un Sprint, la evaluación de varias arquitecturas de software mediante la técnica de evaluación temprana SAAM.

A. CAPTURA DE FUNCIONALIDADES

El proceso de identificación y documentación de las funcionalidades se efectuará mediante la técnica JAD [23], la cual se desarrolló en una reunión participativa entre director de SMSV, y el coordinador de Prevención y Promoción [PyP], quien será el Product Owner, el Scrum Máster y los desarrolladores. En la reunión los temas se centrarán más en las necesidades de la secretaria que en lo técnico. Se realizarán preguntas como: que necesitan, quien lo necesita, por qué lo necesitan y para cuando lo necesitan, quien es el responsable de los procesos, quien aporta la información, etc.

Al finalizar la JAD se cuenta con las Historias de Usuario [HU], ver modelo en la tabla 2, y se ha priorizado las funcionalidades del proyecto, en la tabla 3 se listan estas.

Tabla 2. *Modelo historia de usuario*

Identificador	HU1
Como	Director de PyP

Quiero	verificar estructura del archivo plano según Resolución 4505
Para	Generar reportes al Ministerio
Aceptación	El archivo plano cumple con los 118 campos

Tabla 3. *Product Backlog*

Identificador	Enunciado de la HU
HU1	Verificar estructura archivo plano con respecto a la de Resolución 4505/2012
HU2	Validar información del archivo plano con respecto a la de Resolución 4505/2012
HU3	Vigilancia y alertas de las enfermedades de interés en salud pública
HU4	Generación de reporte de archivo plano de la Resolución 4505/2012 archivo validado.

A continuación, se listan algunos de los requerimientos de calidad o no funcionales:

- La aplicación debe procesar los archivos de la IPS en menos de 5 minutos.
- La aplicación debe adaptarse en menos de un mes a los cambios que estipule el Ministerio.
- La aplicación debe adaptarse en menos de un mes a los cambios que estipule la SMSV en cuanto a las alarmas de los indicadores.
- La aplicación contara con un solo usuario, en al menos un año.
- La aplicación debe cumplir con los lineamientos de accesibilidad de gobierno en línea.
- La aplicación debe implementarse con los recursos software y hardware disponibles en la SMSV.

Se estableció como prioritario el atributo de calidad, modificabilidad, guiados por la siguiente definición:

> La modificabilidad de un sistema de software es la facilidad con la que puede ser adaptado para cambios en el entorno, requisitos o especificaciones no funcionales [24].

B. DISEÑO, SPRINT 0

Comprendido el dominio del problema, establecido los requerimientos no funcionales y el Product Backlog estimado y priorizado, se procede a iniciar a un Sprint 0 [25] o kick off [26], en el que se establecen las herramientas tecnológicas con las que cuenta la SMSV para ejecutar el software, se define la plataforma de desarrollo, en este caso es Java EE, porque la SMSV busca una solución software generada con herramientas de licencia libre. Además, se analizan los requerimientos no funcionales para establecer condiciones o restricciones sobre la AS, dada la preocupación de la SMSV por los continuos cambios en el anexo técnico de la Resolución, el cual define los campos de información que contiene cada archivo plano. Es por esto, que se presta atención a la fase de diseño de la arquitectura y su proceso de evaluación. La AS se construye de forma iterativa mediante un análisis de los requisitos funcionales y no funcionales o de calidad.

No es necesario tener todos los requisitos claros para comenzar la fase de diseño arquitectónico, seguido de una evaluación temprana a la arquitectura mediante SAAM, en la figura 1 se observa la secuencia de pasos a realizar en este Sprint 0.

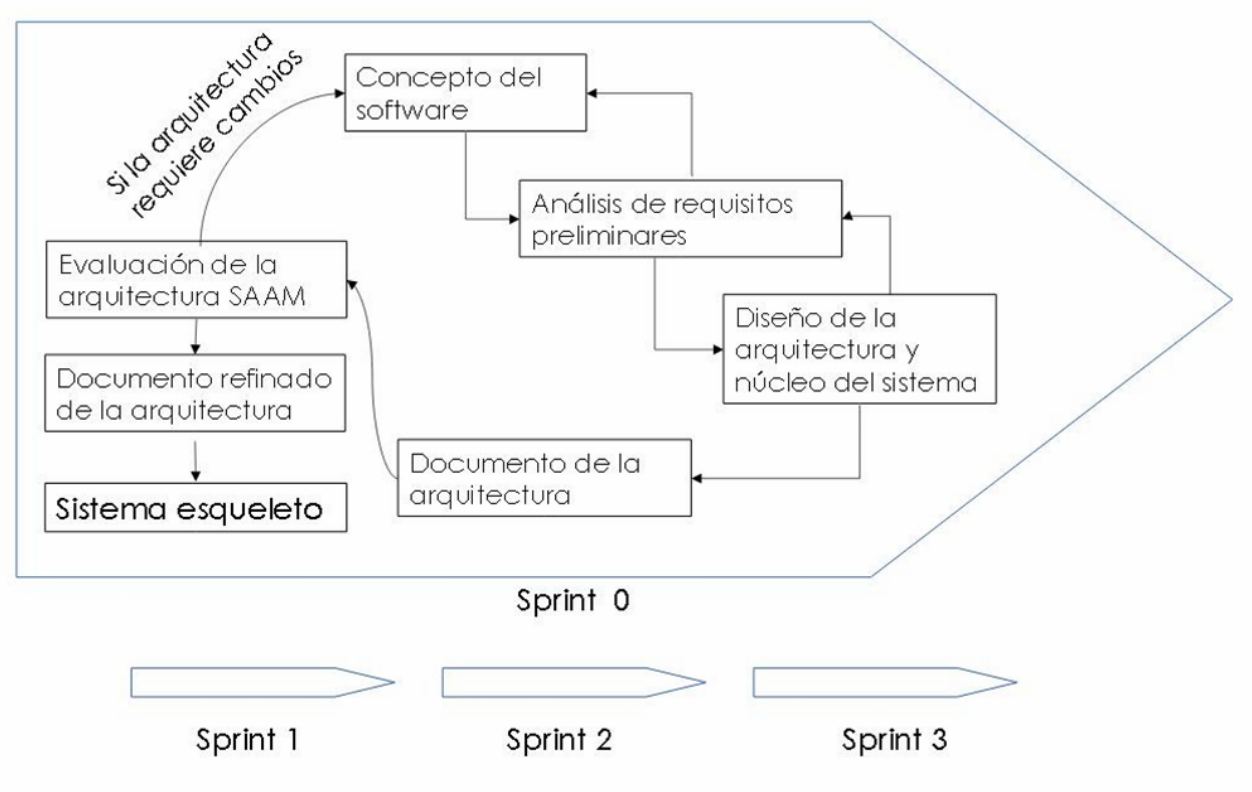

Figura 1. *Gestión de la arquitectura, [26]*

Continuando con el Sprint 0,
Los 7 pasos para desarrollar una sesión de evaluación de SAAM, basado en [26] son los siguientes:

1. Identificar y reunir a los interesados [stakeholders]
2. Desarrollar y priorizar los escenarios, mediante la técnica de lluvia de ideas, los stakeholders proponen situaciones que han ocurrido o pueden ocurrir, y se priorizan según su ocurrencia o necesidades de la SMSV.
3. Presentar y describir las arquitecturas candidatas a los stakeholders utilizando palabras no técnicas, con sus ventajas y desventajas. Las arquitecturas propuestas para brindar una solución son:
   - Cliente/Servidor
   - Orientada a Servicios
   - MVC

En la figura 2 se observa una vista general de la arquitectura MVC y cual es orden del procesamiento para ilustrar a los interesados.

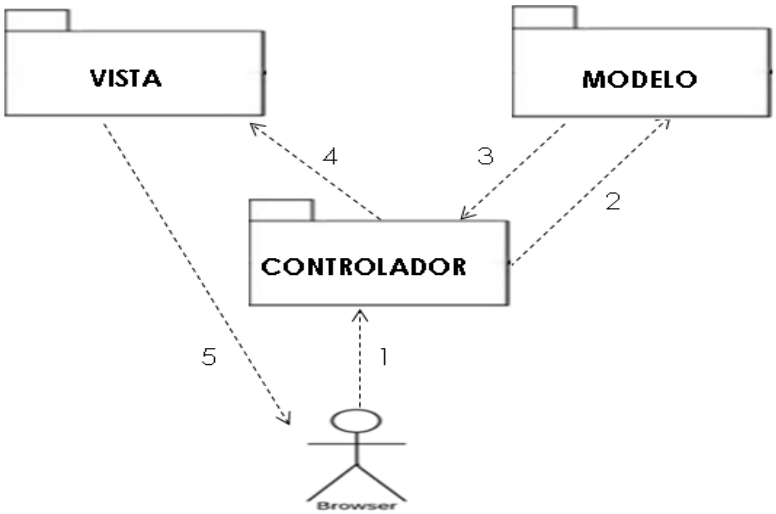

Figura 2. *Procesamiento MVC*

En la Figura 3 se observan las clases importantes y sus relaciones del paquete modelo, en una estructura básica que permite su estudio y análisis.

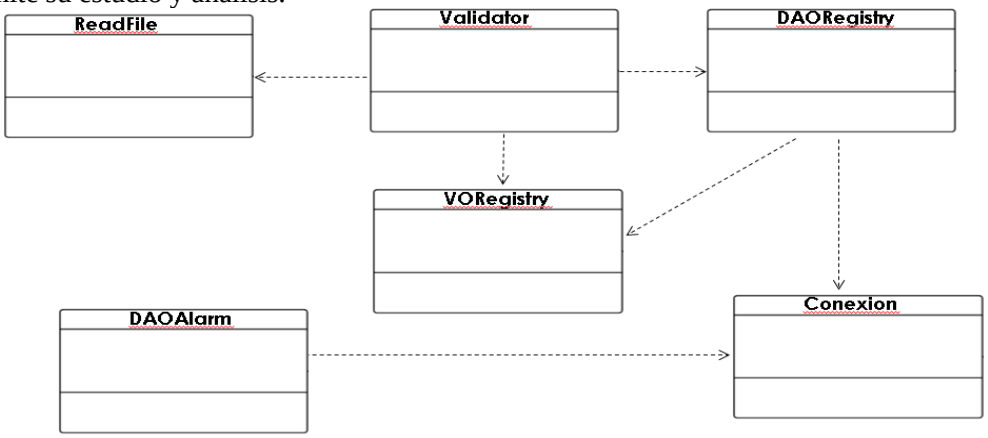

Figura 3. *Estructura básica del Modelo*

4. Clasificar los escenarios directos o indirectos. Los escenarios directos son soportados por la AS y los indirectos son aquellos que implican cambios en el software, estos cambios deben ser cuantificados en la evaluación del escenario para así poder establecer estimaciones del costo del cambio.

Para dar cumplimiento a los pasos ya establecidos, se muestran los escenarios indirectos seleccionados y priorizados, presentados según formato [27]:

Escenario 1

El Ministerio cambia valores de una columna en la estructura de datos [18], los cuales deben reflejarse en el próximo informe.

1. Fuente: Ministerio de Salud y Protección Social
2. Estímulo: cambia valores en columna de la estructura de datos
3. Entorno: sistema en funcionamiento
4. Artefacto: código, base de datos e informe
5. Respuesta: Los informes reflejaran los nuevos valores
6. Medida de la Respuesta: Procesar los nuevos archivos de texto enviados por las IPS
7. Atributo de calidad afectado: Modificabilidad

Escenario 2

El Ministerio agrega columnas en la estructura de datos, los cuales deben reflejarse en el próximo informe.

1. Fuente: Ministerio de Salud y Protección Social
2. Estímulo: Agrega columnas de la estructura de datos
3. Entorno: sistema en funcionamiento
4. Artefacto: código, base de datos e informe
5. Respuesta: Los informes reflejaran las nuevas columnas
6. Medida de la Respuesta: Procesar los nuevos archivos de texto enviados por las IPS
7. Atributo de calidad afectado: Modificabilidad

Escenario 3

La SMSV cambia las enfermedades de las alarmas de indicadores, los cuales deben reflejarse en el próximo informe de indicadores.

1. Fuente: SMSV
2. Estímulo: cambio de enfermedades para la alarma
3. Entorno: sistema en funcionamiento
4. Artefacto: código e informe
5. Respuesta: Las alarmas responden a los nuevos indicadores.
6. Medida de la Respuesta: El informe presenta los indicadores de las nuevas enfermedades.
7. Atributo de calidad afectado: Modificabilidad

5. Evaluación individual de los escenarios, para cada escenario indirecto, se deben establecer los cambios necesarios y estimar el costo de realizarlos para cada arquitectura candidata.

La Tabla 4 muestra las clases, los cambio a realizar y su costo.

Tabla 4 . Evaluación de Escenarios

Escenario	Módulo	Cambios	Unidad de Costo
1	Validator	Agregar nuevo valor al proceso de validación.	1
2	Validator VO Registry	Modificar estructura, al agregar un campo nuevo en la tabla y en la clase agregar método que valide ese nuevo campo	5
3	DAO Alarm	Modificar estructura de alarma	3

6. Interacción de escenarios, la interacción se presenta cuando dos o más escenarios involucran una misma clase o un módulo, generando una revisión de la arquitectura y en especial de las clases involucradas para descomponer algunas de sus funcionalidades.

Se considera que no hay interacción entre los escenarios 1 y 2, ya que el cambio propuesto es muy discreto.

7. Evaluación global, como paso final, se asigna un valor a cada escenario basado en la importancia o

prioridad que tienen para la SMSV, en la Tabla 5 se puede observar los escenarios, los costos y la prioridad.

8.

Tabla 5. Ponderación de Escenarios

Escenario	Unidad de Costo	Prioridad	Arquitectura
1	1	Alto	MVC
2	5	Bajo	
3	3	Medio	
1	2		Cliente/servidor
2	6		
3	3		
1	1		SOA
2	6		
3	4		

3. RESULTADOS Y DISCUSIÓN

Guiados por la tabla número 5 se establece que la arquitectura MVC ofrece un costo bajo para afrontar los cambios que requieran los escenarios establecidos por la SMSV. Las arquitecturas SOA y Cliente/Servidor salen desfavorecidas en el escenario 2 por las unidades de costo que asumiría la SMSV para las actualizaciones, este escenario 2 es el más habitual.

La arquitectura de software MVC, pertenece al estilo Llamada y Retorno [17], ofrece una capacidad para afrontar la escalabilidad y modificabilidad, dado que la arquitectura favorece la modularidad [cohesión y acoplamiento], permitiendo una fácil actualización, reduce la introducción de errores al contar con rutas de procesos separados, adicionalmente se cuenta con la separación de la vista que admitiría a futuro la visualización en diversos dispositivos. En la actualidad existen varios framework, tales como Struts, Spring MVC y JSF, que facilitan el desarrollo de este tipo de proyectos.

La AS tiene un impacto importante en la calidad de las aplicaciones software, para este desarrollo era primordial favorecer la modificabilidad. La utilización de SAAM para evaluar la AS permite a los stakeholders mejorar la comunicación, visionar o predecir cambios, entender la arquitectura, captar la relación de los cambios con la AS y una mejor interpretación de las historias de usuario, por otro lado, la documentación relacionada con arquitectura de software es mejorada. Sin embargo, la descripción de las arquitecturas puede resultar confusa a la hora de realizar comparaciones.

Es de tener en cuenta que SAAM es susceptible de ser afectada por los temores o intereses de los stakeholders en la formulación de los escenarios y su evaluación, es primordial que el equipo o el arquitecto controle eso aspectos. Adicionalmente no hay métricas o guías confiables en la aplicación de la evaluación.

III. CONCLUSIONES

La inclusión del Sprint cero permitió realizar una evaluación temprana de las arquitecturas de software candidatas, con lo cual se mejora la calidad de las decisiones de diseño adoptadas para el proyecto.

La evaluación basada en escenarios ofreció a los interesados de la SMSV identificar y proponer posibles cambios futuros al sistema, lo cual permitió diseñar el aplicativo de forma que se pudiera ajustar rápidamente.

SAAM es un método para la evaluación temprana de arquitecturas, con respecto a algunos atributos de calidad expresados en el contexto de circunstancias específicas (escenarios), que posee algunos puntos en común con los conceptos tradicionales de acoplamiento y cohesión.

El apropiado desarrollo e implementación de la arquitectura MVC para el sistema de seguimiento a la Resolución 4505, le permite a la SMSV ajustarse a los cambios que generen el Ministerio o la SMSV.

Este proceso se puede realizar en cualquier secretaria de salud que deba dar cumplimiento a la Resolución 4505, solo es necesario ajustar algunos de los escenarios a sus requerimientos particulares.

REFERENCIAS

[1] Salud y Protección Social. Ministerio [Internet], [Desconocido] Resolución 4505 de 2012, Colombia, 2012. 2004 [citado 12 septiembre 2019] disponible en https://www.minsalud.gov .co/sites/rid/Lists/BibliotecaDigital/RIDE/DE/DIJ/Resolucion-4505-de-2012.PDF

[2] Salud y Protección Social. Ministerio [Internet], [Desconocido] Guías de promoción de la salud y prevención de enfermedades en la salud pública. Colombia. 2012. 2004 [citado 12 septiembre 2019] disponible en http://www.minsalud.gov.co/Documentos%20y%20Publicaciones/GUIAS%20DE%20ATENCION%20-TOMO%20DOS.pdf

[3] Salud y Protección Social. Ministerio [Internet], [Desconocido] ABECÉ Resolución 4505 de 2012. Colombia. 2016. [citado 12 septiembre 2019] disponible en https://www.minsalud. gov.co/sites/rid/Lists/BibliotecaDigital/RIDE/VS/ED/GCFI/abece-resolucion-4505.pdf

[4] Salud y Protección Social. Ministerio [Internet]. [Desconocido] Anexo Tecnico No. 1 Resolución 4505 de 2012. Colombia. 2012. [citado 10 septiembre 2019] disponible en: https://hsam .gov.co/sites/default/files/2019-10/Anexo-1-Estructura-base-datos-Resolucion-4505.pdf

[5] Dijkstra E.The Structure of the THE multiprogramming system. Communications of the ACM. 1983 ene; 26 [1]:49-52.

[6] Perry D, Wolf L. Foundations for the study of software architecture. Software Engineering Notes. 1992 oct; 17[4]: 40–52.

[7] Department of Defense [Internet], [Lugar desconocido] Modeling and Simulation Glossary.2017. [Citado 10 dic 2019]. Disponible en: https://apps.dtic.mil/dtic/tr/fulltext/ u2/a349800.pdf

[8] Bass L, Clements P, Kazman R. Software Architecture in Practice 3ª ed, Addison-Wesley 2012. 4p

[9] ISO/IEC/IEEE [Internet], [Lugar desconocido] Standard 42010:2011 Systems and software engineering — Architecture description. [citado 15 enero 2020]. Disponible en: https://www.iso.org/ obp/ui/#iso:std:iso-iec-ieee:42010:ed-1:v1:en

[10] Bass L, Clements P, Kazman R. Software Architecture in Practice 1ª ed, Addison-Wesley 1998. 11p

[11] Shaw M, Garlan D. Software Architecture: Perspectives on an Emerging Discipline, Financial Times/Prentice Hall. 1996

[12] Padilla J. [Internet]. [Lugar desconocido] Diseño de una arquitectura institucional tecnológica basada en el estándar ISO/IEC/IEEE 42010 que permita diversificar el modelo educativo dentro de la Universidad Técnica del Norte. 2019 [Citado 10 ene 2020]. Disponible en: http://repositorio.utn.edu.ec/ bitstream/123456789/9537/2/04%20ISC%20526%20TRABAJO%20GRADO.pdf#page=30&zoom=100,92,96

[13] Escobar J [Internet]. [Lugar desconocido] Propuesta de un marco de trabajo de arquitectura para el aseguramiento de la calidad en el diseño de videojuegos serios orientados a la rehabilitación física de acuerdo a la Norma ISO/IEC/IEEE 42010. Escuela Politécnica Nacional, 2019 [Citado 5 feb 2019]. Disponible en: https://bibdigital.epn.edu. ec/bitstream/15000/20119/1/CD%209554.pdf
[14] Garlan D, Shaw M. An Introduction to Software Architecture. Advances in Software Engineering and Knowledge Engineering, 1994 Volume I, World Scientific Publishing Company, New Jersey.
[15] Clements P. A Survey of Architecture Description Languages. Proceedings of the International Workshop on Software Specification and Design, Alemania, 1996.
[16] Shaw M, Clements P. A field guide to Boxology: Preliminary classification of architectural styles for software systems. Documento de Computer Science Department and Software Engineering Institute, Carnegie Mellon University. Publicado en Proceedings of the 21st International Computer Software and Applications Conference. 1997
[17] Allen R, Garlan D. The Wright Architectural Description Language, Technical Report, Carnegie Mellon University. 1996.
[18] Reynoso C, Kicillof N [Internet], [Desconocido] Introducción a la Arquitectura de Software, Universidad de Buenos Aires. 2004 [citado 12 enero 2020]. Disponible en: https://www. academia.edu/5472068/Estilos_PDF
[19] Ionita M, Hammer D, Obbink H. Scenario-based software architecture evaluation methods: An overview. In Workshop on methods and techniques for software architecture review and assessment at the international conference on software engineering [pp. 19-24]. 2002
[20] Nord, Robert L., et al. Integrating the Architecture Tradeoff Analysis Method [ATAM] with the cost benefit analysis method [CBAM]. Carnegie-Mellon Univ Pittsburgh Pa Software Engineering Inst, 2003.
[21] Bengtsson P., Lassing N., Bosch J., Vliet less H. Architecture-level modifiability analysis [ALMA], Journal of Systems and Software, vol 69, 2004
[22] Scrum Master [Internet]. [Desconocido] Scrum Manager Body of Knowledge.2012. [citado 10 agosto 2019] disponible en: https://www. scrummanager.net/bok/index.php?title=Scrum_Manager_BoK
[23] Duggan, Evan W.; THACHENKARY, Cherian S. Integrating nominal group technique and joint application development for improved systems requirements determination. Information & Management, 2004.
[24] Bosch J, Bengtsson P, Lassing N, Vliet H. Analyzing Software Architectures for Modifiability. Department of Software Engineering and Computer Science, University of Karlskrona/Ronneby Ronneby, Sweden. 2000
[25] Mago E, Alférez G. El Papel de la Arquitectura de Software en Scrum, Software Guru n° 30, 2011 Disponible en: https://sg.com.mx/revista /30/el-papel-la-arquitectura-software-scrum#.V-HsbfArLIU.
[26] Agripino P.,[Internet] Sprint 0, clave en la gestión de proyectos ágiles, Paradigma Digital. España 2017, [citado 12 enero 2020] disponible en https://www.paradigmadigital.com/ techbiz/sprint-0-clave-la-gestion-proyectos-agiles/
[27] Marta Jiménez Franco [Internet], 2014, [citado 12 septiembre 2019] disponible en https://github.com/Arquisoft/ObservaTerra11/wiki/Escenarios-de-calidad