Universidad de las Ciencias Informáticas

Facultad 4

Aplicación de un procedimiento para evaluar la Arquitectura de Software del proyecto Modernización del Sistema Bancario Cubano.

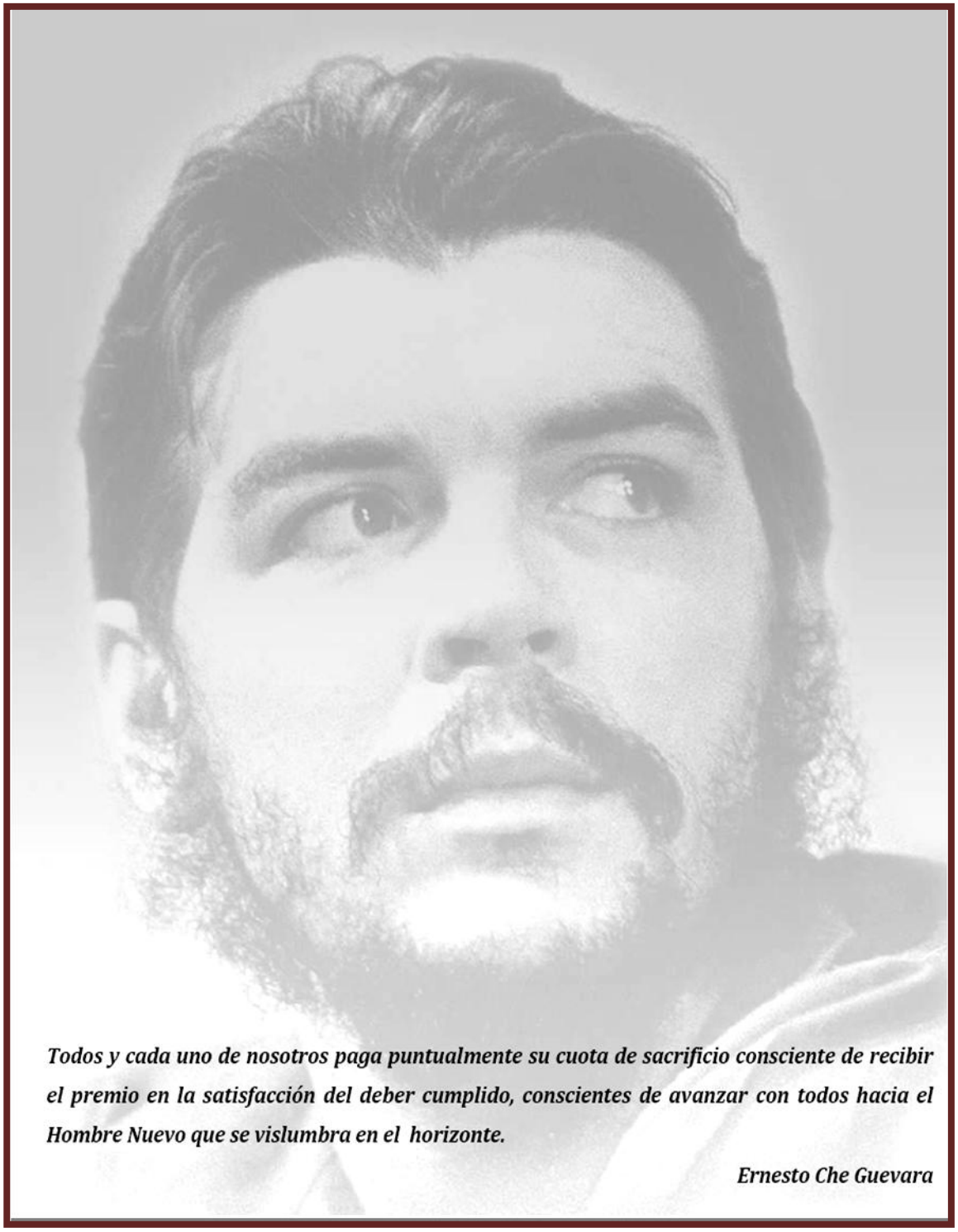
Trabajo de Diploma para optar por el título de Ingeniero en Ciencias Informáticas

Autores: Yurisel Aguirre Pérez
Carel Torres Oliva

Tutor: Ing. Adolfo Miguel Iglesias Chaviano

Ciudad de La Habana, Junio 2009

Año del 50 Aniversario del Triunfo de la Revolución.


Todos y cada uno de nosotros paga puntualmente su cuota de sacrificio consciente de recibir el premio en la satisfacción del deber cumplido, conscientes de avanzar con todos hacia el Hombre Nuevo que se vislumbra en el horizonte.
Ernesto Che Guevara

Declaración de Autoría

Declaramos ser autores de la presente tesis y reconocemos a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma, con carácter exclusivo.

Para que así conste firmo la presente a los ____ días del mes de _____ del año _____.

Yurisel Aguirre Pérez

_____

Firma del Autor

Carel Torres Oliva

_____

Firma del Autor

Adolfo Miguel Iglesias Chaviano

_____

Firma del Tutor

Datos de Contacto

Tutor: Ing. Adolfo Miguel Iglesias Chaviano.

Profesor adiestrado. Graduado en Julio del 2007 de Ingeniero en Ciencias Informáticas en la Universidad de las Ciencias Informáticas con Título de Oro y Promedio General de 5.09 puntos. Seleccionado Premio “Julio Antonio Mella” en la primera graduación de dicha Universidad. Ha impartido las asignaturas de Programación I y II, Ingeniería de Software I y II y el curso de Patrones de Diseño en la Universidad de las Ciencias Informáticas. Formó parte del equipo que elaboró dicho curso.

Miembro de la Reserva del Comandante en Jefe. Participó como desarrollador en el proyecto SAFRE y en un Sistema de encuesta sobre temas de Nefrología. Analista y desarrollador del Subsistema Sala Situacional del Sistema de Gestión Penitenciaria para la República de Venezuela. Ha elaborado e impartido curso de capacitación al proyecto relacionado con la modernización del Sistema Bancario Cubano. Arquitecto Principal del proyecto relacionado con la modernización del Sistema Bancario Cubano.

Correo electrónico: aiglesias@uci.cu

Agradecimientos

A nuestro tutor Adolfo Miguel Iglesias Chaviano, que sin su ayuda y sus consejos nos hubiera sido imposible la confección de este documento, por su paciencia a la hora de revisarnos la tesis y por las horas que tuvo que dedicarle en su tiempo extra.

Gracias por todo.

A la Revolución y a nuestro Sistema Social que nos permite a nosotros, jóvenes sin grandes recursos, hacer realidad nuestros sueños de convertirnos algún día en graduados universitarios.

A nuestros padres por su apoyo y confianza en nosotros.

A nuestras amistades, esas que son tantas y que por temor a pecar no podemos mencionar a algunos y a otros no. A ustedes que en algún momento nos ayudaron con un consejo o alguna idea.

A nuestros compañeros de aula, que nos alentaron y confiaron en nosotros.

A nuestros profesores, que sin su enseñanza nuestros sueños no se hubieran hecho realidad.

Y a todos aquellos que de una forma u otra contribuyeron a la realización de este trabajo.

Yurisel y Carel

Dedicatoria

A mis padres, Ana Luisa Pérez Matos y Víctor Aguirre de la Cruz, a quienes les estaré eternamente agradecida por darme esta oportunidad de estudiar y saber siempre que cuento con su apoyo desinteresado, en los momentos buenos y sobre todo, en los malos. A ustedes les digo que me esforcé todo lo que pude y que ya casi tengo un gran sueño realizado, el tener un título universitario, en una carrera que me gusta y se además, que este es un sueño de ustedes también. Sólo me queda decirles lo siguiente, lo cual he tratado de que se convierta en una línea durante toda mi vida:

"Que adorno más grande puede haber para un hijo que la gloria de un padre, o para un padre que la conducta honrosa de un hijo."

A mi hermano Yunior Aguirre Pérez, el cual siempre me dio su apoyo. A ti te digo que no importa cuántas veces la vida te haga caer, ella te está poniendo a prueba, lo realmente importante es levantarse y seguir con más bríos y fuerzas que antes. Te deseo lo mejor en tu profesión de informático, se que tienes para eso y mucho más porque lo has demostrado más de una vez. Confío en ti.

A mis abuelos Aurea y Agustín quienes quiero y aprecio mucho ya que siempre han estado presentes con su ejemplo y me han dado consejos en la vida a través de sus cuentos y sus regaños. Para ustedes también va dedicada esta tesis, y aunque quizás no puedan leerla completa sepan que los llevo en el corazón y su ejemplo está en mi recuerdo cada vez que escribo una letra.

A mi familia universitaria, Anay, Weeden, Sucel, Suri, Ariadna, Felipe y a mi tía Odaisy, muchas gracias por ser su hija recogida y la mejor familia adoptiva que he tendido y a Maybis por la ayuda prestada en la elaboración de este documento, muchas gracias.

Y a todas aquellas personas que han confiado en mí y en algún que otro momento me han tendido su mano solidaria, o me han dado algún consejo.

A todos muchas gracias.

Yurisel

DEDICATORIA

A mis padres, por el apoyo constante que me han brindado en todos mis años de estudio. A todos mis amigos, los cuales siempre me han brindado en un momento u otro su ayuda.

C*arel*

Resumen

En el presente Trabajo de Diploma se realiza un estudio del estado del arte de los métodos de evaluación de arquitectura SAAM, ATAM, MECABIC y BOSCH, brindando sus principales características y pasos que siguen para efectuar la evaluación, se plasman además diferentes técnicas empleadas a la hora de evaluar arquitecturas, en dependencia del estado en que esta se encuentre. También se efectúa la evaluación de la arquitectura del proyecto Modernización del Sistema Bancario Cubano, brindando la posibilidad de observar en la práctica el desarrollo de una evaluación de arquitectura.

Palabras claves

Arquitectura de Software, Atributo de Calidad, Modelo de calidad, Evaluación de Arquitectura

TABLA DE CONTENIDO

TABLA DE CONTENIDO

TABLA DE CONTENIDO

INDICE DE TABLAS

ÍNDICE DE FIGURAS

Introducción

La informática como ciencia que estudia el tratamiento automático de la información, está sujeta a los cambios tecnológicos que aparecen continuamente con tendencia al desarrollo constante. Es una corriente innovadora, dinámica y muy popular que ha cautivado la atención de la población y de las empresas a nivel internacional. Mediante la tecnología digital se pueden realizar operaciones que facilitan el intercambio de información seguro, rápido y eficiente.

La Revolución Cubana, a pesar del recrudecimiento del criminal bloqueo impuesto por los gobiernos estadounidenses desde hace más de una década, se ha propuesto la tarea de informatizar la sociedad como una de las medidas de la Batalla de Ideas que libra nuestro pueblo, asegurando el proceso de automatización de escuelas, empresas e instituciones estatales.
El objetivo que se propone es que nuestro país se encuentre eventualmente entre las principales potencias de la producción de software a través de la creación de instituciones productoras de software, con el fin de ampliar nuestras posibilidades de competir en un mercado lucrativo como es el caso de la Informática y las Telecomunicaciones.

El diseño de la arquitectura de software es una de las etapas fundamentales y, en muchos casos, la más importante en el desarrollo de software, pues es aquí donde se aportan todos los conocimientos, creatividad y experiencia de los desarrolladores para crear la mejor propuesta de solución que se dará al cliente.
A lo largo de este proceso de desarrollo y diseño, los atributos de calidad juegan un papel importante, pues en base a estos se generan las decisiones de diseño y argumentos que los justifican. Dado que la arquitectura de software inhibe o facilita los atributos de calidad, resulta de particular interés analizar la influencia de ciertos elementos de diseño utilizados para la definición de la misma, determinando sus características. Los atributos de calidad deben tenerse en cuenta para determinar la estructura y el comportamiento que tendría un sistema. Hacer un adecuado balance entre funcionalidad, eficiencia,

mantenibilidad y la usabilidad, entre otros atributos, es esencial para obtener un sistema que cumpla las expectativas de las distintas partes interesadas.

En la Universidad de las Ciencias Informáticas (UCI, por sus siglas), como parte del proceso de producción de software y de las necesidades actuales que tiene para el logro de sus objetivos, se demanda la construcción de grandes y complejos sistemas de software que requieren la combinación de diferentes tecnologías y plataformas de hardware y software para alcanzar un funcionamiento acorde con dichas necesidades.

Lo anterior exige poner especial atención y cuidado al diseño de la arquitectura bajo la cual estará soportado el funcionamiento de sus sistemas. En la Facultad 4 de la UCI se desarrollan varios proyectos informáticos uno de ellos es: Modernización del Sistema Bancario Cubano, centrando su objetivo en el desarrollo de un nuevo sistema basado en plataformas libres y la suma de nuevas funcionalidades. Con vista a lograr calidad en los productos a desarrollar por el proyecto se creó un equipo interno de calidad, el cual tiene como objetivos: guiar y velar por el buen funcionamiento del proyecto durante su desarrollo y validar internamente los productos que se desarrollen antes de ser evaluados por el grupo de calidad de la UCI.

Se elaboró el Plan de Aseguramiento de la calidad de modo que describa cada una de las actividades a desarrollar en aras de lograr un proceso de desarrollo de software con la calidad requerida. El análisis y revisión de la arquitectura es uno de los aspectos que se tiene en cuenta a la hora de elaborar dicho plan; pero solamente se hace referencia al momento de ejecución de esta actividad.

Por tanto, se pudiera decir que no existe ningún artefacto que refleje qué, cómo y cuándo se hace necesaria la evaluación de la arquitectura definida para el desarrollo del sistema. En caso de no definir correctamente los atributos de calidad que debe cumplir una arquitectura de software para un producto determinado, puede traer consigo incorrectas decisiones de diseño, por consiguiente, se afectaría el proyecto final.

Por otra parte, si se determinaran los atributos de calidad; pero las decisiones arquitectónicas no fueran correctas, también afectaría al producto final. Una identificación de estos defectos en un nivel avanzado del desarrollo del software implicaría atraso en el cronograma, deficiencia en la producción y falta de competitividad en el desarrollo de un sistema. Por tanto, debido a que no existe un proceso definido para evaluar los elementos de calidad que debe cumplir la arquitectura propuesta queda definido como:

Problema a resolver

La carencia de un procedimiento para controlar y evaluar la arquitectura de software del proyecto Modernización del Sistema Bancario Cubano.

Objeto de estudio

La evaluación de arquitecturas de software.

Campo de acción

La evaluación de la arquitectura de software del proyecto Modernización del Sistema Bancario Cubano.

Objetivo general

Proponer un procedimiento de evaluación para la arquitectura de software definida en el proyecto Modernización del Sistema Bancario Cubano.

Hipótesis

La aplicación de un procedimiento para la evaluación de la arquitectura de software del proyecto Modernización del Sistema Bancario Cubano, permitirá evaluar la arquitectura del software deseado, lo que tributará a la calidad del producto que se desarrolla.

Tareas de investigación

- Caracterización de los procedimientos de evaluación de arquitectura.
- Definición de los atributos de calidad que se pueden evaluar en la arquitectura de software del proyecto.
- Selección de un método para evaluar la arquitectura de software del proyecto.

- Evaluación de la arquitectura del proyecto Modernización del Sistema Bancario Cubano.

Metodología de la investigación

Desde el punto de vista metodológico, para la investigación subyacente relacionada con el presente Trabajo de Diploma se emplearon los siguientes métodos de investigación científica:

Métodos teóricos

Analítico-Sintético: Para el procesamiento de la información y arribar a las conclusiones de la investigación, así como para precisar las características del procedimiento arquitectónico propuesto.

Análisis histórico-lógico: Permite deducir lógicamente cómo ha evolucionado la arquitectura.

Métodos empíricos

Observación: Para la percepción selectiva de las restricciones y propiedades del sistema, y sistémica para el control de la evolución de la arquitectura.

Capítulo 1: Fundamentación Teórica

1.1 Introducción

En el presente capítulo se caracterizan cuatro métodos de evaluación de arquitectura de software y se plasman los conceptos fundamentales relacionados con la misma.

1.2 Arquitectura de Software

El concepto de arquitectura de software se usa de forma amplia y en campos muy distintos, por lo que su significado es algo difuso. Actualmente, es posible encontrar numerosas definiciones del término arquitectura de software, cada una con planteamientos diversos. Se hace evidente que su conceptualización sigue todavía en discusión, puesto que no es posible dirigirse a un diccionario en busca de un significado y tampoco existe un estándar que pueda ser tomado como marco de referencia. [1]

Algunos autores plantean que la arquitectura de software:

"…es el nivel del diseño del software donde se definen la estructura y propiedades globales del sistema". [2]

(Garlan y Perry, 1995)

"…se centra en aquellos aspectos del diseño y desarrollo que no pueden tratarse de forma adecuada dentro de los módulos que forman el sistema." [3]

(Shaw y Garlan, 1996)

La IEEE Std 1471-2000 la define así: "La Arquitectura del Software es la organización fundamental de un sistema formada por sus componentes, las relaciones entre ellos y el contexto en el que se implantarán, y los principios que orientan su diseño y evolución". [6][17]

1.3 Atributo de calidad

Los atributos de calidad son los aspectos del sistema, que en general, no afectan directamente a la funcionalidad necesaria, sino que definen la calidad y las características que el sistema debe soportar. Según Barbacci et al. (1995) la calidad de software se define como el grado en el cual el software posee una combinación deseada de atributos. Tales atributos son requerimientos adicionales del sistema (Kazman et al., 2001), que hacen referencia a características que este debe satisfacer, diferentes a los requerimientos funcionales. [4][23]

Estas características o atributos se conocen con el nombre de atributos de calidad, los cuales se definen como las propiedades de un servicio que presta el sistema a sus usuarios (Barbacci et al. 1995). A grandes rasgos, Bass et al. (1998) establece una clasificación de los atributos de calidad en dos categorías: [5]

- Observables vía ejecución: aquellos atributos que se determinan del comportamiento del sistema en tiempo de ejecución.

- No observables vía ejecución: aquellos atributos que se establecen durante el desarrollo del sistema.

1.3.1 Relación de los atributos de calidad con la arquitectura

Para alcanzar un atributo de calidad específico, es necesario tomar decisiones de diseño arquitectónico que requieren un pequeño conocimiento de funcionalidad. Por ejemplo, el desempeño depende de los procesos del sistema y su ubicación en los procesadores, caminos de comunicación y otros. Por otro lado, se establece que al considerar una decisión de arquitectura de software, el arquitecto se pregunta cuál será el impacto de ésta sobre ciertos atributos. [7]

Por esta razón, se afirma que cada decisión incorporada en una arquitectura de software puede afectar potencialmente los atributos de calidad. Cada decisión tiene su origen en preguntas acerca del impacto sobre estos atributos, y el arquitecto puede argumentar cómo la decisión tomada permite alcanzar algún objetivo. Con frecuencia, el objetivo es un atributo de calidad en particular. Si las decisiones que se toman

sobre esa arquitectura determinan las características de calidad del sistema; es necesario evaluar las decisiones de tipo arquitectónico con respecto al impacto que ellas causarán sobre estos atributos de calidad. [7]

1.4 Modelos de calidad

Los modelos de calidad resultan de utilidad para la predicción de confiabilidad y en la gerencia de calidad durante el proceso de desarrollo, así como para efectuar la medición del nivel de complejidad de un sistema de software (Kan et al, 1994). Es interesante destacar que la organización y descomposición de los atributos de calidad ha permitido el establecimiento de modelos específicos para efectos de la evaluación de la calidad arquitectónica. [24]

Pressman (2002) indica que los factores que afectan a la calidad del software no cambian, por lo que resulta útil el estudio de los modelos de calidad que han sido propuestos en este sentido desde los años 70 dado que los factores de calidad presentados para entonces siguen siendo válidos. Un modelo de calidad es un conjunto de buenas prácticas para el ciclo de vida del software, enfocado en los procesos de gestión y desarrollo de proyectos. [31]

1.4.1 Modelo McCall

El modelo de McCall et al. (1977) describe la calidad como un concepto elaborado mediante relaciones jerárquicas entre factores de calidad, en base a criterios y métricas de calidad. Este enfoque es sistemático, y permite cuantificar la calidad a través de las siguientes fases: [31]

- Determinación de los factores que influyen sobre la calidad del software.
- Identificación de los criterios para juzgar cada factor.
- Definición de las métricas de los criterios y establecimiento de una función de normalización que define la relación entre las métricas de cada criterio y los factores correspondientes.

- Evaluación de las métricas.

- Correlación de las métricas a un conjunto de guías que cualquier equipo de desarrollo podría seguir.

- Desarrollo de las recomendaciones para la colección de métricas en el modelo de McCall. Los factores de calidad se concentran en tres aspectos importantes de un producto de software: características operativas, capacidad de cambios y adaptabilidad a nuevos entornos. En este modelo, el término factor de calidad define características claves que un producto debe exhibir. Los atributos del factor de calidad que define el producto son los nombrados criterios de calidad. Las métricas de calidad denotan una medida que puede ser utilizada para cuantificar los criterios. McCall et al. (1977) identifica una serie de criterios, tales como rastreabilidad, simplicidad, capacidad de expansión, etc.

1.4.2 Modelo de Dromey

Dromey propuso un marco de referencia o metamodelo para construcción de modelos de calidad, basado en cómo las propiedades medibles de un producto de software pueden afectar los atributos de calidad generales, como por ejemplo, confiabilidad y mantenibilidad. El problema que se plantea es cómo conectar tales propiedades del producto con los atributos de calidad de alto nivel. [31]

Para solventar esta situación, Dromey sugiere el uso de cuatro categorías que implican propiedades de calidad, que son: correctitud, internas, contextuales y descriptivas. El proceso de construcción de modelos de calidad propuesto por Dromey consta de 5 pasos, basados en las propiedades mencionadas. Los pasos del marco de referencia propuesto son: [31]

- Especificación de los atributos de calidad de alto nivel (por ejemplo, confiabilidad, mantenibilidad).

- Determinación de los distintos componentes del producto a un apropiado nivel de detalle (por ejemplo, paquetes, subrutinas, declaraciones).

- Para cada componente, determinación y categorización de sus implicaciones más importantes de calidad.

- Proposición de enlaces que relacionan las propiedades implícitas a los atributos de calidad, o, alternativamente, uso de enlaces de las cuatro categorías de atributos propuestas.

- Iteración sobre los pasos anteriores, utilizando un proceso de evaluación y refinamiento para ilustrar sus planteamientos, Dromey (1996) demuestra el uso de su procedimiento para la construcción de un modelo de calidad de implementación, un modelo de calidad de requerimientos, y un modelo de calidad de diseño.

1.4.3 Modelo FURPS

El modelo de McCall ha servido de base para modelos de calidad posteriores, y este es el caso del modelo FURPS, producto del desarrollo de Hewlett-Packard (Grady et al., 1987). En este modelo se desarrollan un conjunto de factores de calidad de software, bajo el acrónimo de FURPS: funcionalidad, usabilidad, confiabilidad, desempeño y capacidad de soporte. [31]

El modelo FURPS incluye, además de los factores de calidad y los atributos, restricciones de diseño y requerimientos de implementación, físicos y de interfaz (Grady et al., 1987). Las restricciones de diseño especifican o restringen el diseño del sistema. Los requerimientos de implementación especifican o restringen la codificación o construcción de un sistema (por ejemplo, estándares requeridos, lenguajes, políticas). Los requerimientos de interfaz especifican el comportamiento de los elementos externos con los que el sistema debe interactuar. Por último, los requerimientos físicos especifican ciertas propiedades que el sistema debe poseer, en términos de materiales, forma, peso, tamaño (por ejemplo, requisitos de hardware, configuración de red). [31]

1.4.4 Modelo ISO/IEC 9126

El estándar ISO/IEC 9126 ha sido desarrollado en un intento de identificar los atributos claves de calidad

para un producto de software (Pressman, 2002). Este estándar es una simplificación del Modelo de McCall (Losavio et al., 2003), e identifica seis características básicas de calidad que pueden estar presentes en cualquier producto de software. Este provee una descomposición de las características en subcaracterística. [31]

Es interesante destacar que los factores de calidad que contempla el estándar ISO/IEC 9126 no son necesariamente usados para mediciones directas (Pressman, 2002), pero proveen una valiosa base para medidas indirectas, y una excelente lista para determinar la calidad de un sistema. [31]

1.4.5 ISO/IEC 9126 Adaptado para arquitecturas de software

Proponen una adaptación del modelo ISO/IEC 9126 de calidad de software para efectos de la evaluación de arquitecturas de software. El modelo se basa en los atributos de calidad que se relacionan directamente con la arquitectura: funcionalidad, confiabilidad, eficiencia, mantenibilidad y portabilidad. [34]

Los autores plantean que la característica usabilidad propuesta por el modelo ISO/IEC 9126 puede ser refinada para obtener atributos que se relacionan con los componentes de la interfaz con el usuario. Dado que estos componentes son independientes de la arquitectura, no son considerados en la adaptación del modelo. [34]

1.5 Evaluación de Arquitectura

La evaluación de arquitectura de software es un estudio de factibilidad que pretende detectar posibles riesgos, así como soluciones para enmendarlos. “En general, una evaluación debe realizarse cuando hay suficiente de la arquitectura como para justificarlo. Una buena regla sería: realizar una evaluación cuando el equipo de desarrollo empieza a tomar decisiones que dependen de la arquitectura y el costo de deshacerlas sobrepasa al costo de realizar una evaluación.” En la actualidad existen varios métodos de evaluación de arquitectura, como por ejemplo: MECABIC, ATAM, SAAM y Bosch, los cuales son caracterizados en este trabajo teniendo en cuenta características como: los atributos de calidad que evalúan y las etapas de sus procesos de evaluación, entre otras. [8][9][21]

1.5.1 Mecabic

El método Mecabic es una solución al problema de la inexistencia de métodos que evalúen la Calidad de las Arquitecturas Basadas en Componentes. Los arquitectos de software que apliquen este método podrán determinar si la arquitectura sugerida por el conjunto de componentes integrados en un sistema, promueven o no características como confidencialidad, seguridad y mantenibilidad, entre otras. [42]

1.5.1.1 Objetivos trazados por el método.

Es un método que se enfoca en evaluar y analizar la calidad exigida por los usuarios de Arquitectura de Software Basadas en Componentes. Adapta diferentes elementos de algunos métodos de evaluación arquitectónica (ATAM, Bosch, ABD, ARID, Losavio) y establece un conjunto de pasos para determinar la calidad de los sistemas de software basados en componentes. Incluye orientaciones para generar y discutir escenarios iniciales a evaluar, y un conjunto de preguntas a partir de las cuales estudiar las decisiones arquitectónicas consideradas. [42]

1.5.1.2 Fases del método

Fase 1: En la primera fase (presentación), se da a conocer el método entre todos los grupos, el sistema y su organización, y finalmente se presenta cuál es la arquitectura que se evaluará.

Fase 2: La segunda fase es la de investigación y análisis, y en ella se determina de qué manera se va a estudiar la arquitectura, se pide a los stakeholders[1] que expresen de una forma precisa los escenarios de calidad que se deseen y se analiza si la arquitectura es apropiada o se requieren modificaciones para que lo sea. En esta fase solo participa el grupo evaluador y grupo de arquitectos.

Fase 3: La tercera fase es de prueba, consiste en la revisión de la segunda fase y en ella participan todos los grupos.

[1] Aquellas personas que están relacionadas, de cierta forma, con el sistema, ya sea un usuario, un desarrollador o gerente, etcétera.

Fase 4: En la última fase se lleva a cabo la presentación de los resultados, participan todos los grupos.

1.5.1.3 Equipo de evaluación

Grupos participantes del método Mecabic		
Equipo	Definición	Fases en las que participan
Arquitectos	Responsables de generar y documentar una arquitectura de software (AS) para el sistema estudiado.	Todas
Evaluador	Integrado por personas expertas en asuntos de calidad quienes guiarán el proceso de evaluación de la arquitectura.	Todas
Relacionados	Son las personas involucradas de alguna manera con el sistema: programadores, usuarios, gerentes, entre otros.	Fases 1, 3 y 4

Tabla 1. Equipo de evaluación.

1.5.1.4 Instrumentos y técnicas de evaluación

Para la especificación de la calidad se hace uso de un árbol de utilidad, el cual tiene como nodo raíz la "bondad" o "utilidad" del sistema. En el segundo nivel del árbol se encuentran los atributos de calidad. Las hojas del árbol de utilidad son escenarios, los cuales representan mecanismos mediante los cuales extensas (y ambiguas) declaraciones de cualidades son hechas específicas y posibles de evaluar. La generación del árbol de calidad incluye un paso que permite establecer prioridades. Para el análisis de la arquitectura se utiliza la técnica de escenarios, soportada por cuestionarios, para identificar lo que desean los participantes. [42]

1.5.1.5 Atributos de calidad que evalúa

- Funcionalidad: Interoperabilidad, Precisión, Seguridad, Obediencia.
- Fiabilidad: Madurez, Tolerancia a fallas, Capacidad de restablecimiento o Recuperación.
- Eficiencia: Tiempo de comportamiento, Recursos utilizados.
- Mantenibilidad: Habilidad de cambio, estabilidad, prueba.
- Portabilidad: Adaptabilidad, Capacidad de instalación, Co-existencia.

1.5.2 Método de Análisis de Arquitectura de Software (SAAM)

Según Kazman ET AL. (2001), el método de Análisis de Arquitectura de Software (SAAM, por sus siglas en inglés) es el primero que fue ampliamente promulgado y documentado. El método fue originalmente creado para el análisis de la modificación de una arquitectura, pero en la práctica ha demostrado ser muy útil para evaluar de forma rápida distintos atributos de calidad, tales como modificabilidad, portabilidad, escalabilidad e integrabilidad, así como el cubrimiento funcional que tiene la arquitectura sobre los requerimientos del sistema. [25]

Este método puede ser utilizado para evaluar aspectos más ligados con la arquitectura como desempeño o confiabilidad. Está enfocado en la enumeración de un conjunto de escenarios que representan los cambios probables a los que estará sometido el sistema en el futuro. Como entrada principal, es necesaria alguna forma de descripción de la arquitectura a ser evaluada. De acuerdo con Kazman et al. (2001), las salidas de la evaluación del método son las siguientes:

Una proyección sobre la arquitectura de los escenarios que representan los posibles cambios ante los que puede estar expuesto el sistema, e incluso una comparación de múltiples arquitecturas con respecto al nivel de funcionalidad que cada una soporta sin modificación. Con la aplicación de este método, si el objetivo de la evaluación es una sola arquitectura, se obtienen los lugares en los que la misma puede fallar, en términos de los requerimientos de modificabilidad. Para el caso en el que se cuenta con varias arquitecturas candidatas, el método produce una escala relativa que permite observar qué opción satisface mejor los requerimientos de calidad con la menor cantidad de modificaciones.

1.5.2.1 Utilidad

Puede ser utilizado para evaluar uno o múltiples arquitecturas. Si se comparan dos o más se culmina el análisis con una tabla indicando las fortalezas y debilidades de cada una en cada escenario. Si se evalúa una sola, se culmina con un reporte señalando los componentes computacionales donde la arquitectura no alcanza el nivel requerido.

En ningún caso se emite un valor absoluto acerca de la calidad arquitectónica. Utiliza el agrupamiento de escenarios como criterio para evaluar la arquitectura. Esto significa que si se agrupa un conjunto de escenarios por ser similares y luego se observa que son equivalentes, la agrupación ha sido exitosa, porque significa que la funcionalidad del sistema ha sido modularizada adecuadamente.

Por lo contrario, si la asignación de estos escenarios similares afecta diferentes componentes (no son equivalentes), la arquitectura posiblemente debe ser corregida.

1.5.2.2 Pasos

Pasos del método SAAM	
1	Desarrollo de escenarios: La meta principal es capturar las principales actividades que el sistema debe soportar.
2	Descripción de la arquitectura: En este paso se presentan las arquitecturas candidatas.
3	Clasificación de escenarios: En este paso debemos clasificar los escenarios.
4	Evaluación de escenarios: Se debe listar los cambios necesarios en la arquitectura para soportarlo, y el costo de llevarlos a cabo debe ser estimado.
5	Interacción de escenarios: Se debe determinar las interacciones de escenarios sobre cada componente de cada arquitectura.
6	Evaluación general: Un peso asignado a cada escenario en término de su influencia para que el sistema sea exitoso. El peso puede ser elegido de acuerdo a los objetivos del negocio, costo, riesgo, etc.

Tabla 2. Pasos del método SAAM.

1.5.2.3 Principales fortalezas

- Los interesados entienden en profundidad la arquitectura o arquitecturas que se analizan.
- En algunos casos, luego de la sesión de evaluación de SAAM, la documentación relacionada con la arquitectura del software es mejorada.
- Aumenta la comunicación entre los interesados.
- Con respecto a la modificabilidad, los escenarios correspondientes a futuros cambios pueden ser evaluados en la arquitectura o arquitecturas candidatas, logrando estudiar e identificar las áreas potencialmente complejas.
- El esfuerzo y costo de los cambios mencionados pueden ser estimados con anticipación.

1.5.2.4 Principales debilidades

- El proceso para generar los escenarios está basado en la visión de los interesados. En general, existe un muy pequeño esfuerzo para imaginar los escenarios indirectos.
- No provee una métrica clara sobre la calidad de la arquitectura analizada.
- La descripción de la arquitectura puede resultar confusa a la hora de realizar comparaciones si no se adopta una notación y un nivel común de detalles.
- El equipo de evaluación confía en la experiencia de los arquitectos para identificar arquitecturas candidatas. En general, este equipo no tiene conocimiento sobre el conjunto completo de requerimientos y los objetivos del negocio.

1.5.3 Bosch

Bosch (2000) plantea, en su método de diseño de arquitectura de software, que el proceso de evaluación debe ser visto como una actividad iterativa, que forma parte del proceso de diseño, también iterativo. Una vez que la arquitectura es evaluada, pasa a una fase de transformación, asumiendo que no satisface todos los requerimientos. Afirma que la evaluación de una arquitectura de software es una tarea no trivial, puesto que se pretende medir propiedades del sistema en base a especificaciones abstractas, como por ejemplo los diseños arquitectónicos. Por ello, la intención es más bien la evaluación del potencial de la arquitectura diseñada para alcanzar los atributos de calidad requeridos. [8][30]

1.5.3.1 Actividades

1. Analizar los requerimientos funcionales y no funcionales principales del sistema, para establecer las metas de calidad.

2. Utilizar el método de calidad ISO/IEC 9126 adaptado para arquitecturas de software. Algunas métricas pueden definirse con mayor nivel de detalles.

3. Presentar las arquitecturas candidatas iniciales.

4. Construir la tabla comparativa para las arquitecturas candidatas.

5. Establecer propiedades para las subcaracterística de calidad tomando en cuenta los requerimientos de calidad del sistema.

6. Analizar los resultados obtenidos, de acuerdo con las propiedades establecidas.

Bosch menciona tres objetivos para realizar mediciones sobre una arquitectura de software: cualitativos, cuantitativos y máximos y mínimos teóricos.

La medición cualitativa*:* Se aplica para la comparación entre arquitecturas candidatas y tiene relación con

la intención de saber la opción que se adapta mejor a cierto atributo de calidad. Este tipo de medición brinda respuestas afirmativas o negativas, sin mayor nivel de detalle.

La medición cuantitativa: *B*usca la obtención de valores que permitan tomar decisiones en cuanto a los atributos de calidad de una arquitectura de software. El esquema general es la comparación con márgenes establecidos, como lo es el caso de los requerimientos de desempeño, para establecer el grado de cumplimiento de una arquitectura candidata, o tomar decisiones sobre ella. Este enfoque permite establecer comparaciones, pero se ve limitado en tanto no se conozcan los valores teóricos máximos y mínimos de las mediciones con las que se realiza la comparación.

La medición de máximo y mínimo teórico contempla los valores teóricos para efectos de la comparación de la medición con los atributos de calidad especificados. El conocimiento de los valores máximos o mínimos permite el establecimiento claro del grado de cumplimiento de los atributos de calidad. De esta forma presenta los objetivos para efectos de la medición de los atributos de calidad.

1.5.3.2 Técnicas de evaluación

Bosch propone cuatro técnicas de evaluación de arquitecturas de software: evaluación basada en escenarios, evaluación basada en simulación, evaluación basada en modelos matemáticos y evaluación basada en experiencia.

Evaluación basada en escenarios (Kazman (2001):

Escenario: Según Kazman et al. (2001), es una breve descripción de la interacción de alguno de los involucrados en el desarrollo del sistema con este.

Según Kazman et al. (2001), un escenario consta de tres partes: el estímulo, el ambiente y la respuesta. El estímulo es la parte del escenario que explica o describe lo que el involucrado en el desarrollo hace para iniciar la interacción con el sistema. Puede incluir ejecución de tareas, cambios en el sistema, ejecución de pruebas, configuración, etc. El ambiente describe qué sucede en el sistema al momento del estímulo. La respuesta describe, a través de la arquitectura, cómo debería responder el sistema ante el

estímulo. Este último elemento es el que permite establecer cuál es el atributo de calidad asociado. Entre las ventajas de su uso están:

- Son simples de crear y entender.
- Son poco costosos y no requieren mucho entrenamiento.
- Son efectivos.

Técnica basada en escenarios propuesta por Bosch: Perfiles:

Un perfil es un conjunto de escenarios, generalmente con alguna importancia relativa asociada a cada uno de ellos. El uso de perfiles permite hacer especificaciones más precisas del requerimiento para un atributo de calidad (Bosch 2000). Los perfiles tienen asociados dos formas de especificación: perfiles completos y perfiles seleccionados.

- Perfiles completos**:** Definen todos los escenarios relevantes como parte del perfil. Esto permite al ingeniero de software realizar un análisis de la arquitectura para el atributo de calidad estudiado de una manera completa, puesto que incluye todos los posibles casos. Su uso se reduce a sistemas relativamente pequeños y sólo es posible predecir conjuntos de escenarios completos para algunos atributos de calidad (Bosch, 2000).

- Perfiles seleccionados: Se asemejan a la selección de muestras sobre una población en los experimentos estadísticos. Se toma un conjunto de escenarios de forma aleatoria, de acuerdo a algunos requerimientos. La aleatoriedad no es totalmente cierta por limitaciones prácticas, por lo que se fuerza la realización de una selección estructurada de elementos para el conjunto de muestra. Si bien es informal, permite hacer proposiciones científicamente válidas (Bosch, 2000).

Dado que los escenarios se construyen cuidadosamente, Bosch (2000) plantea que puede asumirse que el perfil representa una imagen exacta de la población de escenarios. Para la definición de un perfil, es

necesario seguir tres pasos: definición de las categorías del escenario, selección y definición de los escenarios para la categoría asignación del “peso” a los escenarios.
Se establece que la definición de categorías de escenarios divide la población de escenarios en poblaciones más pequeñas, que cubren aspectos particulares del sistema. La selección y definición de escenarios para cada categoría selecciona un conjunto de escenarios representativo para la subpoblación. Luego, en la asignación del peso a los escenarios, dependiendo del perfil, el peso de un escenario tiene diferentes significados. Se definen escalas que de alguna forma sean cuantificables y puedan ser convertidas a pesos relativos.
Cada atributo de calidad tiene un perfil asociado. Algunos perfiles pueden ser usados para evaluar más de un atributo de calidad. La evaluación basada en escenarios puede ser empleada para comparar dos arquitecturas o para la evaluación absoluta de la misma. La diferencia está en que la evaluación absoluta requiere mayor cantidad de datos estimados y cuantitativos necesarios para la evaluación.

Evaluación basada en simulación:

Establece que la evaluación basada en simulación utiliza una implementación de alto nivel de la arquitectura de software. El enfoque básico consiste en la implementación de componentes de la arquitectura y la implementación a cierto nivel de abstracción del ambiente del sistema donde se supone va a ejecutarse. La finalidad es evaluar el comportamiento de la arquitectura bajo diversas circunstancias.

Una vez disponibles estas implementaciones, pueden usarse los perfiles respectivos para evaluar los atributos de calidad. El proceso de evaluación basada en simulación sigue los siguientes pasos (Bosch, 2000):

- Definición e implementación del contexto. Consiste en identificar las interfaces de la arquitectura de software con su contexto, y decidir cómo será simulado el comportamiento del contexto en tales interfaces.

- Implementación de los componentes arquitectónicos. La descripción del diseño arquitectónico debe definir, por lo menos, las interfaces y las conexiones de los componentes. El comportamiento de los componentes en respuesta a eventos sobre sus interfaces puede no ser especificado

claramente, aunque generalmente existe un conocimiento común y es necesario que el arquitecto lo interprete, por lo que éste decide el nivel de detalle de la implementación.

- Implementación del perfil. Dependiendo del atributo de calidad que el arquitecto de software intenta evaluar usando simulación, el perfil asociado necesitará ser implementado en el sistema. El arquitecto de software debe ser capaz de activar escenarios individuales, así como también ejecutar un perfil completo usando selección aleatoria, basado en los pesos normalizados de los mismos.

- Simulación del sistema e inicio del perfil. El arquitecto de software ejecutará la simulación y activará escenarios de forma manual o automática, y obtendrá resultados de acuerdo al atributo de calidad que está siendo evaluado.

- Predicción de atributos de calidad. Dependiendo del tipo de simulación y del atributo de calidad evaluado, se puede disponer de cantidades excesivas de datos, que requieren ser condensados. Esto permite hacer conclusiones acerca del comportamiento del sistema.

En el ámbito de las simulaciones, se encuentra la técnica de implementación de prototipos. Esta técnica implementa una parte de la arquitectura de software y la ejecuta en el contexto del sistema. Es utilizada para evaluar requerimientos de calidad operacional, como desempeño y confiabilidad. Para su uso se necesita mayor información sobre el desarrollo y disponibilidad del hardware, y otras partes que constituyen el contexto del sistema de software. Se obtiene un resultado con mayor exactitud (Bosch, 2000). La exactitud de los resultados de la evaluación depende, a su vez, de la exactitud del perfil utilizado para evaluar el atributo de calidad y de la precisión con la que el contexto del sistema simula las condiciones del mundo real.

En términos de los instrumentos asociados a las técnicas de evaluación basadas en simulación, se encuentran los lenguajes de descripción arquitectónica, y los modelos de colas.

Evaluación basada en modelos matemáticos:

Establece que la evaluación basada en modelos matemáticos se utiliza para evaluar atributos de calidad

operacionales. Permite una evaluación estática de los modelos de diseño arquitectónico, y se presentan como alternativa a la simulación, dado que evalúan el mismo tipo de atributos. Ambos enfoques pueden ser combinados, utilizando los resultados de uno como entrada para el otro. El proceso de evaluación basada en modelos matemáticos sigue los siguientes pasos (Bosch, 2000):

- Selección y adaptación del modelo matemático. En la mayoría de las investigaciones orientadas a los atributos de calidad se han desarrollado modelos matemáticos para medir sus atributos de calidad, los cuales tienden a ser muy elaborados y detallados, así como también requieren de cierto tipo de datos y análisis. Parte de estos datos requeridos no están disponibles a nivel de arquitectura, y la técnica requiere mucho esfuerzo para la evaluación arquitectónica, por lo que el arquitecto de software se ve obligado a adaptar el modelo.

- Representación de la arquitectura en términos del modelo. El modelo matemático seleccionado y adaptado no asume necesariamente que el sistema que intenta modelar consiste de componentes y conexiones. Por lo tanto, la arquitectura necesita ser representada en términos del modelo.

- Estimación de los datos de entrada requeridos. El modelo matemático aún cuando ha sido adaptado, requiere datos de entrada que no están incluidos en la definición básica de la arquitectura. Es necesario estimar y deducir estos datos de la especificación de requerimientos y de la arquitectura diseñada.

- Predicción de atributos de calidad. Una vez que la arquitectura es expresada en términos del modelo y se encuentran disponibles todos los datos de entrada requeridos, el arquitecto está en capacidad de calcular la predicción resultante del atributo de calidad evaluado.

Entre las desventajas que presenta esta técnica se encuentra la inexistencia de modelos matemáticos apropiados para los atributos de calidad relevantes (Bosch, 2000), y el hecho de que el desarrollo de un modelo de simulación completo puede requerir esfuerzos sustanciales. Entre los instrumentos que se cuentan para las técnicas de evaluación de arquitecturas de software basada en modelos matemáticos, se

encuentran las Cadenas de Markov y los Reliability Block Diagrams.

Evaluación basada en experiencia:

Existen dos tipos de evaluación basada en experiencia: la evaluación informal, que es realizada por los arquitectos de software durante el proceso de diseño, y la realizada por equipos externos de evaluación de arquitecturas.

1.5.3.3 Atributos de calidad que evalúa Mecabic

Tales atributos son: desempeño, mantenibilidad, confiabilidad, seguridad externa y seguridad interna.

1.5.4 Método de Análisis de Acuerdos de Arquitectura (ATAM)

ATAM (por sus siglas en inglés) obtiene su nombre no solo porque nos dice cuán bien una arquitectura particular satisface las metas de calidad, sino porque también provee ideas de cómo esas metas de calidad interactúan entre ellas, cómo realizan concesiones mutuas entre ellas. Este método se encuentra inspirado en tres áreas distintas: los estilos arquitectónicos, el análisis de los atributos de calidad y el método de evaluación SAAM. Se concentra en la identificación de los estilos arquitectónicos o enfoques arquitectónicos utilizados. [22][27]

Kazman et al. (2001) proponen el término enfoque arquitectónico dado que no todos los arquitectos están familiarizados con el lenguaje de estilos arquitectónicos, aún haciendo uso indirecto de éstos. De cualquier forma, éstos elementos representan los medios empleados por la arquitectura para alcanzar los atributos de calidad, así como también permiten describir la forma en la que el sistema puede crecer, responder a cambios, e integrarse con otros sistemas, entre otros (Kazman et al., 2001). El método de evaluación ATAM comprende nueve pasos, agrupados en cuatro fases. [26][27][40]

1.5.4.1 Fases y etapas del método

Fase 0: Presentación

1- Presentar ATAM. El método se describe a las partes interesadas (normalmente los representantes de los clientes, el arquitecto o el equipo de arquitectura, representantes de los usuarios, mantenedores, administradores, gerentes, probadores, integradores y otros).

2- Presentar los objetivos del negocio. El líder del proyecto describe los objetivos que motivan el esfuerzo de desarrollo y, por tanto, lo que será la arquitectura primaria (por ejemplo, alta disponibilidad, tiempo de salida al mercado o alta seguridad, requerimientos funcionales de alto nivel y requerimientos de atributos de calidad).

3- Presentar la arquitectura. El arquitecto presenta un panorama general de la arquitectura. Pueden ser las restricciones técnicas, como el sistema operativo, hardware, software o medios previstos para su uso; así como otros sistemas con los que el sistema debe interactuar. También se mencionan los enfoques arquitectónicos utilizados para hacer frente a los requisitos de los atributos de calidad que persigue la arquitectura, estableciendo una descripción de cómo los enfoques intentan obtenerlos.

Fase 1: Investigación y Análisis:

4- Identificar los enfoques arquitectónicos. Comienzan a identificar los enfoques arquitectónicos que son fundamentales para la realización de los atributos de calidad objetivos.

5- Generar el árbol de utilidad de los atributos de calidad. Identificar, priorizar y refinar los atributos de calidad más importantes para el sistema, los cuales son expresados en forma de escenarios, priorizados y analizados.

- Un árbol de utilidad es un vehículo para el manejo de los requisitos de atributos de calidad específicos.
- Selecciona los objetivos de calidad para ser los nodos de alto nivel (por lo general, el desempeño, modificabilidad, seguridad y disponibilidad).
- Los atributos de calidad pueden ser refinados tanto como se necesite.

- Los escenarios son las hojas del árbol de utilidad. La salida de un árbol de utilidad es la caracterización y priorización de los requisitos de atributos de calidad específicos.

La importancia para el éxito del sistema es Alta / Media / Baja; la dificultad para alcanzar el objetivo es Alta / Media / Baja, a evaluación del arquitecto.

6- Analizar los enfoques arquitectónicos. El equipo de evaluación identifica los enfoques arquitectónicos desde el punto de vista de los atributos de calidad específicos para identificar los riesgos. Generar preguntas para los atributos de calidad de mayor prioridad (por ejemplo, un enfoque arquitectónico destinado a satisfacer objetivos de rendimiento será sometido a un análisis del rendimiento). Identificar “riesgos”, “no riesgos”, “puntos sensibles” y “puntos de acuerdos (tradeoff)”

Fase 2: Pruebas

7- Lluvia de ideas y priorización de escenarios. Los interesados generan los escenarios para facilitar el trabajo, en la realización de una lluvia de ideas obteniendo un conjunto más amplio de los escenarios. A este conjunto de escenarios se le da prioridad a través de un proceso de votación de la totalidad de los participantes del grupo de interesados. Se agregan los nuevos escenarios al árbol de utilidad.

8- Analizar enfoques arquitectónicos. En esta etapa se realizan las mismas actividades que en la etapa 6 sobre los nuevos escenarios generados en la tormenta de ideas.

Fase 3: Reporte

9- Presentar los resultados. Consiste en un reporte escrito donde se presenta la información obtenida de la evaluación (estilos, escenarios, árbol de utilidad, riesgos, puntos de tradeoff (desacuerdos), puntos sensitivos).

1.5.4.2 Duración de las fases del método

Fase	Duración

Presentación	Normalmente dura algunas semanas
Investigación y análisis	Un día seguido de dos a tres semanas de descanso
Prueba	Dos días
Reporte	Una semana

Tiempo de duración de las fases.

1.5.4.3 Artefactos que genera

Escenarios de Alta Prioridad (High-Priority Scenarios), Preguntas para Atributos Específicos (Attribute-Specific Questions), Enfoques Arquitectónicos (Architectural Approaches), Riesgos (Risk), No Riesgos (No Risk), Árbol de Utilidad (Utility Tree), Puntos Sensitivos y Puntos de Acuerdos (Sensitive and Trade off Points), Estructura para Razonamiento (Structure for Reasoning).(22)

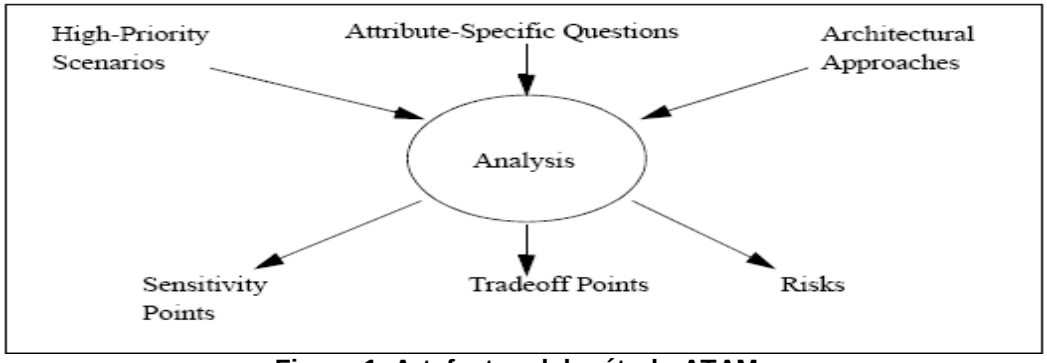


Figura 1. Artefactos del método ATAM.

1.5.4.4 Atributos de calidad que evalúa

Normalmente sus atributos de calidad son desempeño, seguridad, modificabilidad y disponibilidad (propuestos por Kazman), ya que normalmente son los que definen la usabilidad de una arquitectura, pero los involucrados (stakeholders) en la evaluación por mutuo acuerdo son libres de seleccionarlos.

1.6 Comparación entre los métodos de evaluación

Comparación de los métodos de evaluación				
Aspectos a comparar	ATAM	SAAM	MECABIC	BOSCH
Atributos de calidad que evalúa	Modificabilidad Seguridad Confiabilidad Desempeño	Modificabilidad Funcionalidad	Funcionalidad Fiabilidad Eficiencia Mantenibilidad Portabilidad	Desempeño Mantenibilidad Confiabilidad Seguridad interna y externa
Objetos analizados	Estilos arquitectónicos, Documentación, Flujo de datos, Vistas arquitectónicas	Documentación, Vistas arquitectónicas	Documentación, Vistas arquitectónicas	Vistas arquitectónicas, Enfoques arquitectónicos
Etapa del proyecto en la que se aplica	Luego de que el diseño de la arquitectura ha sido establecido.	Luego de que la arquitectura cuenta con funcionalidad ubicada en módulos.	Luego de que el diseño de la arquitectura ha sido establecido.	Luego de que el diseño de la arquitectura ha sido establecido.
Elementos utilizados	Árbol de utilidad y lluvia de ideas para articular los requerimientos de calidad. Análisis arquitectónico que detecta puntos sensibles, puntos de balance y riesgos.	Lluvia de ideas para escenarios y articular los requerimientos de calidad. Análisis de los escenarios para verificar funcionalidad o estimar el costo de los cambios.	Analizar la calidad exigida por los usuarios de arquitectura de software basadas en componentes para evaluar arquitectura. Incluye orientaciones para generar y discutir escenarios iniciales a evaluar.	Análisis de perfiles

Tabla 3. Comparación entre los diferentes métodos de evaluación.

Producto de un análisis sobre la tabla anterior se concluye que el método propuesto para efectuar la evaluación de la arquitectura del proyecto Modernización del Sistema Bancario Cubano es ATAM, debido a que se puede aplicar en el estado actual que presenta la arquitectura del proyecto, los atributos de calidad que propone inicialmente resumen la utilidad del sistema, incluyendo la seguridad, siendo esta un objetivo fundamental en la arquitectura propuesta. El método SAAM no puede aplicarse porque se efectúa cuando la arquitectura presenta cierto nivel de implantación, y la propuesta se encuentra especificada en estos momentos a un alto nivel. MECABIC es un método orientado a componentes, es decir, para evaluar arquitecturas de ese tipo, no siendo este el caso de la propuesta. Bosch es un método que puede

aplicarse a esta arquitectura; pero los atributos de calidad propuestos por el método ATAM se asemejan más (con respecto a la arquitectura evaluada) que los propuestos por Bosch.

1.7 Conclusiones

En el presente capitulo, a partir de la investigación realizada, se definieron los principales modelos de calidad existentes, se describieron las fases de los métodos de evaluación estudiados y sus características, posibilitando realizar una comparación entre ellos, permitiendo definir el método que se aplicará en la evaluación de la arquitectura del proyecto Modernización del Sistema Bancario Cubano.

Capítulo 2: Procedimiento de evaluación de la arquitectura de software del proyecto Modernización del Sistema Bancario Cubano.

2.1 Introducción

En este capítulo se describe el procedimiento propuesto para evaluar la arquitectura de software del proyecto Modernización del Sistema Bancario Cubano, explicando las actividades realizadas para llevar a cabo la evaluación, técnicas y herramientas utilizadas, los roles involucrado en la evaluación así como los artefactos generados.

2.2 ¿Cuándo evaluar con ATAM?

ATAM se aplica cuando la arquitectura presenta un alcance definido y manejable, ya que el equipo evaluador necesita conocer los objetivos del negocio, las decisiones arquitectónicas tomadas para lograr los atributos de calidad especificados, vistas de componentes y conectores, etcétera. Entre las precondiciones para aplicarlo se encuentran: [21]

- Los miembros del equipo de revisión analizan los artefactos arquitectónicos y ayudan a mejorar la documentación. El arquitecto prepara una arquitectura de presentación.
- Los clientes preparan una presentación de los objetivos del negocio.

El equipo de evaluación revisa los artefactos de la arquitectura, presentaciones y lee el material antes para familiarizarse con el dominio. [26]

2.3 Actividades para evaluar la arquitectura

En el transcurso del procedimiento de evaluación de la arquitectura propuesta se hace necesario realizar un conjunto de actividades que permitan lograr una correcta materialización da la misma, se generan artefactos, se utilizan técnicas y herramientas en su desarrollo; cada uno de los miembros del equipo de evaluación juega un rol específico dentro de él, con el objetivo de lograr la mayor calidad posible en el trabajo que se realiza. [26][40]

2.3.1 Presentar la conducción del negocio

Esta presentación es importante para entender el contexto del sistema y los conductores primarios del negocio. Se necesita tener una panorámica de las perspectivas del mismo, por lo que se describen las metas del negocio, las funciones más importantes del sistema, cualquier técnica relevante, económica o contratos contraídos y conductores arquitectónicos (principales objetivos de los atributos de calidad que forman la arquitectura).

2.3.2 Presentar la arquitectura

En esta actividad se hace necesario una descripción en un apropiado nivel de detalle, el cual depende de diferentes factores, como pueden ser: qué tan documentada y diseñada se encuentra la arquitectura, cuánto tiempo está disponible, su naturaleza de comportamiento y requisitos de calidad. El arquitecto menciona las técnicas contraídas, como hardware, sistema operativo y otros sistemas con los que debe interactuar. Se describen los enfoques arquitectónicos (o patrones) usados para llegar a los requerimientos.

2.3.3 Identificar los enfoques arquitectónicos

Desarrollar esta tarea permite saber qué patrones y enfoques arquitectónicos se usaron en el diseño del sistema. Simplemente se toman los patrones y enfoques que son evidentes, dejando sentadas las bases para análisis posteriores.

2.3.4 Generar el árbol de utilidad de los atributos de calidad

Su confección es de gran importancia para identificar, priorizar y refinar los más importantes objetivos de los atributos de calidad del sistema, los cuales son expresados como escenarios. Posibilita hacer los requerimientos completos, forzando al arquitecto y al cliente a definir precisamente los requerimientos de calidad relevantes. Se anotan los estímulos y respuestas y se establece la prioridad entre los atributos de calidad.

2.3.5 Analizar los enfoques arquitectónicos

En este análisis se examinan los principales escenarios uno a uno y se evalúa cómo la arquitectura soporta a cada uno de ellos. En esta etapa se documentan las decisiones arquitectónicas y se identifican y catalogan sus riesgos, no riesgos, puntos sensitivos y puntos de acuerdos entre los atributos de calidad, realizando una lista de ellos.

2.3.6 Presentar los resultados

En esta etapa se presenta la información que necesitó ATAM para su ejecución, presentándose las salidas producidas producto de la aplicación de la evaluación como son: los enfoques arquitectónicos, los escenarios y sus priorizaciones, el árbol de utilidad, los riesgos, los puntos de sensibilidad y los puntos de acuerdos.

2.4 Técnicas y herramientas utilizadas

Todas las tareas mencionadas anteriormente se efectúan bajo las condiciones del método de evaluación ATAM, el cual se aplica para evaluar la arquitectura propuesta.

2.5 Especificación de roles, características y habilidades

Para el desarrollo de ATAM se requiere de la participación y mutua cooperación de tres grupos:

2.5.1 Equipo evaluador

Este grupo es externo al proyecto al cual se le realizará la evaluación de la arquitectura, cada miembro juega un número determinado de roles durante la evaluación.

2.5.2 Formuladores de decisiones del proyecto

Estas personas están autorizadas a hablar por el desarrollo del proyecto y de autorizar cambios en el mismo. Usualmente se incluye al jefe del proyecto y el arquitecto siempre es incluido.

2.5.3 Interesados en la arquitectura

Tienen los intereses creados en el desempeño de la arquitectura como anunciadores. Su trabajo durante una evaluación es articular los objetivos específicos de los atributos de calidad que la arquitectura podría conocer y que para el sistema son considerados un éxito.

2.6 Los roles del equipo de evaluación de ATAM

En este procedimiento existen varias personas involucradas y cada una juega un rol determinado dentro del desarrollo del proyecto, a continuación se describen cada uno de estos roles.

2.6.1 Líder del equipo

Tiene la responsabilidad de establecer la evaluación, coordinar con el cliente, asegurar que sus necesidades sean conocidas, establecer el contrato de evaluación así como formar el equipo, ver que el reporte final es producido y deliberado. Debe ser una persona organizada, con habilidades de dirección y hábil interactuando con el cliente.

2.6.2 Líder de evaluación

Es el encargado de ejecutar la evaluación, facilita la elicitación de escenarios, administra el proceso de selección\priorización de escenarios, facilita la evaluación de escenarios junto a la arquitectura. Debe ser una persona de facilidad de palabras frente a la audiencia, con excelentes habilidades, buen entendedor en cuestiones arquitectónicas y práctico en evaluaciones de arquitecturas.

2.6.3 Documentador de escenarios

Su función es escribir los escenarios durante su elicitación, capturando los acuerdos hablados en cada escenario. Debe ser hábil escribiendo a mano y debe conocer la esencia de las discusiones técnicas.

2.6.4 Documentador de procedimientos

Su función es capturar los procedimientos en formato digital, escenarios, los aspectos que motivan cada escenario y la resolución de cada uno de ellos cuando son aplicados a la arquitectura. Debe ser una

persona con habilidades de mecanografía y buen organizador para rápidas recogidas de información.

2.7 Artefactos generados por ATAM

Durante el desarrollo de la evaluación se generan documentos los cuales son presentados como parte de la última etapa del método ATAM y son los siguientes:

- Documento de enfoque arquitectónico.
- El conjunto de escenarios y su priorización de la tormenta de ideas.
- El árbol de utilidad.
- Los riesgos descubiertos.
- Los no-riesgos documentados.
- Los puntos sensibles y puntos de acuerdos encontrados.

2.8 Métricas de medición de atributos de calidad

Las métricas son estándares de mediciones (fórmulas) que desempeñan un importante papel a la hora de medir el comportamiento de determinado atributo de calidad de una arquitectura. Son usadas cuando la arquitectura se encuentra en un avanzado nivel de especificación. Cuando no se haya logrado implementar como mínimo parte de la arquitectura, resulta imposible su aplicación, ya que no pueden existir datos suficientes que las satisfagan. A continuación se muestra la ISO 9126-3, que particularmente especifica los atributos internos de calidad. [18][19][31][32][35][36][37]

2.8.1 Métricas para medir la Funcionalidad

En este grupo se conjunta una serie de atributos que permiten calificar si un producto de software maneja de forma adecuada determinadas funciones que satisfagan las necesidades para las cuales fue diseñado. Para este propósito se establecen los siguientes atributos:

Métrica Interna de interoperabilidad									
Nombre	**Propósito**	**Método de aplicación**	**Medición, Fórmula y elementos**	**Interpretación del valor medido**	**Tipo de Escala Métrica**	**Tipo de medición**	**Fuente de Entrada**	**ISO/IEC 12207 SLCP**	**Objetivo de Audiencia**

			de datos computacionales				**Medida**	**Referencia**	
Habilidades de intercambio de datos (Base de estructura de datos)	¿Cuán correcto es el formato de la interfaz de datos que se está implementando?	Cuenta el número de interfaz de datos que están implementados y especificados correctamente y se compara con el número de dato del formato de cambio y de especificación.	X=A/B A=Número de interfaz de datos que están implementados y especificados correctamente. B=Número de datos del formato de cambio y su especificación.	0<=X<=1 Más cerca a 1, más completos	Absoluto	X=cuenta /cuenta A=cuenta B=cuenta	Especificación de Diseño del Informe de revisión	Verificación Revisión conjunta	Requiere Reveladores
Consistencia de interfaz (protocolo)	¿Cuán correcta está la interfaz de protocolo que se implementa ?	Cuenta el número de interfaz de protocolos que se implementa correctamente en su especificación y lo compara con el número de interfaz de protocolos de implementación especificados.	X=A/B A=Número de interfaz de protocolos implementados en el formato consistente de especificación confirmado en la revisión. B=Número de interfaz de protocolos implementados en la especificación.	0<=X<=1 Más cerca a 1, más consistente	Absoluto	X=cuenta /cuenta A=cuenta B=cuenta	Especificación de Diseño del Informe de revisión	Verificación Revisión conjunta	Requiere Reveladores

Tabla 4. Métrica interna de interoperabilidad.

Métrica interna de seguridad									
Nombre	**Propósito**	**Método de aplicación**	**Medición, Fórmula y elementos de datos computacionales**	**Interpretación del valor medido**	**Tipo de Escala Métrica**	**Tipo de medición**	**Fuente Entrada Medida**	**ISO/IEC 12207 SLCP Referencia**	**Objetivo Audiencia**
Auditabilidad del acceso	¿Cuán auditable es el acceso login?	Cuenta el número del tipo de accesos y se anota	X=A/B A= número del tipo de accesos especificad	0<=X<=1 Más cerca a 1, más auditable	Absoluto	X=cuenta /cuenta A=cuenta B=cuenta	Especificación de Diseño del Informe	Verificación Revisión conjunta	Requiere Reveladores

		correctamente las características técnicas y compara el número de los tipos de acceso requeridos en las características especificadas.	os B= número del tipo de accesos requeridos en las características especificadas				de revisión		
Controlabilidad del acceso	¿Cuán controlables el acceso al sistema?	Cuenta el número de acceso de controlabilidad requerido que fue implementado y especificado correctamente y compara el número de acceso de controlabilidad requerido en la especificación.	X=A/B A= número de acceso de controlabilidad requerido que fue implementado y especificado correctamente B= número de acceso de controlabilidad requerido en la especificación	0<=X<=1 Más cerca a 1, más controlable	Absoluto	X=cuenta/cuenta A=cuenta B=cuenta	Especificación de Diseño del Informe de revisión	Verificación Revisión conjunta	Requiere Reveladores
Prevención de la corrupción del datos	¿Cuán completa es la implementación de la prevención de la corrupción de los datos?	Cuenta en número de implementaciones intanciadas en la prevención de la corrupción de los datos especificados y compara con el número de interfaz de operaciones/acceso en los requerimientos especificado capaz de alterar/destruir datos.	X=A/B A=Número de interfaz de implementación especificado para la prevención de la corrupción de datos revisados. B= el número de interfaz de operaciones/acceso en los requerimientos capaz de alterar/destruir datos.	0<=X<=1 Más cerca a 1, más completos	Absoluto	X=cuenta/cuenta A=cuenta B=cuenta	Especificación de Diseño del Informe de revisión	Verificación Revisión conjunta	Diseñadores

Nota de página Considerar los niveles de seguridad usados para las métricas.									
Encriptación de datos	¿Cuán completa es la implementación de la encriptación de los datos?	Cuenta el número de interfaces implementadas encriptados/desencintadas en la especificación de los datos y compara con el número de artículos de datos requeridos encriptación de datos/facilidad de desencriptación como en características técnicas.	X=A/B A= número de interfaces implementadas encriptados/desencintadas en la especificación de los datos revisados. B= el número de artículos de datos requeridos encriptación de datos /facilidad de desencriptación especificados.	0<=X<=1 Más cerca a 1, más completos	Absoluto	X=cuenta /cuenta A=cuenta B=cuenta	Especificación de Diseño del Informe de revisión	Validación	Diseñadores
Nota de página Encriptación de datos, base de datos abiertos, datos en facilidad de comunicación pública.									

Tabla 5. Métrica interna de seguridad.

2.8.2 Métricas para medir la Usabilidad

La capacidad del producto de software para ser atractivo, entendido, aprendido, y utilizado por el usuario bajo condiciones específicas. Consiste de un conjunto de atributos que permiten evaluar el esfuerzo necesario que deberá invertir el usuario para utilizar el sistema.

Métrica interna de comprensibilidad									
Nombre	**Propósito**	**Método de aplicación**	**Medición, Fórmula y elementos de datos computacionales**	**Interpretación del valor medido**	**Tipo de Escala Métrica**	**Tipo de medición**	**Fuente de Entrada Medida**	**ISO/IEC 12207 SLCP Referencia**	**Objetivo de Audiencia**
Descripción de integridad	¿Qué proporción de funciones (o tipos de función) se describe en la	Cuenta el número de funciones que se describen adecuadamente y las compara con el número	X=A/B A= Número de funciones (o tipos de funciones) descritas en la	0<=X<=1 Más cerca a 1, más completos	Absoluto	X=cuenta /cuenta A=cuenta B=cuenta	Especificación de Diseño del Informe de revisión	Verificación Revisión conjunta	Requiere Reveladores

	descripción del producto?	total de funciones del producto.	descripción del producto B= Número del total de funciones (o tipos de funciones)						
Nota de página 1) Éste indica que usuarios potenciales entenderán la capacidad del producto después de lectura la descripción del producto. 2) Ve también ISO/ IEC 9127 software del Consumidor de empaquetamiento.									
Capacidad de la demostración	¿Qué proporción de demostración de capacidad contienen las funciones de demostración de los requerimientos?	Cuenta el número de funciones que están demostradas adecuadamente y las compara con el número total de funciones de capacidad que requiere la demostración.	X=A/B A= Número de funciones demostradas y confirmadas en la revisión. B= Número del total de funciones de capacidad requeridas en la demostración	0<=X<=1 Más cerca a 1, más capaz	Absoluto	X=cuenta /cuenta A=cuenta B=cuenta	Especificación de Diseño del Informe de revisión	Verificación Revisión conjunta	Requiere Reveladores
Nota de página Demostración de los pasos para la exhibición del proceso cómo se usa el producto. La inclusión de " aplicación que ayuda al usuario a ejecutar una tarea en forma eficaz".									
Funciones evidentes	¿Qué proporción de las funciones del producto es evidente a los usuarios?	Cuenta el número de funciones que son evidentes al usuario y la compara con el número total de funciones.	X=A/B A= Número de funciones (o tipos de funciones) evidente al usuario. B= Número del total de funciones (o tipos de funciones)	0<=X<=1 Más cerca a 1, el mejor	Absoluto	X=cuenta /cuenta A=cuenta B=cuenta	Especificación de Diseño del Informe de revisión	Verificación Revisión conjunta	Requiere Reveladores
Nota de página Éste indica si los usuarios podrán localizar funciones para explorar la interfaz ([e.g], por inspeccionar los menús).									
Comprencibilidad de la función	¿Qué proporción de las funciones del producto podrá el usuario entender correctamente?	El número de funciones del usuario tiene como propósito entender al usuario y compararlo con el número de funciones de la interfaz de usuarios.	X=A/B A=El Número de la interfaz de usuario tiene como propósito el entendimiento de los usuarios. B=Número de	0<=X<=1 Más cerca a 1, el mejor	Absoluto	X=cuenta /cuenta A=cuenta B=cuenta	Especificación de Diseño del Informe de revisión	Verificación Revisión conjunta	Requiere Reveladores

			funciones de usuario.						

Tabla 6. Métrica interna de comprensibilidad.

Métrica interna de aprendizaje									
Nombre	**Propósito**	**Método de aplicación**	**Medición, Fórmula y elementos de datos computacionales**	**Interpretación del valor medido**	**Tipo de Escala Métrica**	**Tipo de medición**	**Fuente de Entrada Medida**	**ISO/IEC 12207 SLCP Referencia**	**Objetivo de Audiencia**
Integridad de documentación del usuario y/ o facilidad de la ayuda	¿Qué proporción de funciones se describe en la documentación del usuario y/ o facilidad de la ayuda?	Cuenta el número de funciones llevadas a cabo con facilidad de la ayuda y/ o documentación y se compara con el número total de funciones del producto.	X=A/B A= Número de funciones descritas. B= Número total de funciones provistas	0<=X<=1 Más cerca a 1, más completo	Absoluto	X=cuenta /cuanta A=cuanta B=cuanta	Especificación de Diseño del Informe de revisión	Verificación de la revisión conjunta	Requiere Reveladores
Nota de página Las métricas son posibles: integridad en las documentaciones, integridad en la facilidad de ayuda o integridad de la ayuda y documentación, uso en la documentación.									

Tabla 7. Métrica interna de aprendizaje.

2.8.3 Métricas para medir la Eficiencia

Es la capacidad del producto de software para mantener un nivel especificado de rendimiento cuando es utilizado bajo condiciones especificadas.

Métrica interna de rendimiento									
Nombre	**Propósito**	**Método de aplicación**	**Medición, Fórmula y elementos de datos computacionales**	**Interpretación del valor medido**	**Tipo de Escala Métrica**	**Tipo de medición**	**Fuente de Entrada Medida**	**ISO/IEC 12207 SLCP Referencia**	**Objetivo de Audiencia**
Tiempo de Respuesta	¿Cuál es el tiempo estimado para completar una determinada tareas?	Evaluar la eficiencia del sistema operativo y la aplicación llamadas del sistema. Estimar el tiempo de respuesta	X=Tiempo (calculado o simulada)	La corta es la mejor.	Relación	X=Tiempo	Saber sistema operativo. Tiempo estimado de llamadas del sistema.	Verificación Revisión conjunta	Requiere Reveladores

		sobre la base de este. La siguiente puede ser medido; -todos o partes de especificaciones de diseño - prueba completa transacción camino -prueba completa modules/partes del producto de software -completa producto de software durante el ensayo fase							
Rendimiento de Tiempo	¿Cuál es el número estimado de las tareas que pueden ser realizadas en una unidad de tiempo?	Evaluaron la eficiencia de manipulación recursos en el sistema. Hacer un factor basado en la aplicación pide al sistema en manipulación recursos.	X=Ninguna de las tareas por unidad de tiempo.	La mayor es la mejor.	Relación	X=Cuenta	Saber sistema operativo. Tiempo estimado de llamadas del sistema.	Verificación Revisión conjunta	Requiere Reveladores
Punto de cambio (lapso entre el principio de la instalación de un programa hasta el recibimiento de una salida)	¿Cuál es el tiempo estimado para completar un grupo de tareas conexas como un lote de empleos?	Evalúan la eficiencia del sistema operativo y la aplicación llamadas del sistema. Estimar el tiempo de respuesta para completar un grupo de tareas conexas sobre la base de este. La siguiente puede ser medido, -todos o partes de especificacio	X=tiempo (calculado o simulada)	La menor es el mejor.	Relación	X=Tiempo	Saber sistema operativo. Tiempo estimado de llamadas del sistema.	Verificación Revisión conjunta	Requiere Reveladores

		nes de diseño - prueba completa transacción path - prueba completa modules/parts del producto de software - completa producto de software durante el ensayo fase							

Tabla 8. Métrica interna de rendimiento.

2.8.4 Métricas para medir la Mantenibilidad

Las métricas de mantenibilidad se refieren a los atributos que permiten medir el esfuerzo necesario para realizar modificaciones al software, ya sea por la corrección de errores o por el incremento de funcionalidad.

Métrica interna de Analisabilidad									
Nombre	**Propósito**	**Método de aplicación**	**Medición, Fórmula y elementos de datos computacionales**	**Interpretación del valor medido**	**Tipo de Escala Métrica**	**Tipo de medición**	**Fuente de Entrada Medida**	**ISO/IEC 12207 SLCP Referencia**	**Objetivo de Audiencia**
Impacto del cambio	¿Cuál es la frecuencia de los efectos adversos después de la modificación?	Contar el número de detectado los impactos adversos después de la modificación y compararlo con el número de modificaciones realizadas	X=1-A/B A= Detecta el número de los impactos adversos después de la modificación B=Número de modificaciones.	0<=X<=1 La más cercana a 1, el mejor.	Absoluto	X=Cuenta/Cuenta A=Cuenta B=Cuenta	A proviene del informe de revisión. B proviene del informe de revisión.	Verificación Revisión conjunta.	Requiere Reveladores de desarrollo

Localización del impacto de la Modificación	¿Cuán grande es el impacto de la modificación en el producto de software?	Contar el número de afectados de las variables de una modificación y compararlo con el número total de variables en el producto.	X=A/B A=Número de datos de las variables afectados por la modificación, confirmación de la revisión. B=Número total de variables.	0<=X<=1 La más cercana a 0, el menor impacto de modificación.	Absoluto	X=Cuenta/Cuenta A=Cuenta B=Cuenta	A proviene del informe de revisión. B proviene del informe de revisión.	Verificación Revisión conjunta.	Requiere Reveladores de desarrollo

Tabla 9. Métrica interna de analizabilidad.

Métrica interna de mutabilidad									
Nombre	**Propósito**	**Método de aplicación**	**Medición, Fórmula y elementos de datos computacionales**	**Interpretación del valor medido**	**Tipo de Escala Métrica**	**Tipo de medición**	**Fuente de Entrada Medida**	**ISO/IEC 12207 SLCP Referencia**	**Objetivo de Audiencia**
Registro de cambio	¿Son los cambios a las especificaciones y módulos del programa grabado adecuadamente en el código con líneas de comentarios?	Relacionar los expedientes del modulo cambiar información.	X=A/B A= número de cambios en las funciones/módulos de cambio observaciones confirmaron en examen B= número total de funciones/módulos cambió de código original	0<=X<=1 La más cercana a 1, el más grabable. El control de cambios 0 indica pobre control de cambios o pequeños cambios, alta estabilidad	Absoluto	X=Cuenta/Cuenta A=Cuenta B=Cuenta	Control de configuración Versión del sistema logs Especificaciones	Verificación Revisión conjunta.	Requiere Reveladores de desarrollo

Tabla 10. Métrica interna de mutabilidad.

Métrica interna de estabilidad									
Nombre	**Propósito**	**Método de aplicación**	**Medición, Fórmula y elementos de datos computacionales**	**Interpretación del valor medido**	**Tipo de Escala Métrica**	**Tipo de medición**	**Fuente de Entrada Medida**	**ISO/IEC 12207 SLCP Referencia**	**Objetivo de Audiencia**
Impacto del	¿Cuál es la frecuencia	Contar el número de	X=1-A/B A= número detectado	0<=X<=1 La más cercana a 1,	Absoluto	X=Cuenta/Cuenta	A proviene del informe de revisión.	Verificación Revisión conjunta.	Requiere Reveladores de

cambio	de los efectos adversos después de la modificación?	detectado los impactos adversos después de la modificación y compararlo con el número de modificaciones realizadas	de los impactos adversos después de la modificación. B= número de modificaciones.	el mejor.		A=Cuenta B=Cuenta	B proviene del informe de revisión.		desarrollo
Localización del impacto de la modificación	¿Cuán grande es el impacto de la modificación en el producto de software?	Contar el número de afectados por variables de una modificación y compararlo con el número total de variables en el producto. Comentarios (S) Todas las variables en la instrucción que fue cambiado b) variable que está en la misma instrucción con la variable definida por una).	X=A/B A= A=Número de datos de las variables afectados por la modificación, confirmación de la revisión. B=Número total de variables.	0<=X<=1 La más cercana a 0, el menor impacto de modificación.	Absoluto	X=Cuenta/Cuenta A=Cuenta B=Cuenta	A proviene del informe de revisión. B proviene del informe de revisión.	Verificación Revisión conjunta.	Requiere Revelador es de desarrollo

Tabla 11. Métrica interna de estabilidad.

2.9 Conclusiones

En el presente capítulo se presentaron los diferentes roles, técnicas y herramientas a emplear, teniendo en cuenta la especificación de cada uno de ellos, así como la definición del conjunto de actividades que se realizarán para llevar a cabo el procedimiento de evaluación de la arquitectura de software aplicando el método ATAM. Se propuso el uso de las métricas internas definidas por la ISO para cuando la arquitectura a evaluar posea la información requerida que dichas métricas necesitan.

Capitulo 3: Evaluación de la Arquitectura

3.1 Introducción

En este capítulo se efectúa la evaluación de la arquitectura propuesta con el método de evaluación ATAM. En el desarrollo del mismo se omiten los pasos siete y ocho (realización de la tormenta de ideas y priorización de los escenarios; así como el análisis de los nuevos escenarios obtenidos de dicha tormenta de ideas), debido al escaso tiempo que se dispuso para efectuar la evaluación, ausencia de interesados, (representantes del banco). De esta forma queda plasmado de forma práctica la aplicación del método, sirviendo de guía o ejemplo para evaluaciones futuras.

3.2 Evaluación

3.2.1 Presentación del método ATAM

En esta etapa el líder del equipo de evaluación presenta el método a los involucrados en la evaluación de la arquitectura. Debido a que en el capítulo dos se hace una caracterización de ATAM, incluyendo sus fases, no se hace necesario exponerlo nuevamente aquí. (Ver ATAM Capítulo1) [20][21][24][26]

3.2.2 Presentación del negocio

En la actualidad, en el Banco Nacional se utiliza un sistema informático desarrollado por la empresa de Servicios Informáticos para el Banco Central (SIBANC). Este sistema se encuentra sobre la plataforma MSDOS y fue desarrollado sobre Fox Pro. El gestor de base de datos utilizado para persistir la información es SQL Server 2005. Esta aplicación contiene la lógica de negocio en el gestor de base de datos, agrupados en funciones y procedimientos almacenados. SIBANC ha actualizado este sistema y lo ha desplegado en la plataforma Windows en el Banco Metropolitano y en el BICSA.

La actualización del sistema fue sobre el mismo lenguaje con el que se desarrolló anteriormente, Fox Pro y SQL Server 2005. Una de sus características fundamentales es la presencia de un núcleo central dedicado principalmente a la contabilización de las operaciones bancarias. Dado un estudio de este núcleo central se decidió reutilizar las funcionalidades del mismo. La interacción con este componente se

hará mediante sus procedimientos almacenados. El Banco Nacional de Cuba tiene la particularidad que no posee centros de costos, por lo que el sistema que se despliegue en este banco debe comportarse como centro informativo y centro contable. La política de distribución de las transacciones contables será resuelta por la reutilización del núcleo central mencionado anteriormente.

El sistema a desarrollar está concebido para desplegarlo en un entorno Web. El mismo intercambiará información con otros sistemas internos o externos a la institución por lo que se concibe un módulo para el desarrollo de esta interacción. La comunicación se concibe que sea a través de servicios web, tanto publicarlos como consumirlos, invocaciones remotas en el caso que no implique riesgo para la seguridad de la información que se maneja y la utilización de mensajería asíncrona.

Para el intercambio de información con otras instituciones, se deberá definir un protocolo de comunicación con los mismos. Para conseguir este objetivo se evaluará primero, las condiciones para esta integración y posteriormente se definirán los protocolos establecidos entre los ministerios o instituciones y el BCC.

El sistema está encaminado a lograr ciertos objetivos de calidad, como se podrá observar a continuación:

Soporte: Solo se necesita de un navegador para ejecutar la aplicación en las estaciones de trabajo, las actualizaciones solamente son en el servidor. La instalación del sistema consiste en publicar la aplicación en un servidor.
Seguridad: El uso del protocolo HTTPs, utilizando el SSL, además la información será encriptada. La lógica de la aplicación reside en el servidor.

Usabilidad: Peticiones asíncrona con la tecnología Ajax, componentes visuales de java script.

Prestaciones en el cliente: El sistema reside en el servidor por lo que las máquinas clientes no necesitan muchas prestaciones para ejecutar la aplicación.

Interacción con dispositivos: Para interactuar con dispositivos desde la Web están los applets de Java. Un componente Java que se ejecuta en el navegador y es capaz de interactuar con el cliente. Es necesario tener la máquina virtual instalada en cada computadora que necesite interacción con dispositivo.
Integridad en los datos: Como la lógica de la aplicación reside en el servidor de la aplicación, todas las funcionalidades se pueden ejecutar transaccionalmente, asegurando la integridad en la información.
Escalable y rendimiento: Se pueden establecer Balanceo de cargas y Clusters del lado del servidor para soportar una gran cantidad de peticiones concurrente.

Despliegue: Las aplicaciones Web están desplegadas generalmente en tres capas: Cliente, Servidor web o de aplicaciones y Servidor de Base de datos.

Entre las características más importantes del sistema se encuentra su desarrollo, el cual será a través de tecnologías y componentes open source, y sobre la plataforma Java. La aplicación será multiplataforma soportando sistemas Unix y Windows, y se desplegará en un entorno Web. Una vez desplegado el sistema, deberá permitir la comunicación con otros sistemas por diferentes vías, tanto por servicios web, mensajería, invocación remota u otros tipos de comunicación según se establezca.

Para la seguridad se propone la utilización del protocolo SSL en la transmisión de los datos, utilización de encriptación de la información, incluyendo otras verificaciones y métodos que se identifiquen. Se auditarán las operaciones que se definan para conocer: qué, quiénes, dónde y cuándo ejecutaron una operación determinada. Todas las operaciones estarán envueltas en transacciones para asegurar que no exista inconsistencia en las informaciones que se gestionan. El sistema informático a desarrollar soportará como gestor de base de datos a Microsoft SQL Server.

3.2.3 Presentación de la arquitectura

Para el desarrollo de la aplicación se propuso el siguiente entorno de desarrollo:

- Plataforma Java (Java Virtual Machine [OpenJDK], Java Enterprise Edition, Java Standard Edition).
- Servidor Web: Apache Tomcat
- Control de versiones: SubVersion
- Base de Datos: SQL Server
- Herramienta de modelado: Visual Paradigm
- Herramienta para modelar la Base de datos: Erwin

Presenta los siguientes componentes y framworks:

- Spring Framework
- Spring Web Flow
- Spring Web Services
- Spring Security System
- Spring Batch
- Spring JMS
- Spring JCA
- AspectJ
- Jcaptcha
- EhCache
- Dynamic Jasper
- Jasper Report
- JfreeChart
- Hibernate
- Dojo Toolkit
- Wife

Se utiliza convenciones o estándares de código y recursos. Para las clases Java se adoptan las convenciones estándares presentadas en la Especificación del Lenguaje Java por la compañía Sun Microsystem.

A continuación se muestra el diagrama de despliegue que se utilizará en el Banco Nacional de Cuba:

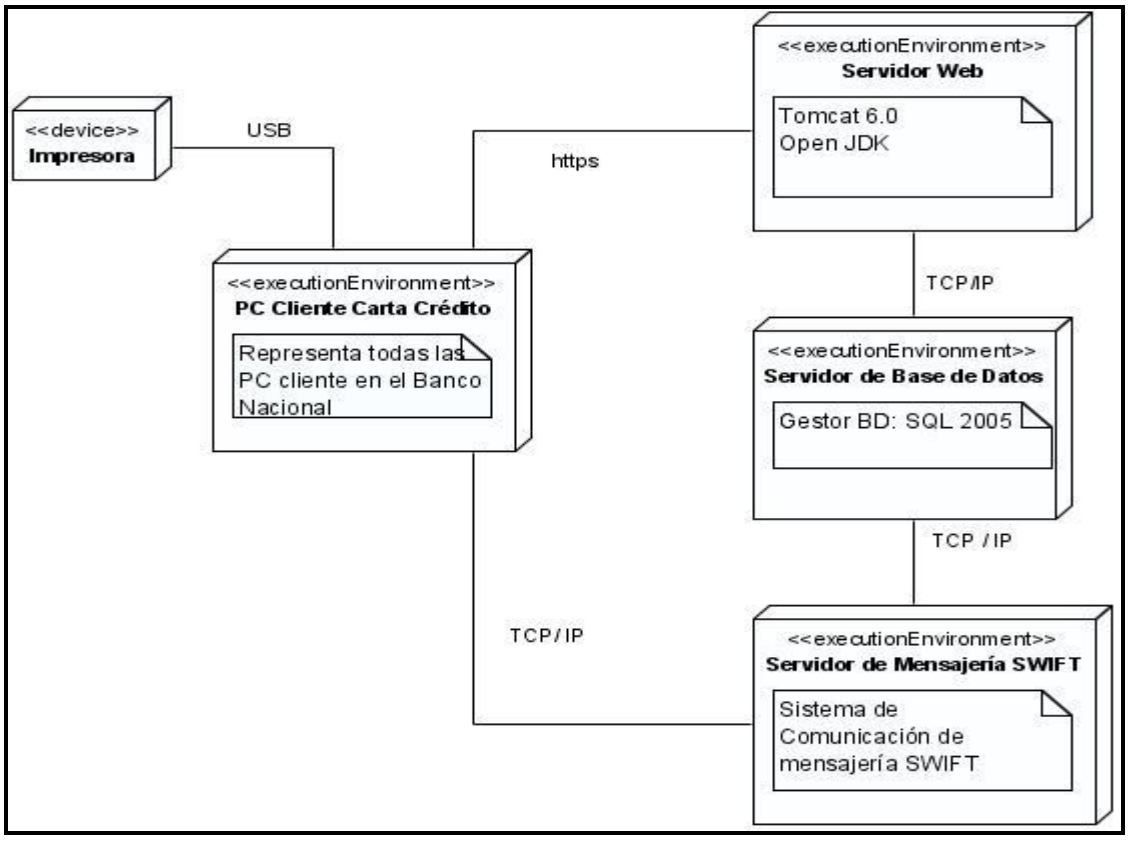


Figura 2. Modelo de despliegue para el Banco Nacional de Cuba.

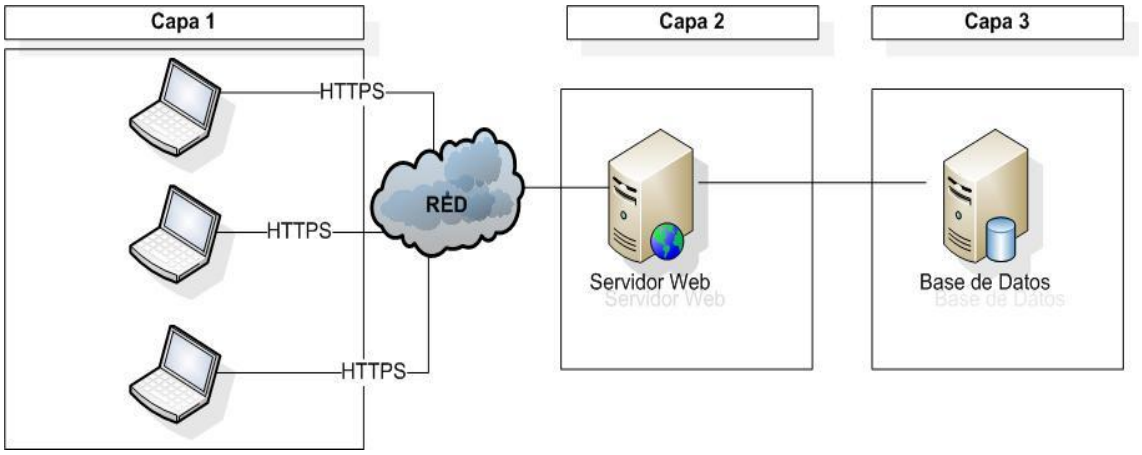


Figura 3. Capas.

En la mayoría de los puestos de trabajo se utilizarán impresoras, conectándose a cada puesto de trabajo por USB. La comunicación entre las estaciones de trabajo y el servidor Web será por HTTPS. Por otra parte, la comunicación del servidor de mensajería con las estaciones de trabajos será por TCP / IP. El intercambio de información del servidor Web y el servidor de mensajería SIWFT con el servidor de Base de datos será por TCP / IP.

En las estaciones de trabajo estarán instaladas computadoras P4, 512 Mb RAM y procesador de 2.4 HZ. El servidor Web es un servidor IBM con 2 Hg RAM, 250 Hg de memoria en el disco duro y 3.0 Hz la velocidad del microprocesador. Para la base de datos se utilizará un servidor con las mismas prestaciones que el servidor Web. El servidor de mensajería constará con una computadora con 1 Hg RAM, a 3.0 Hz de velocidad en el microporcesador y 120 Hg de disco duro.

Para ganar en organización en el desarrollo y en el despliegue del sistema, se agruparán los Módulos y Componentes por Subsistema, cada Subsistema tendrá uno o más Módulos y/o Componentes estrechamente relacionados con las funcionalidades que ejecutan. Los Módulos y/o Componentes estarán separados por diferentes capas lógicas según la naturaleza de los mismos. Las capas lógicas definidas son:

Capa de Presentación

Lógica de presentación.

Capa de Negocios

Fachada y Lógica de negocio.

Capa de Acceso a Datos

Desarrollo de DAOs. Clases para el acceso a la base de datos.

Capa de Dominio

Dominios o entidades persistentes.

La capa de presentación estará dividida en dos partes. Una subcapa del lado del servidor, encargada de recibir todas las peticiones de la interfaz de usuario, controlar el flujo de presentación del sistema y enviar las respuestas correspondientes a la interfaz de usuario. La otra subcapa estará en el cliente, utilizándose los componentes visuales de Java Script para manejar los eventos y validaciones del lado de cliente. La subcapa colocada en el lado del servidor estará relacionada con la Capa de Negocios y de Dominio.

La capa de negocio está dividida en dos subcapas principales (Fachada y Manager) sin dejar de incluir otras que se necesiten y que estén relacionadas con el negocio. En la Fachada se expondrán todas las funcionalidades que la capa de presentación necesitará. Esta capa invocará métodos de la subcapa Manager. En esta capa se implementará el negocio de los módulos en cuestión, y de aquí se accederá de ser necesario a la Capa de Acceso a Datos, a otras Capas de Negocios y/o a la Capa de Dominio.

En la capa de acceso a datos se implementarán los métodos que interactúan con el gestor de Base de Datos. Esta capa (DAO) tendrá solamente dependencia con la Capa de Dominio. En la capa de dominio se declararan todas las clases que representan entidades del negocio. Estas clases de dominio estarán presentes en todas las capas anteriormente descritas. A continuación se muestra una figura con la estructura de las capas lógicas descritas anteriormente.

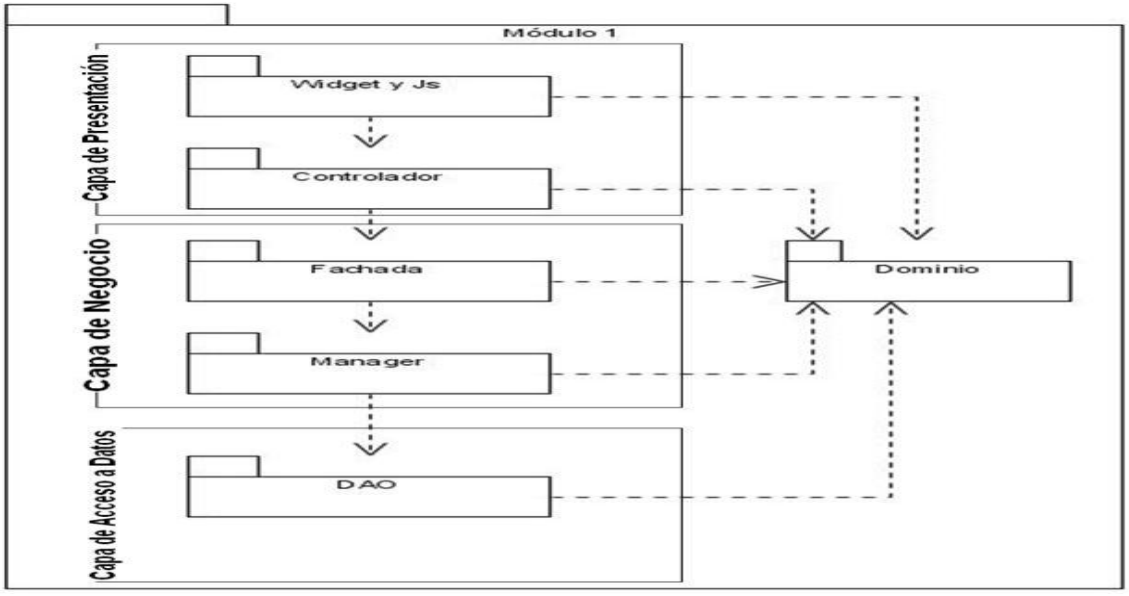


Figura 4. Estructura de las capas lógicas del sistema.

A continuación se muestra otra figura con las capas lógicas y los framework que se utilizarán en cada una:

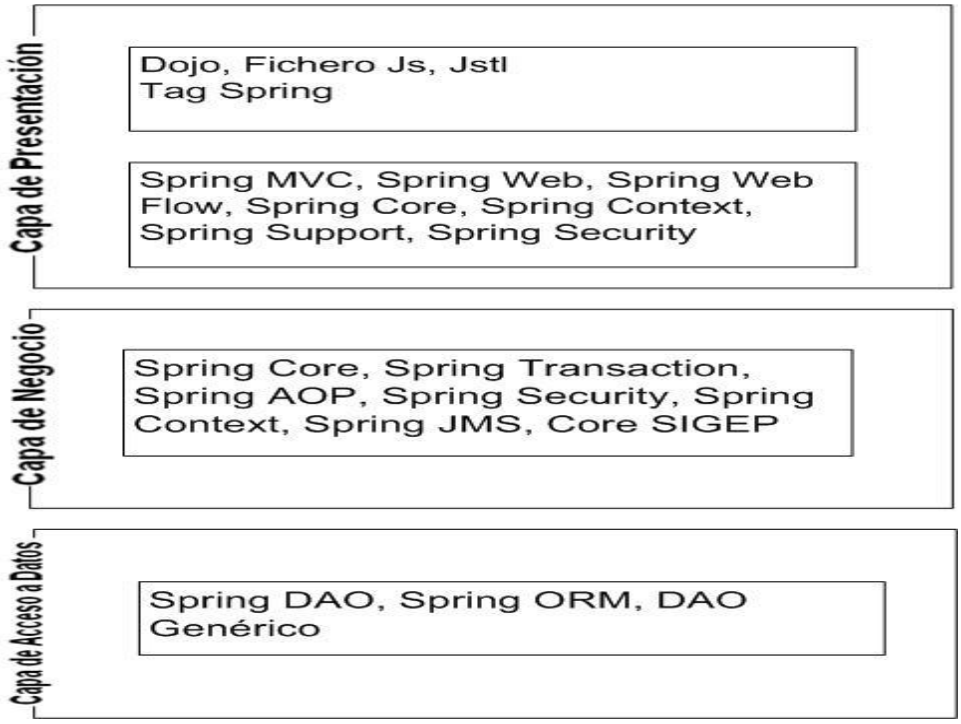


Figura 5. Capas lógicas y frameworks.

Módulos y submódulos del sistema:

Contabilidad.

- Plan de Cuentas.
- Transacciones
- Libros Contables.
- Balanza de Pago.
- Análisis de los estados financieros.
- Análisis de cuentas.

- Clientes.
  - Gestión de Clientes.
  - Involucrados.
  - Personas autorizadas.
- Cuentas de Clientes
- Préstamos
- Depósitos a Plazo.
- Comercio Exterior.
  - Emisión de Cartas de Crédito.
  - Negociación de Cartas de Crédito.
  - Pago de Cartas de Créditos
- Garantías.
- Transferencias.
- Títulos Valores.
  - Módulo de Cheques.

  - Módulo de Letra de Cambio.
  - Módulo de Pagaré
  - Módulo de Bonos.
  - Otros.
- Créditos.
- Factoraje.
- Arrendamiento (Leasing).
- Cajas
- Cajas de Seguridad
- Fiducias.
- Fondos de Inversión.
- Reportes.
- Seguridad
- Mensajería

El mecanismo de inicialización de los módulos se encarga de instanciar primeramente los módulos que no presentan dependencia con otros. Luego se inicializan los siguientes módulos que dependen de los primeros y así consecutivamente. La inicialización de un módulo consiste en inicializar todas las clases declaradas en los contextos definidos en el módulo. Primero se inicializa el contexto que contiene las clases de acceso a datos, luego el contexto con las clases de la capa de negocio y finalmente el contexto con las clases de la capa de presentación.

A continuación se muestra el diagrama que visualiza lo descrito anteriormente.

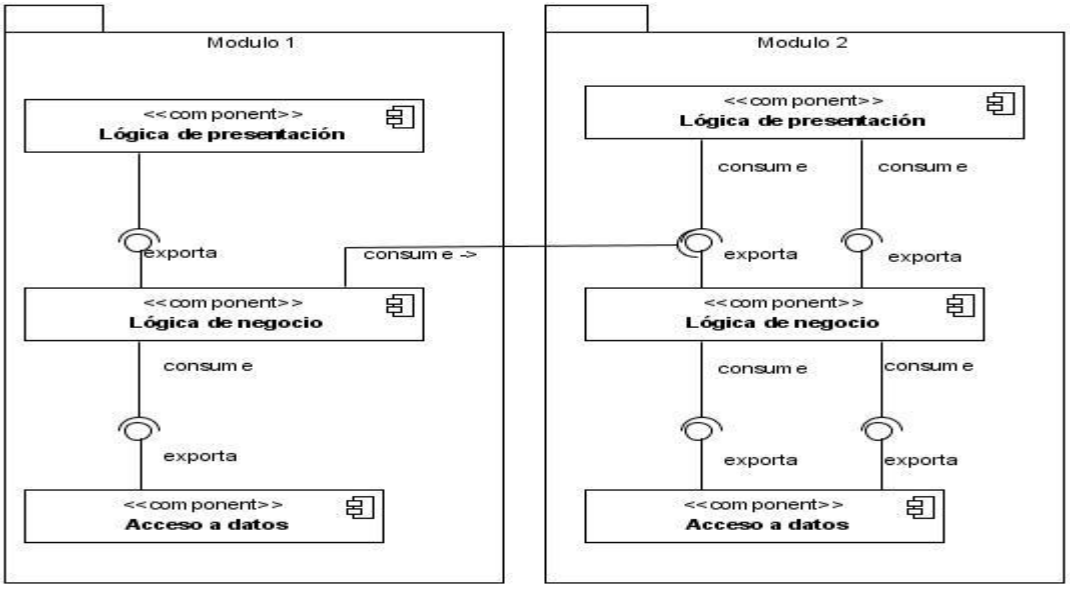


Figura 6. Diagrama de componente.

Durante el desarrollo de la arquitectura se definieron patrones tanto de diseño como de arquitectura, estos son:

- MVC
- GoF (Gangs of Four)
- DAO (Data Acces Object)
- Patrones propuestos para JEE (Front Controller)

3.2.4 Identificación de los enfoques arquitectónicos

Se utiliza una arquitectura en capas, el patrón MVC en la capa de presentación, así como el patrón DAO para el acceso a los datos. Se utiliza el patrón de repositorio de datos

3.2.5 Generación del árbol de utilidad

Atributos de calidad	Descomposición del atributo	Escenarios

Seguridad	Auditabilidad de acceso	(A,A) Un usuario intercambia datos con el sistema.
	Prevención de la corrupción de datos	(A,M) Comunicación entre el cliente y el servidor web.
	Confidencialidad de los datos	(A,A) Seguridad de las transacciones en un 100%.
Funcionalidad	Adecuación funcional	(A,M) Se realiza una transacción contable.
Rendimiento	Tiempo de respuesta < 3 segundos	(A,A) Comunicación con la base de datos.

Tabla 12. Generación del árbol de utilidad.

3.2.6 Análisis de los enfoques arquitectónicos

Escenario #: E1	Un usuario intercambia datos con el sistema		
Atributo de calidad:	Seguridad, Auditabilidad de acceso		
Ambiente:	Operaciones CRUD		
Estímulo:	Cualquier actividad que el usuario realice sobre el sistema (modificar, insertar, eliminar, etc.)		
Respuesta:	El sistema debe auditar todas las acciones que realiza el usuario con la mayor cantidad de datos posibles.		
Decisiones arquitectónicas	Puntos de sensibilidad	Puntos de acuerdos	Riesgos

DA1 Programación orientada a aspectos.	Las convenciones de nombres establecidas en la arquitectura en los métodos que se auditarán		Retrasa el tiempo de respuesta por petición, ya que se intercepta cada una de ellas.
Razonamiento:	Con la programación orientada a aspecto se puede garantizar que se auditen las operaciones que el desarrollar desee. Para esto es necesario cumplir con las convenciones de nombres que se establezcan en el desarrollo. Esta decisión arquitectónica puede afectar en algo el tiempo de respuesta de las peticiones.		

Tabla 13. Análisis de los enfoques arquitectónicos. Escenario 1.

Escenario #: E2	Comunicación entre el cliente y el servidor web		
Atributo de calidad:	Seguridad, Prevención de la corrupción de datos		
Ambiente:	Normal		
Estímulo:	Se realiza una petición desde el cliente hacia el servidor web		
Respuesta:	Se responde a la petición sin afectar los datos que se envían desde el cliente al servidor y viceversa.		
Decisiones arquitectónicas	Puntos de sensibilidad	Puntos de acuerdos	Riesgos
DA: Componente de validación en el cliente y en el servidor	Complejidad en las validaciones		Retrasa el tiempo de respuesta por petición, ya que se intercepta cada una de ellas. Que no se valide correcta mente.
Razonamiento:	Con el componente de validación se garantiza que se realice la validación por una única estructura y que cada desarrollador posea la misma herramienta para validar cualquier tipo de dato. La complejidad en las validaciones tendrá que ser tomada en cuenta para garantizar que no exista corrupción en los datos enviados y recibidos.		

Tabla 14. Análisis de los enfoques arquitectónicos. Escenario 2.

Escenario #: E3	Comunicación con la base de datos		
Atributo de calidad:	Rendimiento, Tiempo de respuesta		
Ambiente:	Normal		
Estímulo:	Se realiza una petición al servidor de base datos.		
Respuesta:	El sistema debe responder en un tiempo menor a 3 segundos		
Decisiones arquitectónicas	Puntos de sensibilidad	Puntos de acuerdos	Riesgos
DA: Pool de conexiones	Número de conexiones habilitadas en el pool de conexiones a la base de datos.		Congestión del servidor de base de datos
El grado de normalización de la base de datos	Alto nivel de normalización de la base de datos.		Atrasar el tiempo de respuesta según lo establecido
Razonamiento:	Con el pool de conexiones se habilitan los canales para conectarse al servidor de base de datos. De acuerdo al número de conexiones se puede o no congestionar el servidor, por lo que es necesario tener en cuenta la cantidad de conexiones concurrente que soporta el gestor de base de datos. Por otro lado, el grado de normalización de la base de datos puede afectar al tiempo de respuesta de una petición, por lo que es necesario normalizar la base de datos mientras no afecte el tiempo de respuesta.		

Tabla 15. Análisis de los enfoques arquitectónicos. Escenario 3.

3.2.7 Resultados

En la evaluación de la arquitectura del proyecto se obtuvo como salidas del método los escenarios utilizados, el árbol de utilidad, los puntos sensibles, puntos de acuerdos y los riesgos, como muestran las siguientes tablas:

Escenarios identificados:

Lista de escenarios	
(A,A)	Un usuario intercambia datos con el sistema
(A,M)	Comunicación entre el cliente y el servidor web
(A,A)	Seguridad de las transacciones en un 100%
(A,M)	Se realiza una transacción contable.
(A,A)	Comunicación con la base de datos

Tabla 16. Lista de escenarios.

Árbol de utilidad:

Atributos de calidad	Descomposición del atributo	Escenarios
Seguridad	Auditabilidad de acceso	(A,A) Un usuario intercambia datos con el sistema
	Prevención de la corrupción de datos	(A,M) Comunicación entre el cliente y el servidor web
	Confidencialidad de los datos	(A,A) Seguridad de las transacciones en un 100%
Funcionalidad	Adecuación funcional	(A,M) Se realiza una transacción contable.

Rendimiento	Tiempo de respuesta < 3 seg	(A,A) Comunicación con la base de datos

Tabla 17. Árbol de utilidad.

Riesgos:

Escenario	Riesgos
1	Retrasar el tiempo de respuesta por petición, ya que se intercepta cada una de ellas.
2	Con el componente de validación se garantiza que se realice la validación por una única estructura y que cada desarrollador posea la misma herramienta para validar cualquier tipo de dato. La complejidad en las validaciones tendrá que ser tomada en cuenta para garantizar que no exista corrupción en los datos que se envían y que se reciban.
3	-Congestión del servidor de base de datos -Atrasar el tiempo de respuesta según lo establecido

Tabla 18. Riesgos encontrados.

Puntos de sensibilidad y puntos de acuerdos:

Escenario	Puntos de sensibilidad	Puntos de acuerdos
1	Las convenciones de nombres establecidas en la arquitectura en los métodos que se auditarán	No hay
2	Complejidad en las validaciones	No hay
3	Número de conexiones habilitadas en el pool de conexiones a la base de datos. Nivel alto de normalización de la base de datos.	No hay

Tabla 19. Puntos sensibles y de acuerdos.

3.3 Conclusiones

En este capítulo se realizó la evaluación de la arquitectura del proyecto Modernización del Sistema Bancario Cubano, teniendo en cuenta el procedimiento descrito en el capítulo dos (método ATAM), arrojando como resultado una serie de riesgos y puntos sensibles, así como los escenarios identificados. Estos resultados serán presentados al arquitecto del proyecto facilitándole la toma de decisiones.

Conclusiones Generales

En el presente trabajo, el estudio de estado del arte realizado permitió actualizar el conocimiento acerca de los métodos de evaluación de arquitectura ATAM, SAAM, MECABIC, BOSCH, sus características así como el modo en que cada uno de ellos evalúa la arquitectura. Se definieron los pasos a seguir para efectuar la evaluación y los roles que juegan cada uno de los involucrados en el proceso, posibilitando una mejor organización en el desarrollo de la evaluación de la arquitectura.

Con la aplicación del método ATAM para la evaluación se analizó el comportamiento de la seguridad del sistema, el cual es de vital importancia en el proyecto, así como el rendimiento, aunque este último de forma insuficiente, debido al estado actual de la arquitectura evaluada, la cual no presenta implementación y por tanto se torna imposible realizar algún cálculo de rendimiento real. Fueron detectados puntos de sensibilidad así como los riesgos que pueden afectar la arquitectura, producto del conocimiento adquirido y de cómo las decisiones arquitectónicas influyeron en los atributos de calidad evaluados.

Recomendaciones

Debido a la importancia que representa para la universidad la calidad de sus productos se recomienda:

1) Incluir el proceso de evaluación de la arquitectura como una tarea más dentro de cualquier equipo de calidad de proyectos.

2) Realizar la evaluación de la arquitectura del proyecto Modernización del Sistema Bancario Cubano cuando esta se encuentre en un mayor nivel de especificación, o módulos implantados.

Bibliografías

[1] Erika Camacho, Fabio Cardesco, Gabriel Núñez. Arquitecturas de Software. 2004.

[2] Reynoso, Carlos Billy. Introducción a la Arquitectura de Software. 2004.

[3] Canal, Carlos. Arquitectura, marcos de trabajo y patrones. 2005.

[4] Buschmann, F., R. Meunier, et al. (1996). Pattern-oriented software architecture – A system of patterns, John Wiley & Sons.

[5] Fowler, M. (2002). Patterns of Enterprise Application Architecture Addison-Wesley Professional.

[6] Husted, T., C. Dumoulin, et al. (2003). Struts in Action, Manning Publications Co. IEEE (2000). IEEE Std 1471-2000 IEEE Recommended Practice for Architectural Description of Software- Intensive Systems.

[7] Pressman, R. S. (1998). Ingeniería de Software, un enfoque práctico. Cuarta Edición, Mc Graw Hill.

[8] PAUL CLEMENTS, R. K., MARK KLEIN. Evaluating Software Architectures: Methods and Case Studies. Addison-Wesley, 2001. 323 p.

[9] Evaluación de la arquitectura de software.www.fing.edu.uy. Disponible en: 5, 2008.

[10] REGALADO, I. Y. V. y CABANA, I. E. F. La arquitectura de software como disciplina científica. 2008, Disponible en: http://www.gestiopolis.com/administracion-estrategia/arquitectura-de-sofware-como-disciplina-cientifica.htm.

[11] La Ingeniería de Software y RUP 2007, Disponible en: http://www.slideshare.net/dersteppenwolf/la-ingeniera-de-software-y-rup

[12] Arquitectura para los Sistemas que Conforman la Intranet Universitaria. 2007.

[13] Camacho E, Caradeso F. y Núñez G. (2004). Guía de estudio de la Arquitectura de Software.

[14] FLOWER, M. Patter of Enterprise Application Architecture. 2002. p.

[15] Pressman R. (2005). Ingeniería del Software. Un enfoque practico. Parte 1. La Habana Cuba: Félix Varela.

[16] (Bylli Reynoso, 2006): Bylli Reynoso, Carlos. 2006. Wylli.net, 26 de 6 de 2006. [Citado el: 28 de 12 de 2007.] Disponible en: http://www.microsoft.com/spanish/msdn/arquitectura/roadmap_arq/intro.mspx.

[17] REYNOSO, B. Introducción a la Arquitectura de Software Buenos Aires

[18] Giraldo. Métricas, Estimación y Planificación en Proyectos de Software. Disponible en: www.WilyDev.Net.

[19] Heidi González Doria**.** Las Métricas de Software y su Uso en la Región. Disponible en: http://catarina.udlap.mx/u_dl_a/tales/documentos/lis/gonzalez_d_h/capitulo1.pdf.

[20] Kazman, R, Clements, P y Klein, M. Evaluating Software Architectures. Methods and case studies. s.l. Addison Wesley. , 2001.

[21] Gómez, Omar Salvador Gómez. 2007. Evaluando Arquitecturas de Software. Parte 1. Panorama General. México: Brainworx S.A, 2007. 1870-0888.

[22] Kazman, R., M. Klein, et al. (2000). ATAM: Method for Architecture Evaluation.

[23] Nguyen Xuan Huy, "Software Engineering", Institute of Information Technology, National Center for Natural Science and Technology. (Sección 1.2 - "Quality attributes of software products"). Disponible

en: http://www.netnam.vn/unescocourse/se/software.htm

[24] P. Clements, R. Kazman, and M. Klein, "Evaluating Software Architectures – Methods and Case Studies," Addison-Wesley, 2002.

[25] R. Kazman, L. Bass, G. Abowd, and M. Webb, "SAAM: A Method for Analyzing the Properties of Software Architectures," in Proceedings of 16th International Conference on Software Engineering, pp. 81-90, May 1994.

[26] M. Barbacci, P. Clements, A. Lattanze, L. Northrop, and W. Wood, "Using the Architecture Tradeoff Analysis Methods (ATAM) to Evaluate the Software Architecture for a Product Line of Avionics Systems: A Case Study," Technical Note, CMU/SEI-2003-TN-012, July 2003.

[27] J. K. Bergey, M. J. Fisher, and L. G. Jones, "Use of the Architecture Tradeoff Analysis Method (ATAM) in Source Selection of Software-Intensive Systems," Technical Note, CMU/SEI-2001-TN-010, June 2002.

[28] P. C. Clements, "Active Reviews for Intermediate Designs," Technical Note, CMU/SEI-2000-TN-009, August 2000.

[29] T. J. Dolan, "Architecture Assessment of Information-System Families," Ph.D. Thesis, Department of Technology Management, Eindhoven University of Technology, February 2002.

[30] F. Linden, J. Bosch, E. Kamsties, K. Kansala, and H. Obbink, "Software Product Family Evaluation," in Proceedings of the 3rd International Conference on Software Product Lines, pp. 110-129, Boston, September 2004.

[31] Mendoza, G.M. (2006) ISO 9126-3: Métricas Internas de la Calidad del Producto de Software. Volume.

[32] Melian., J.M.M. and J.F.H. Ballesteros., La calidad del software y su medida.

[33] Doria, H.G., Las Métricas de Software y su Uso en la Región. , in Departamento de Ingeniería en Sistemas Computacionales, Escuela de Ingeniería. 2001, Universidad de las Américas-Puebla.

[34] Sans, M.C. (1998) Las normas ISO.Volume.

[35] Santos, J.M. Mediciones. Enfoques de las métricas. Volume.

[36] Rodríguez., F.S. (1999) Medida del Tamaño Funcional de Aplicaciones Software. Volume, 65.

[37] Estévez, I.P., Métricas para el control de proyectos de software. 2002, Instituto Superior Politécnico "José Antonio Echeverría".Facultad de Ingeniería Industrial. Centro de Estudios de Ingeniería de Sistemas: Ciudad de la Habana.

[38] Rick Kazman, Mark Klein, Paul Clements, "Software Engineering Institute| Carnegie Mellon". 2009, Universidad Camegie Mellon. Disponible en: http://www.sei.cmu.edu/publications/documents

[39] Kevin Henry, JDBC – Conectividad de la Base de Datos de Java, 2009. Disponible en: http://www.acm.org/crossroads/espanol/xrds7-3/ovp_marzo2001.html

[40] Rick Kazman, Mark Klein, Paul Clements. ATAM: Method for Architecture Evaluation. Pittsburgh, PA 15213-3890 : Carnegie Mellon, August 2000.

[41] Lenn Bass, Paul Clements, Rick Kazman. Software Architecture in Practice, Second Edition. s.l. : Addison Wesley, April 11, 2003.

[42] Alekander González, Marizé Mijares, Luis E. Mendoza, Anna Grimán, María Pérez. Método de Evaluación de Arquitectura de Software Basadas en Componentes (MECABIC). Universidad Simón Bolívar: s.n.

Referencias Bibliográficas

A.Galvis (2000). Ingeniería de software educativo. Colombia, 2da. Reimpresión.

Estévez, I. P. (2002). Métricas para el control de proyectos de software. Ciudad de la Habana, Instituto Superior Politécnico “José Antonio Echeverría”.

FENTON, E. N. (1991). Software Metrics A rigorous approach.

Albin, Stephen. 2003. The Art of Software Architecture: Design methods and techniques. Nueva York: Wiley, 2003.

Bengtsson, P. 1999. University of Karlskrona. Design and Evaluation of Software Architecture. Disponible en: 1999. http://www.ipd.hk-r.se/pob/archive/thesis.pdf

Brey, Gustavo Andrés, et al. 2005. Arquitectura de Proyectos de IT. Evaluación de Arquitecturas.

Glosario de Términos

QAW: Quality Atribute Workshop, taller enfocado a la captura de requerimientos orientados a la arquitectura.

ADD: Atribute Driven Design, se toman escenarios y se van realizando elecciones de diseño que permitan satisfacer los requerimientos descritos por los escenarios.

VaB: Views and Beyond, como resultado de ADD, se obtienen distintas estructuras del sistema, formadas de componentes y sus relaciones.

Hardware: Es la parte física de un computador y más ampliamente de cualquier dispositivo electrónico.

Software: Se refiere al equipamiento lógico o soporte lógico de un computador digital, comprende el conjunto de los componentes lógicos necesarios para hacer posible la realización de una tarea específica. Incluyen, entre otras, aplicaciones informáticas y software de sistema.

Escenario: Breve descripción de la interacción de algunos de los involucrados en el desarrollo del sistema.

Arquitectura de Software: Según el estándar IEEE (Institute of Electrical and Electronics Engineers): "La Arquitectura de Software es la organización fundamental de un sistema encarnada en sus componentes, las relaciones entre ellos y el ambiente y los principios que orientan su diseño y evolución".

Componente: "Es una parte de una arquitectura de software claramente identificable, independiente a la aplicación en la que se utiliza y de otros componentes (partes), que describe y realiza funciones específicas y claras dentro del contexto de la arquitectura, puede ser modificado durante el diseño, posee una documentación clara que permite conocer sus características, atributos y comportamiento, reutilizable y su interoperabilidad con otros componentes (partes) no reduce el nivel de eficiencia de la arquitectura".

DAO: En software de computadores, un Data Access Object (DAO, Objeto de Acceso a Datos) es un componente de software que suministra una interfaz común entre la aplicación y uno o más dispositivos de almacenamiento de datos, tales como una Base de Datos o un archivo. El término se aplica frecuentemente al Patrón de Diseño Object.

Spring: El Spring Framework (también conocido simplemente como Spring) es un framework de código abierto de desarrollo de aplicaciones para la plataforma Java.

ORM: Mapeo de Objetos Relacionales (ORM). Es una abstracción a la base de datos.

Hibernate: Es una herramienta de Mapeo objeto-relacional para la plataforma Java (y disponible también para .Net con el nombre de NHibernate) que facilita el mapeo de atributos entre una base de datos relacional tradicional y el modelo de objetos de una aplicación, mediante archivos declarativos (XML) que permiten establecer estas relaciones.

SGBD: Un Sistema Gestor de base de datos es un conjunto de programas que permiten crear y mantener una Base de Datos, asegurando su integridad, confidencialidad y seguridad.

BD: (Base de Datos) es un conjunto exhaustivo no redundante de datos estructurados organizados independientemente de su utilización y su implementación en máquina accesibles en tiempo real y compatibles con usuarios concurrentes con necesidad de información diferente y no predicable en tiempo.

JDBC: La conectividad de la base de datos de Java (Java Database Connectivity) es un marco de programación para los desarrolladores de Java que escriben los programas que tienen acceso a la información guardada en bases de datos, hojas de cálculo, y archivos "planos". JDBC se utiliza comúnmente para conectar un programa del usuario con una base de datos por “detrás de la escena”, sin importar qué software de administración o manejo de base de datos se utilice para controlarlo. De esta manera, JDBC es una plataforma-cruzada.

Facade: El patrón de diseño facade sirve para proveer de una interfaz unificada sencilla que haga de intermediaria entre un cliente y una interfaz o grupo de interfaces más complejas.

Manager: Cualquier persona que utiliza la gestión de competencias o la organización posee título de "director".

Spring MVC: Model View Controller (Modelo Vista Controlador) Patrón arquitectónico aportado por SmallTalk, Modelo 2 de aplicaciones WEB, Reparte las responsabilidades de la aplicación entre tres elementos: El Modelo: Corre de la aplicación. Reglas de negocio y capa de persistencia, La Vista: Reenderezado del sistema, El Controlador: Control del flujo de navegación de la aplicación.

Spring Web Flow: Es un módulo del framework Spring dirigido a definir y gestionar los flujos de páginas dentro de una aplicación web.

Librería Dojo: Dojo es conocido como “la navaja suiza del ejército de las bibliotecas Javascript”. Proporciona una gama más amplia de opciones en una sola biblioteca JavaScript y es compatible con navegadores antiguos.”

Sistemas Operativos: Es el programa (o software) más importante de un ordenador. Para que funcionen los otros programas, cada ordenador de uso general debe tener un sistema operativo. Los sistemas operativos realizan tareas básicas, tales como reconocimiento de la conexión del teclado, enviar la información a la pantalla, no perder de vista archivos y directorios en el disco, y controlar los dispositivos periféricos tales como impresoras, escáner, etc.

Anexos

Pasos del método Mecabic	
Fase	**Pasos**
1) Fase de presentación	1) Presentación del método. En el primer paso el evaluador principal presenta el método ante los participantes con el proyecto. Durante una reunión se explica el proceso que debe cumplir cada participante del proyecto, se responden preguntas y se establecen las expectativas y el contexto sobre las actividades o tareas restantes 2) Presentación de las directrices del negocio. Se explica a los stakeholders el contexto del sistema y los requerimientos del negocio dentro del cual va a funcionar el sistema. Algunos tópicos que debería contener esta presentación son: las funcionalidades más importantes del sistema, los objetivos del negocio y el contexto relacionado con el proyecto, el grupo dirigente de los stakeholders, las directrices arquitectónicas (Requerimientos de calidad a ser satisfechos por la arquitectura), restricciones técnicas, gerenciales, económicas y políticas y obertura de la evaluación y límites del sistema. 3) Presentación de la arquitectura. El arquitecto o grupo de representantes, debe explicar a los stakeholders cuál es la arquitectura evaluada. Debe contener una clara diferenciación estructural, la cual deberá mostrar claramente las relaciones entre componentes de la arquitectura y las diferentes granularidades involucradas. En dicha presentación, el arquitecto cubre restricciones técnicas (Sistema operativo, hardware, otros sistemas con el cual el sistema debe interactuar, etc.) y los alcances arquitectónicos usados para alcanzar los requerimientos. El arquitecto debe transmitir lo esencial y de esta manera abrevia la información de la arquitectura que requiere el equipo.
2)Investigación y análisis	4) Identificación de elementos de diseño. En este paso se contemplan alternativas de diseño de la arquitectura, que posteriormente serán analizadas para ver cómo responden a los requerimientos de calidad solicitados. La arquitectura debe ser evaluada completamente, desde el sistema con sus componentes de mayor granularidad, hasta el mínimo nivel de granularidad que corresponde a los componentes. En este método se propone identificar elementos de diseño dentro de una arquitectura basada en componentes como un componente y el conjunto de componentes con los cuales interactúa, un conjunto de asociaciones que sea de interés sobre alguna característica de calidad particular o conjunto de decisiones que tenga influencia considerable sobre otros elementos de diseño. Para identificar un elemento de diseño es necesario considerar los servicios ofrecidos por los componentes disponibles y las diferentes opciones para definir los servicios de los componentes desarrollados para el sistema. La manera de documentar los elementos de diseño depende de la importancia que pueda tener dentro de la evaluación, del conjunto de decisiones o adaptaciones que sea necesario realizar, y del conocimiento de los stakeholders presentes en la evaluación. 5) Generación del árbol de utilidad. MECABIC proporciona un árbol de utilidad inicial

	especifico para ASBC a partir del cual seleccionan un conjunto de escenarios de interés, el cual está basado en el modelo de calidad ISO 9126-1 para arquitecturas de software propuesto por Losavio. Se asume que al aplicar el método, las funcionalidades presentes en el sistema son las que necesitan los usuarios, y en consecuencia, no se incluye escenarios de aptitud (subcaracterística de funcionalidad). Posteriormente, se consideran escenarios no contemplados en el árbol inicial de utilidad. A continuación se describen los pasos a seguir para obtener el árbol de utilidad. 5.1) Seleccionar los escenarios iniciales a considerar. En este paso el grupo evaluador presenta los diferentes escenarios a considerar al resto de los participantes y se decide cuáles son los que serán incluidos en el árbol de utilidad. 5.2) Proposición de escenarios no contemplados. Este paso busca brindar a los stakeholders la posibilidad de que verifiquen si es necesario incluir en el árbol de utilidad algún otro escenario. Si se determinan escenarios de calidad que cuenten con un alto consenso de los participantes deberían ser incluidos directamente dentro del árbol de utilidad. La selección del resto de los escenarios puede realizarse por votación como propone originalmente el ATAM. Luego se organizan los escenarios de calidad introducidos no contemplados en la tabla de generación del árbol de utilidad inicial por características de calidad. Una vez obtenido los escenarios a incluir en el árbol de utilidad se procede a priorizarlos. 5.3) Priorización de los escenarios de calidad. En este paso el objetivo es ordenar los escenarios, ya que de esta manera los arquitectos cuentan con más orientación para tomar decisiones arquitectónicas, y los stakeholders pueden estar más conscientes de lo que esperan del sistema, y obtener una idea de la medida en que va a ser satisfecho. Una vez que se ha decidido incluir en el árbol de utilidad los escenarios más importantes, se procede a asignar un orden a los escenarios de calidad de un sistema utilizando dos criterios, a saber: • Evaluar el riesgo de no considerar el escenario. Se deben responder las preguntas: ¿qué pasa si este escenario no se cumple?, ¿cuántas personas se ven afectadas?, ¿es posible compensar el no responder a este escenario? • Evaluar el esfuerzo necesario para lograr cumplir con el escenario. En este caso se determina qué cambios o integraciones de nuevos componentes se deben realizar para satisfacer el escenario. Los escenarios más difíciles son aquellos en los que se debe consumir más tiempo de análisis y el resto debe ser guardado como parte del registro. Finalmente, se construye el árbol de utilidad con las salidas del paso anterior. La prioridad del escenario define en este método como un par (X, Y) en el cual X define el esfuerzo de satisfacer el escenario, y la Y indica los riesgos que se corren al excluirlos del árbol de utilidad. Los posibles valores para X y Y son A (Alto), B (Bajo) y M (Medio). El árbol de utilidad generado se toma como un plan para el resto de la evaluación que realiza el método. Indica además al equipo evaluador dónde ocupar su tiempo y dónde probar aproximaciones y riesgos arquitectónicos.

	6) *Análisis de elementos de diseño para ASBC.* En este paso se analizan los elementos de diseño identificados en el paso 4 o de nuevos elementos de diseño que puedan ser identificados, para determinar cómo influyen sobre la realización de los escenarios de calidad presentes en el árbol de utilidad generado en el paso 5. 6.1) Análisis de elementos de diseño estudiados en el paso 4. Dependiendo del tipo de elemento de diseño, la evaluación será diferente. Los componentes proveen un conjunto de puertos que se deben ir extendiendo para proporcionar nuevos servicios, los cuales a su vez pueden ser ofrecidos para que otros componentes los extiendan. Se debe evaluar un conjunto de opciones que indiquen una manera de definir una relación entre puertos (planteadas en el paso 4 y que pueden ser extendidas en este paso, considerando el árbol de utilidad generado), de acuerdo a algún criterio de calidad. Los escenarios de calidad deben ser aplicados a la arquitectura que ha sido generada. El equipo de evaluación se orienta con las preguntas de análisis propuestas por el método para determinar decisiones sobre la arquitectura. Se debe preguntar si las decisiones tomadas permiten alcanzar los escenarios de calidad planteados. Si la respuesta es negativa se deben reconsiderar cualquiera de las decisiones que han sido tomadas. 6.2) Documentación de los puntos de negociación, puntos de equilibrio, riesgos, no-riesgos y asuntos no resueltos. Las decisiones identificadas en el paso 6.1, pueden relacionarse con determinados elementos de diseño. Se deben estudiar los riesgos que introduce la decisión en particular, o si ésta contribuye a algún riesgo ya determinado. También se pueden documentar decisiones que no han sido tomadas o asuntos no resueltos que a juicio del equipo de evaluación pueden ser resueltos en un futuro estudiando con más detalle el elemento de diseño considerado.
3) Prueba	7) *Revisión del árbol de utilidad.* El MECABIC propone utilizar escenarios de calidad para representar los intereses de todos los grupos de la evaluación y confirmar que se han evaluado los requerimientos deseados de calidad. El equipo de evaluación puede facilitar la elaboración de escenarios haciendo uso del conjunto de pasos propuestos en el paso 6. Los escenarios del árbol de utilidad que no hayan sido considerados, son colocados por los stakeholders en el paquete de tormenta de ideas, lo que les da la oportunidad de revisar escenarios poco atendidos. Los escenarios generados se comparan con la lista inicialmente considerada en el árbol de utilidad. En caso de que la consideración y priorización coincida, entonces se puede decir los escenarios evaluados son adecuados para el grupo de interesados en el sistema. Si no coinciden, los nuevos escenarios y/o su prioridad deben ser reconsiderados. Cuando no se logra un acuerdo entre los dos grupos de stakeholders posiblemente no se representaron adecuadamente los intereses de los involucrados. Los stakeholders deben analizar también los escenarios que ya han sido evaluados, para verificar que hayan recibido la importancia adecuada. Una vez que el nuevo escenario ha sido considerado se pueden presentar tres casos: 1) El escenario duplica un escenario ya considerado en el árbol de utilidad. 2) El escenario pasa a ser una hoja de una rama ya existente.

	3) No existe rama para el escenario debido a que no había sido considerado previamente. Los primeros dos casos sugieren que la arquitectura de software ha sido creada de acuerdo con las expectativas de los stakeholders, mientras que en el tercer caso, se ha fallado al considerar algún aspecto de calidad importante y puede existir un riesgo a documentar. 8) Revisión de los elementos de diseño definidos. El equipo evaluador debe estudiar de qué manera los elementos de diseño analizados en el paso 6, promueven los escenarios propuestos por el nuevo grupo de stakeholders. En caso que no promuevan las características de calidad deseadas, deben ser reconsiderados.
4) Generación de la arquitectura final y reporte	9) Generación de los acuerdos. En este paso se decide cuál será la arquitectura adoptada por el sistema. Para ello se debe buscar un consenso en cuanto a las opciones que existan, en caso de que no se haya identificado alguna notablemente mejor. Si existen requerimientos de calidad que no han sido logrados se debe brindar la posibilidad de que los stakeholders acepten replantear los requerimientos de calidad que no hayan podido ser alcanzados, para aprovechar posibles ventajas de la arquitectura. Para este momento los stakeholders tienen una idea de cuáles beneficios pueden ser obtenidos mediante las alternativas que se han estudiado. Esta es una negociación crítica, ya que de fallar, implicaría la necesidad de replantear la arquitectura, y de tener éxito hay que cuidar que los requerimientos de calidad no resueltos sean equivalentes, para que los stakeholders estén satisfechos con el sistema. Finalmente, se documentan todos aquellos requerimientos de calidad para los cuales no es posible encontrar una solución, justificando las razones. 10) *Presentación de los resultados.* Para finalizar la aplicación del método se presenta un resumen de la aplicación del método, de formal oral y/o escrita. Se deben incluir entonces, los siguientes documentos a partir de los cuales se inició la evaluación: • Contexto del negocio • Requerimientos de Calidad • Restricciones • Arquitectura producida • Análisis de elementos de diseño identificados • Conjunto de escenarios negociados • Conjunto de preguntas para evaluación de atributos • Árbol de Utilidad • No-riesgos documentados Puntos sensibles y de negociación
Estos documentos a partir de los cuales se inició la evaluación son a su vez los artefactos que se generan.	

Criterios asociados a los factores de calidad – Modelo de McCall

Factor	Criterio
Correctitud.	Rastreabilidad, Completitud, Consistencia
Confiabilidad.	Consistencia, Exactitud, Tolerancia a fallas
Eficiencia.	Eficiencia de ejecución, Eficiencia de almacenamiento
Integridad	Control de acceso, Auditoría de acceso
Usabilidad.	Operabilidad , Entrenamiento, Comunicación
Mantenibilidad.	Simplicidad, Concreción
Capacidad de Prueba	Simplicidad, Instrumentación, Auto-descriptividad, Modularidad
Flexibilidad.	Auto-descriptividad, Capacidad de expansión, Generalidad, Modularidad
Portabilidad.	Auto-descriptividad, Independencia del sistema, Independencia de maquina
Reusabilidad.	Auto-descriptividad, Generalidad, Modularidad, Independencia del sistema, Independencia de maquina
Interoperabilidad.	Modularidad, Similitud de comunicación, Similitud de datos

Relación entre propiedades del producto y atributos de calidad

Propiedades del producto	Atributos de calidad
Correctitud	Funcionalidad, Confiabilidad
Internas	Mantenibilidad, Eficiencia, Confiabilidad
Contextuales	Mantenibilidad, Reusabilidad, Portabilidad, Confiabilidad
Descriptivas	Mantenibilidad, Reusabilidad, Portabilidad, Usabilidad

Atributos de Calidad – Modelo FURPS

Factor de Calidad	Atributos
Funcionalidad	Características y capacidades del programa, Generalidad de las funciones, Seguridad del Sistema
Facilidad de uso	Factores humanos, Factores estéticos, Consistencia de la interfaz, Documentación
Confiabilidad	Frecuencia y severidad de las fallas, Exactitud de las salidas

	Tiempo medio de fallos, Capacidad de recuperación ante fallas, Capacidad de predicción
Rendimiento	Velocidad del procesamiento, Tiempo de respuesta, Consumo de recursos, Rendimiento efectivo total, Eficacia
Capacidad de Soporte	Extensibilidad, Adaptabilidad, Capacidad de pruebas, Capacidad de configuración, Compatibilidad, Requisitos de instalación

Características y Sub-Características de calidad – Modelo ISO/IEC 9126

Característica	**Sub-Característica**
Funcionalidad	Adecuación, Exactitud, Interoperabilidad, Seguridad
Confiabilidad	Madurez, Tolerancia a fallas, Recuperabilidad
Usabilidad	Entendibilidad, Capacidad de aprendizaje, Operabilidad
Eficiencia	Comportamiento en tiempo, Comportamiento de recursos
Mantenibilidad	Analizabilidad, Modificabilidad, Estabilidad, Capacidad de pruebas
Portabilidad	Adaptabilidad, Instalabilidad, Reemplazabilidad

Subconjunto del Arbol de Utilidad inicial propuesto por el método

Carácterística	**Sub-Catacterística**	**Escenario**
Funcionalidad	Interoperabilidad.	El sistema posee componentes capaces de leer datos provenientes de otros sistemas. El sistema posee componentes capaces de producir datos para otros sistemas.
	Precición.	Los resultados ofrecidos por los componentes del sistema son exactos. La comunicación entre los componentes no altera la exactitud de los datos.
	Seguridad.	El sistema detecta la actuación de un intruso e impide acceso alos componentes que manejen información sensible. El sistema asegura que los componentes no pierdan datos ante un ataque (interno o externo).
	Obediencia (Compliance).	Los componentes respetan un estandar de fiabilidad.

		La comunicación entre los componente no viola los estándares de fiabilidad.
Fiabilidad	Madurez. Tolerancia a fallas. Capacidad de restablecimiento o recuperación.	Los componentes del sistema manejan entradas de datos incorrectas. Todas las operaciones ejecutadas por los componentes se realizan correctamente bajo condiciones adversas. Los componentes del sistema no fallan bajo cuertas situaciones especificadas. Ante problemas con el ambiente un subconjunto determinado de los componentes puede continuar prestando sus servicios.
Eficiencia	Tiempo de comportamiento. Recursos utilizados.	El sistema debe recibir los servicios de sus componentes en el transcurso de un tiempo indicado. Los componentes pueden compartir recursos adecuadamente. El sistema controla que ningun componente se quede sin recursos cuando los necesita.
Mantenibilidad	Habilidad de cambio, estabilidad, prueba.	Es posible verificar el estado de los componentes del sistema. El sistema brinda facilidad para adaptar un componente. El sistema debe facilitar la sustitución/adaptación de un componente.
Portabilidad	Adaptabilidad. Capacidad de instalación. Co-existencia.	El sistema debe continuar funcionando correctamente aun cuando los servicios de los componentes provistos por el ambiente varíen. Los componentes pueden instalarse fácilmente en todos los ambientes donde debe funcionar. Los componentes manejan adecuadamente recursos compartidos del sistema.

Algunas preguntas para analizar los elementos de diseño identificados

Característica	Sub-Característica	Preguntas de análisis
Funcionalidad	Precisión Interoperabilidad	-¿Puede la comunicación entre los componentes introducir imprecisiones en los servicios ofrecidos por los componentes? -¿Dónde se encuentran los componentes que permiten al sistema interactuar con otros sistemas?
Fiabilidad	Madurez	-¿Existen desiciones para minimizar el manejo

	Tolerancia a fallas	incorrecto de datos en la comunicación entre componentes? -¿Cómo se detecta el comportamiento incorrecto de un componente?
Eficiencia	Tiempo de comportamiento	-¿Cómo es la relación entre el número de componentes de las diferentes partes de la arquitectura?
Mantenibilidad	Habilidad de cambio, estabilidad, prueba	-¿Cómo se verifica el funcionamiento correcto de un componente? -¿Cómo se verifica el estado de una comunicación entre componentes?
Portabilidad	Capacidad de instalación Co-existencia	-¿Existe un mecanismo para determinar si todos los componentes del sistema se encuentran correctamente instalados? -¿Existe alguna manera de identificar las reglas para interactuar con componentes de sistemas externos a la arquitectura?

Etapas contempladas por el método de evaluación de arquitectura propuesto por Bosch

Etapa 1	
1-Selección de atributos de calidad	Deben seleccionarse aquellos atributos que se consideran cruciales para el éxito del sistema y cuya satisfacción resulte poco clara a nivel de arquitectura. Resulta necesario porque es poco factible y poco útil evaluar todos los atributos de calidad, dado que requiere una gran cantidad de tiempo.
2-Definición de los perfiles	Para cada atributo de calidad seleccionado, se definen los perfiles respectivos para efectos de la evaluación.
3-Selección de una técnica de evaluación	Para la evaluación de los atributos de calidad dependientes del diseño de la arquitectura se recomienda utilizar la evaluación basada en escenarios, así como también los modelos basados en métricas o modelos matemáticos. Los atributos de calidad operacionales (observables vía ejecución) pueden evaluarse con técnicas de simulación o modelos matemáticos. La selección de la técnica, y la implementación concreta de esta dependen del objetivo y exactitud de la evaluación.
Etapa 2	
4-Ejecución de la evaluación	Para cada atributo de calidad, las técnicas arrojan valores cualitativos.
5-Obtención de resultados	Los resultados se resumen en una tabla que contiene el nivel requerido, el nivel predicho, y un indicador, que puede tener diversos significados: si el atributo se satisface o no, si necesita ser negociado con el cliente, o

	existencia de alguna relación negativa con otro atributo de calidad. El arquitecto puede decidir acerca de la realización de transformaciones sobre la arquitectura actual, y efectuar una nueva evaluación. Una vez que concluye el proceso de evaluación, con los resultados obtenidos es posible decidir entre la continuación, renegociación o cancelación del proyecto.

Perfiles, categorías, pesos y métricas asociados a atributos de calidad según Bosch (2000)

Atributo	Perfil	Categorías	Pesos	Métricas
Mantenibilidad	Perfil de mantenimiento	Se organizan alrededor de las interfaces del sistema (sistema operativo, interfaces con el usuario, interfaces con otros sistemas). Los escenarios de cambio describen modificaciones en los requerimientos.	Indican la probabilidad de ocurrencia del cambio de un escenario en un periodo de tiempo.	-Impacto en términos de líneas de código que tiene que cambiarse. -Se requiere un estimado de líneas de código de los componentes arquitectónicos.
Desempeño	Perfil de uso	Descompone los escenarios de uso basado en los tipos de usuario y/o interfaces del sistema	Representan la frecuencia relativa del escenario.	-Funcionalidad de componentes. -Comportamiento del sistema en respuesta a los escenarios de uso en el perfil. -Promedio y pero caso de latencia por sincronización y sobrecarga en el sistema.
Confiabilidad	Perfil de uso	Confiabilidad de los componentes, genera la confiabilidad de los escenarios de uso.	Indica la robustez del sistema.	-Datos estimados de confiabilidad del componente. -Datos históricos de confiabilidad del componente.
Seguridad Interna	Perfil de seguridad	Basada en todas la interfaces del sistema.	Indica la probabilidad de fallas.	Depende del aspecto de seguridad a ser evaluado. Por ejemplo, la disponibilidad puede evaluarse en términos del

				número de veces que se ejecutan operaciones de seguridad.
Seguridad Externa	Perfil de peligro	Se organizan de acuerdo a documentos de certificación (sistemas médicos, puntos de interacción del sistema con el mundo real, o componentes críticos del sistema).	Indican la probabilidad de falla u ocurrencia de consecuencias desastrosas.	Bosch (2000) no establece ejemplos.

Pasos del método de evaluación ATAM

Fase 1:Presentación	
1-Presentación del ATAM	El líder de la evaluación describe el método a los participantes, trata de establecer las expectativas y responde las preguntas propuestas.
2-Presentacion de las metas del negocio	Se realiza la descripción de las metas del negocio que motivan el esfuerzo y aclara que se persiguen objetivos de tipo arquitectónico.
3-Presentacion de la arquitectura	El arquitecto describe la arquitectura, enfocándose en cómo ésta cumple con los objetivos del negocio.
Fase 2:Investigación y análisis	
4-Identificación de los enfoques arquitectónicos	Estos elementos son detectados pero no analizados.
5-Generación de Utility Tree	Se elicitan los atributos de calidad que engloban la “utilidad” del sistema (desempeño, disponibilidad, seguridad, modificabilidad, usabilidad, etc.), especificados en forma de escenarios. Se anotan los estímulos y respuestas, así como se establece la prioridad entre ellos.
6-Analisis de los enfoques arquitectónicos	Con base en los resultados del establecimiento de prioridades del paso anterior, se analizan los elementos del paso 4. En este paso se identifican riesgos arquitectónicos, puntos de sensibilidad y puntos de balance.
Fase 3:Pruebas	
7-Lluvia de ideas y establecimiento de prioridad de escenarios	Con la colaboración de todos los involucrados, se complementa el conjunto de escenarios.
8-Análisis de los enfoques arquitectónicos	Este paso repite las actividades del paso 6, haciendo uso de los resultados del paso 7. Los escenarios son considerados como casos de prueba para confirmar el análisis realizado hasta el momento.
Fase 4: Reporte	

9-Presentación de los resultados	Basado en la información recolectada a lo largo de la evaluación del ATAM, se presentan los hallazgos a los participantes.

Descripción de atributos de calidad observables vía ejecución.

Atributo de calidad	Descripción
Disponibilidad	Es la medida de disponibilidad del sistema para el uso.
Confidencialidad	Es la ausencia de acceso no autorizado a la información.
Funcionalidad	Habilidad del sistema para realizar el cual fue concebido.
Desempeño	Es el grado en el cual un sistema o componente cumple con sus funciones designadas, dentro de ciertas restricciones dadas como velocidad, exactitud o uso de memoria. Según Smith, el desempeño de un sistema se refiere a aspectos temporales del comportamiento del mismo. Se refiere a capacidad de respuesta, ya sea el tiempo requerido para responder a aspectos específicos o el número de eventos procesados en un intervalo de tiempo. Según Bass et al (1998),se refiere además a la cantidad de comunicación e interacción existente entre los componentes del sistema.
Confiabilidad	Es la medida de la habilidad de un sistema a mantenerse operativo a lo largo del tiempo.
Seguridad Externa	Ausencia de consecuencias catastróficas en el ambiente. Es la medida de ausencia de errores que generan perdidas de información.
Seguridad Interna	Es la medida de la habilidad del sistema para resistir a intentos de uso no autorizados y negación del servicio, mientras se sirve a usuarios legítimos.

Descripción de atributos de calidad no observables vía ejecución

Atributo de calidad	Descripción
Configurabilidad	Posibilidad que se otorga a un usuario experto a realizar ciertos cambios al sistema.
Integrabilidad	Es la medida en que trabajan correctamente componentes del sistema que fueron desarrollados separadamente al ser integrados.
Integridad	Es la ausencia de alteraciones inapropiadas de la información.
Interoperabilidad	Es la medida de la habilidad de que un grupo de partes del sistema trabajen con otro sistema. Es un tipo especial de integrabilidad.
Modificabilidad	Es la habilidad de realizar cambios a futuro sistema.
Mantenibilidad	Es la capacidad de someter a un sistema a reparaciones y evolución. Capacidad de modificar el sistema de manera rápida y a bajo costo.
Portabilidad	Es la habilidad del sistema para ser ejecutado en diferentes ambientes de

	computación. Estos ambientes pueden ser hardware, software o una combinación de los dos.
Reusabilidad	Es la capacidad de diseñar un sistema de forma tal que su estructura o parte de sus componentes puedan ser reutilizados en futuras aplicaciones.
Escalabilidad	Es el grado con el que se pueden ampliar el diseño arquitectónico, de datos o procedimental.
Capacidad de Prueba	Es la medida de la capacidad con la que el software, al ser sometido a una serie de pruebas, puede demostrar sus fallas. Es la probabilidad de que, asumiendo que tiene al menos una falla, el software fallara en su próxima ejecución de prueba.