UNIVERSIDAD TECNOLÓGICA DE BOLÍVAR

FACULTAD DE INGENIERÍAS

Título: Metodología para Evaluar la Calidad de un Producto Software de una Implementación de Internet de las Cosas

Autor: Ivan Alejandro Baños Delgado

Jurado

Jurado

Director: Juan Carlos Martínez Santos

Cartagena, Julio de 2017

Metodología para Evaluar la Calidad de un Producto Software de una Implementación de Internet de las Cosas

Ivan Alejandro Baños Delgado
Director: Juan Carlos Martínez Santos

Universidad Tecnológica de Bolívar
Facultad de Ingenierías
Programa de Ingeniería de Sistemas
Cartagena

Julio de 2017

Metodología para Evaluar la Calidad de un Producto Software de una Implementación de Internet de las Cosas

Ivan Alejandro Baños Delgado

Trabajo de grado para optar al título de

Magister en Ingeniería
Énfasis en Sistemas y Computación

Director: Juan Carlos Martínez Santos

Universidad Tecnológica de Bolívar
Facultad de Ingenierías
Cartagena

Junio de 2017

Resumen

Internet de las Cosas (IoT, por sus siglas en Inglés Internet of Things) se define en [28], como cosas "inteligentes" interactuando y comunicándose con otras cosas, ambientes e infraestructuras; esto da como resultado un gran volumen de datos generados y procesados que usan para tomar acciones que hacen la vida del ser humano mas simple. IoT no es solamente conectar todo a través de Internet. IoT, ademas, necesita que las cosas interactuen entre si creando ambientes capaz de tomar decisiones por si solas. Por ejemplo, podemos llenar una nevera de sensores capaces de obtener la cantidad de cada producto en una nevera, sin embargo, el verdadero reto es hacerlo lo suficientemente inteligente para que realice la compra del mercado por si sola.

Para que IoT tenga mayor inclusión en el mundo, se deben superar ciertos obstáculos y amenazas de la tecnología actual. En [28], listan ciertos obstáculos y amenazas que son explicados a continuación.

El primer obstáculo se debe a la heterogeneidad de la tecnología cosas, protocolos de comunicación entre lo mas importante, por lo que las aplicaciones tienden a ser complejas. Esto hace que realizar aplicaciones de buena calidad sea un reto. El segundo obstáculo se debe a la gran cantidad de elementos conectados, por lo que el consumo energético pasa a ser un factor al momento de tomar decisiones en el desarrollo e implementación de proyectos.

Una amenaza son los riesgos de seguridad informática que se pueden presentar, dado la gran cantidad de canales de comunicación que puede aparecer en una implementación de IoT y que las cosas debido a su naturaleza (consumo de energía, capacidad de almacenamiento, entre otros) no necesariamente van a poder soportar todos los algoritmos y protocolos de seguridad. Por lo que, todos los escenarios de seguridad deben ser reconsiderados.

Las aplicaciones deben ser lo suficientemente sencillas en su configuración para que pueda ser comercializada en masa. Este proceso debe ser transparente al usuario final. La confianza en las aplicaciones se vuelve un reto dado que debido a la limitación de recursos en las cosas, los protocolos de verificación de errores comúnmente usados pueden generar problemas de rendimiento. Ademas, cuando ocurren errores, tareas de reparación y recuperación son complicadas.

Debido a lo mencionado, IoT necesita que todos los interesados pongan de su parte para poder tener implementaciones de calidad. Una forma de ayudar es utilizando arquitecturas de software que se amolden a la solución y que puedan garantizar calidad en el software. El primer paso para garantizar calidad es tener un mecanismo para estimar el nivel de calidad.

En [9] se explica que la forma para poder cumplir con las metas trazadas para el software es teniendo una forma para poder medir su calidad. También explica que, teniendo la posibilidad de medir, se puede controlar y mejorar el desempeño del software. Tener una buena herramienta de medición es importante debido que nos puede dar un panorama particular de cuales son los factores que están afectando la calidad del software y como podemos mejorar ese factor sin afectar o afectando lo menos posible el resto de la implementación.

Con el modelo propuesto buscamos tener una guía para verificar la calidad de software en implementaciones de IoT. Esta metodología puede ser ajustable a cada dominio particular del IoT y ademas agrega simplicidad al momento de aplicarla.

Para la obtención del modelo, nos basamos en la metodología de investigación (DSRM, por sus siglas en inglés) propuesto en [37]. El proceso se dividió en cinco partes. En la primera parte se realizó una revisión de la literatura, para obtener las características que afectan a la calidad en implementaciones del IoT. En la segunda parte se realiza una comparación entre las metodologías y modelos existentes de calidad. En la tercera parte se propuso un modelo de calidad con las métricas y la definición de un algoritmo para la aplicación de estas. en la cuarta parte se realizó una aplicación para mostrar la forma como funciona el algoritmo. Por ultimo, en la quinta parte se realizó la evaluación de la métrica comparándola con otros modelos de

calidad.

El modelo propuesto fué presentado en el VII Festival Internacional de la Ciencia y la Cultura XII Encuentro de Investigadores en la Red de Instituciones de Educación Superior del Caribe (RIESCAR), el cual se realizó en la Fundación Tecnológica Antonio de Arévalo (TECNAR), Noviembre 2016. Adicionalmente, se presentó un articulo en el Colombian Conference on Communications and Computing 2017 (COLCOM 2017), que a la fecha de la entrega de este documento se encontraba abajo revisión. Por último, el modelo propuesta fué usado para medir la calidad de un desarrollo IoT presentado en Workshop on Engineering Applications (WEA 2017) titulado Ïrrigation system for oil palm in Colombia - An Internet of Things approach".

Agradecimientos

A mis padres por el impulso económico y anímico que me dieron al iniciar la maestría, a ellos les debo mas que solo mi formación. A mi futura esposa por siempre creer en mi potencial y estar presente en todo momento.

Al Centro de Enseñanza y Apropiación del Internet de las Cosas (CEA-IoT), por el apoyo prestado en el ultimo año de mi investigación y en momentos claves durante la ejecución del proyecto.

A mi director Juan Carlos Martínez Santos, por su perseverancia y paciencia que tuvo siendo mi director, ademas de sus aportes y consejos para poder terminar con este proyecto. A Edwin Alexander Puerta, por ser un gran apoyo y por preocuparse siempre por saber como estaba y como iba el proyecto. A Luz Stella Robles, por su guia prestada durante los dos años de la maestría.

Índice general

Índice de figuras

Índice de tablas

Capítulo 1.
Introducción

Diferentes arquitecturas de software pueden ser utilizadas para resolver un mismo problema por medio de una aplicación IoT. Cada arquitectura tiene ventajas y desventajas con respecto a las otras teniendo en cuanta diferentes características de calidad. La selección de una de ellas debe hacerse dependiendo de unos criterios mínimos de selección que dependen del contexto del problema y de sus características y restricciones. Sin embargo, el problema radica en que las metodologías no están planteadas para el caso del Internet de las cosas.

1.1. Planteamiento del problema.

Hoy en día, cada arquitectura obtiene un valor distinto para todos los atributos de calidad cuando se ataca el mismo problema. Por esta razón la buena escogencia de una arquitectura puede impulsar una aplicación mientras que una arquitectura "mala" podría condenar al fracaso el proyecto. En consecuencia, se hace necesario establecer criterios de selección que faciliten el proceso de comparación de arquitecturas IoT con respecto a atributos que garanticen calidad.

Por lo que es preciso preguntar ¿Cuáles son los pasos a seguir para diseñar una

metodología que nos permita garantizar atributos de calidad en la arquitectura escogida para el desarrollo de una aplicación IoT?

Toda solución software tiene una arquitectura, aunque esta no siga ningún patrón o no se halla contemplado antes de realizar el diseño y desarrollo del proyecto. Si no se realiza la arquitectura de software antes de iniciar con el diseño, puede que en alguna parte del proyecto o cuando ya el proyecto esté en producción se generen inconvenientes que puedan afectar el tiempo de finalización, el presupuesto o en el caso más grave tener que re diseñar un módulo o todo el proyecto.

Sin embargo, el problema no acaba con definir la arquitectura. Si la arquitectura escogida no se adapta a la naturaleza del proyecto o no satisface las características y restricciones particulares del proyecto, el software resultante no va a tener la calidad esperada.

Tener un proyecto de baja calidad puede derivar problemas en algunas de las características del sistema. Por ejemplo, si la autenticación, que es una sub-categoría de la seguridad, es de baja calidad, se puede traducir en que el software existe la posibilidad de que un cliente no sea quien dice ser. En IoT, donde las cosas funcionan como clientes, le da cabida a pensar que la información no viene de la cosa que creemos que es.

Otro ejemplo podría ser la tolerancia a fallos, esta sub-categoría pertenece a la fiabilidad. Si el sistema tiene una baja tolerancia al fallo y el sistema no es capaz de auto restablecerse o advertir sus fallas, puede que se crea que el sistema está funcionando

bien cuando no está en su máximo potencial.

Por los ejemplos anteriores se puede observar que tener mala calidad de software incurre en problemas que pueden afectar de forma significativa el proyecto. Como se ha explicado antes, para una misma solución pueden existir varias arquitecturas que se acomoden a la solución y puede que ambas sean de calidad.

Si se selecciona alguna de las dos, que son de calidad, no van a existir problemas. Sin embargo, es posible que si aplicamos alguna de las dos nos vaya mejor en atributos como mantenibilidad, adecuación funcional y usabilidad, que puedan mejorar la coexistencia con la aplicación y se vea reflejado en eficiencia, ahorro, tanto de energía como monetario, que a la final es el objetivo de las aplicaciones de IoT.

1.2. Objetivos

Establecer una metodología orientada a la evaluación de calidad en arquitecturas de software, para aplicaciones implementadas bajo el concepto de Internet de las Cosas.

Para lograr el anterior objetivo se proponen los siguientes objetivos específicos:

- Identificar los atributos que permiten garantizar la calidad del software en aplicaciones de Internet de las Cosas.
- Seleccionar un grupo de metodologías de evaluación de arquitecturas que se adecuen a los criterios de calidad identificados en el objetivo anterior.

- Proponer una metodología de calidad de software para el contexto del Internet de las Cosas.

- Evaluar la metodología propuesta en aplicaciones de Internet de las Cosas de código abierto y que se sepa de antemano la calidad de estas.

Capítulo 2.
Marco Teórico.

En este capítulo se describen los conceptos básicos para entender el proyecto. En la primera sección se explica el concepto de IoT, en la segunda sección se listan las aplicaciones e investigaciones actuales y por ultimo se explica los conceptos de arquitectura de software y calidad de software.

2.1. Internet de la Cosas, IoT

Como hemos mencionado antes, IoT es un ambiente donde cosas actúan como usuarios de un sistema. Para poder tener una idea del poder que IoT tiene tenemos que ir los años 80 donde se creo la disciplina de la computación ubicua que es brevemente explicado en [21]. Para Gubbi et al. la definición de computación ubicua es donde un ambiente inteligente donde sensores, actuadores, displays y otros elementos computacionales están embebidos armónicamente en objetos de uso cotidiano conectados entre si a través una red.

El primer uso del termino fue por Gershenfeld en su libro [19]. En su libro habla de como las cosas están llegando a tener menor tamaño pero siguen siendo funcionales llegando a afectar todos los aspectos de nuestra vida, como una cafetera que pueda

aprender del hábito de las personas para este saber que clase de café quiere la persona por la hora en que lo pide y su estado de ánimo.

Para que el IoT funcione, se debe diferenciar bien cada módulo que interactua en las soluciones de este tipo. Por lo general, existen cuatro módulos distintos. El primer módulo se refiere a la cosa como tal. Esta puede tener dos funciones, sensar o actuar y ademas debe tener una forma de comunicarse con otras cosas o dispositivos locales o inclusive a través de Internet. El segundo módulo es conocido como gateway. Este se encarga de pre-procesar la información y enviarla a través de Internet al servidor central. El tercer módulo es el servidor central que se encarga de procesar y analizar la información y la preparara para su visualización. El ultimo modulo es la capa de visualización donde por lo general son aplicaciones web que muestran información en tiempo real.

En [13], Evan realiza una buena conclusión cuando dice que si combinamos las habilidades de sensado, colección, transmisión y distribución de IoT con la forma en que se procesa la información tendríamos el conocimiento y sabiduría necesarias no solo para sobrevivir sino también para prosperar por siglos.

En IoT se busca que las cosas tengan poder computacional. Este poder computacional no tiene que ser muy potente, solamente lo suficiente para que realice las actividades queridas y no le agregue mucho valor económico. Basándose en eso, Fleisch en, [17], habla acerca de que IoT es solo una extensión del actual Internet y que en realidad debería llamarse Red de las cosas. El problema es que Fleich ve todo a nivel

de aplicación, como servicios basados en internet, dejando por fuera la complejidad ganada por sumarle, ademas del software, el hardware y los canales de comunicación que hace que se mejore las soluciones que no se podrian mejorar si solo tenemos internet.

Sin embargo, Fleisch utiliza un concepto de drivers para explicar como IoT agrega valor a la Internet actual. Explica que existen siete drivers que se explicaran en los siguientes párrafos.

El primer driver es el sensor de proximidad manual, que se ve en las entradas de universidades y para el transporte masivo. Esto le agrega valor a los negocios que quieren automatizar acciones cuando el sensor de proximidad pase por un detector. Estas acciones en el caso del transporte público sería cobrar el pasaje de la persona que pasa la tarjeta.

El segundo driver es el sensor de proximidad automático, que es cuando la acción se ejecuta cuando dos objetos están dentro de un rango de distancia es mas corto que un límite. El tipo de aplicaciones que se pueden generar pueden ser actualización automática de inventario, pudiendo reducir costos, fallas en el proceso e incluso fraude.

El tercer driver es ejecutar acciones a partir de datos sensados. Colocando ciertas reglas, un sistema puede tomar distintas acciones dependiendo del estado de las variables que afecten el sistema. Ejemplos de este tipo de driver son el sistema de riego automático, parrillas eléctricas que se ajusten para mantener los niveles de

consumo de electricidad bajos, monitorear los signos vitales de un paciente que esté en un hospital o en su casa. Las soluciones presentarían una red de sensores capaz de obtener mas información que la que un humano podría obtener, en un tiempo continuo. Con esto lograríamos obtener información que nos pueda dar nuevas pistas de como mejorar cada problema.

El cuarto driver es la seguridad automática de productos. Las cosas a asegurar podrían tener embebido un mini-computador que soporte alguna tecnología de seguridad. Ejemplos de este driver podría ser, encontrar una manera de verificar que la persona que este abriendo un automóvil con una llave sea una persona autorizada para poder abrir el vehículo.

El quinto driver es la retroalimentación directa al usuario. Esto lo vemos actualmente con los dispositivos que tienen una luz encendida para decirle al usuario que está funcionando. Un uso muy bueno para este driver es que los objetos perdidos puedan avisar de alguna forma donde se encuentran.

El sexto driver es la retroalimentación extensiva al usuario. Ya sea utilizando el celular u otro dispositivo. Este dispositivo puede funcionar de enlace entre la aplicación y el sensor para sacar provecho, ya sea verificando la información o ejecutando alguna acción.

El ultimo driver es la retroalimentación para cambiar el pensamiento. Se utiliza para cambiar la forma como las personas piensan, mostrándole al usuario cierta métrica para que este pueda tomar acciones. Por ejemplo, si se mantiene informado

al usuario cuanta agua ha gastado, este podrá ahorrar agua de modo que no tenga que pagar mucho dinero en el próximo recibo y de esa forma ayudar a su economía e incluso al medio ambiente. Otro ejemplo es saber como responden las personas a los productos vendidos en las máquinas expendedoras, para que estas sepan que productos vender y como poder realizar las promociones.

Estos drivers cubren de manera general los aspectos que pueden cambiar nuestras vidas si le embebemos mas aplicaciones IoT. No obstante, para que el IoT se masifique, primero tiene que probar que es necesario para la humanidad. Por lo tanto, las investigaciones y las inversiones que se realizan en IoT apuntan a ver como se puede ver afectada la economía, el tiempo y la salud de las personas y empresas. En la siguiente sección se listan algunas investigaciones y aplicaciones actuales para mostrar el potencial de este tipo de soluciones.

2.2. Aplicaciones e investigaciones actuales

La variedad de aplicaciones viene dado por la forma en como afectan la vida de las personas. Estas vienen separadas por grupos que han venido apareciendo en muchas investigaciones. Los grupos aquí presentados vienen dados por una separación realizada en el centro de enseñanza y apropiación del Internet de las Cosas en sus líneas de trabajo [1]. En la Tabla 2.1 se listan cada una de las lineas de trabajo.

Tabla 2.1: Líneas de trabajo en el CEA-IoT

Línea de trabajo
Salud
Logística
Industria
Vestibles
Seguridad
Agroindustria y medio ambiente
Gobierno

2.2.1. Salud

Las aplicaciones enmarcadas en salud utilizando IoT se centran en utilizar el poder de recoger información de forma masiva para mantener un estado de los pacientes en tiempo real, administrar la información de los pacientes y tener una visión general del sistema de salud [20], [7], [5], [46]. Este tipo de aplicaciones le agregan valor al sistema de salud, debido a que los médicos tendrán información veraz y cuando sea necesaria de sus pacientes tanto en los hospitales como en casa. Ademas ayuda a pacientes con problemas graves de salud a que se mantenga su estado actualizado y obtener ayuda oportuna en casos de emergencia.

En [27], habla de 10 formas distintas de como lograr tener un ambiente inteligente para ayudar en la salud. En los que destaca el IoT movil (m-IoT) que son dispositivos móviles, sensores y tecnologías de comunicación al servicio de los pacientes.

Los retos que tienen este tipo de aplicaciones vienen ligados a la interconectividad y seguridad, dado que el estado actual genera limitaciones de este tipo, como se expresa en [42] y en [27].

2.2.2. Logística

En la logística, IoT tiene como factor que agrega valor el que se pueden automatizar procesos logísticos gracias a que se le pueden agregar partes tecnológicas a objetos cotidianos.

En los procesos de la gestión de cadena de suministros, la logística tiene un rol importante. El utilizar sensores y dispositivos RFID para delegar ciertas funciones a los productos como tal, puede ayudar en la toma de decisiones de forma localizada, descentralizando la información y teniendo ahorros en los tiempos de respuesta [15].

Existen otras áreas donde la logística utilizando IoT puede mejorar procesos. Como el transporte, la industria, inventario entre otros. Se podrán tener contacto en tiempo-real con los elementos dentro de la cadena de suministros. Un ejemplo es poder controlar todos los movimiento de un automóvil, la localización actual y otros datos para predecir su destino siguiente [12].

En [12] también agregan que los retos en logística viene da do por la seguridad y la protección de la privacidad.

2.2.3. Industria

Las aplicaciones que se enmarcan en la industria es conocida como IIoT (debido a sus siglas en ingles Industrial IoT) [23]. Esta busca resolver problemas con respecto a seguridad, optimización de procesos, ahorro de energía y análisis de grandes cantidades de datos, entro otros.

En IIoT uno de los elementos mas importantes son las redes de sensores inalámbricos (WSN, por sus siglas en ingles). Uno de los casos particulares donde se ve su utilidad es en las terminales marítimas, donde las redes de sensores son utilizadas para intercambio y control de datos [10]. Pudiendo realizar acciones como reducción de almacenamiento de contenedores, monitoreo en tiempo real del estado de carga con características especiales, o la seguridad [41], [44].

Manufactura es otro de los campos emergentes con el IIoT. Se estan realizando trabajos en área de producción de poliestireno [34], reduciendo los tiempos de parada de planta. Intel y Mitsubishi crearon de un sistema de fabrica autónoma que utiliza IoT y análisis de grandes cantidades de datos [25], ahorrando tiempo y dinero.

2.2.4. Vestibles

Los dispositivos vestibles ya son una realidad. Sin embargo, la habilidad de obtener información del ambiente no es suficiente para cumplir con el propósito del IoT. Para aprovechar al máximo esto deben ofrecer además la posibilidad de utilizar los datos censados para tomar decisiones que ayuden a las personas que los utilizan.

Comercialmente se tienen una gran variedad de productos. De acuerdo con [45], se prevé que esta industria tendrá un valor de 12 billones de dólares. Entre los más utilizados son los productos para ejercicio y salud.

Entre las investigaciones sobre vestibles se destacan las relacionadas con el ámbito de la salud. En [14] se muestra como mejorar la obtención de información dentro de la sala de cirugía, para que esta pueda ser obtenida en tiempo real.

2.2.5. Seguridad industrial

Las aplicaciones de seguridad tratan de solucionar problemas como la vigilancia de espacios, localización de personas y recursos, mantenimiento de equipos e infraestructuras, sistemas automáticos de alarmas, entre otras [30].

Un ejemplo es presentado en [12] donde se utiliza IoT para realizar la operación de una mina más segura. Para prevenir y reducir accidentes en las minas, se utilizan comunicaciones inalámbricas para mantener conectados la superficie con la cueva. Así las compañías podrán mantener localizados a los mineros y analizar datos críticos para mejorar medidas de seguridad.

Otro caso expresado en el mismo articulo es el uso de IoT para la detección temprana de incendios y realizar llamados de emergencia tempranos para posibles desastres.

2.2.6. Agro-industria y Medio Ambiente

En agro-industria la mayoría de las implementaciones tecnológicas que utilizan el concepto de IoT están ligadas al control y predicción para generar ahorro en utilización de recursos tanto humanos como de materiales.

Un ejemplo es explicado en [33], donde utilizan controles remotos para manejar eficientemente los niveles de agua. Esto ha reducido el tiempo de espera de granjeros, la intervención manual; incrementado la eficiencia de la operación.

Esto hace que la irrigación sea un buen objetivo para el tipo de impacto que

quiere generar el IoT.

2.2.7. Gobierno

En la categoría de gobierno, el CEA-IoT se refiere a todo lo relacionado a lo que se puede realizar a través de administraciones del estado para tener impacto socio-económico a nivel general de una ciudad o del país. Este concepto es conocido como ciudades inteligentes (smart cities), siendo una de las líneas de trabajo que puede generar más impacto para el IoT. En [30], Khan menciona que el IoT puede ayudar a diseñar ciudades inteligentes, realizando aplicaciones como monitoreo de la calidad del aire, descubrir rutas de emergencia, uso eficiente de la maya de luces, irrigación de jardines públicos, etc.

Una de las áreas donde se encuentra un potencial grande para el IoT es en el transporte. Utilizando redes avanzadas de sensores, informaciones de terceros y otras metodologías de ayuda, se puede obtener control y administración de la red de transporte urbano de una ciudad. Pudiendo realizar trazados de emergencia, multas sin necesidad de parar al conductor, chequeo de violaciones a las leyes de transito, reduccción de la contaminación vehicular, sistemas anti-robo, evasión de trancones, reporte de accidentes de trafico, etc [30].

Otro tipo de aplicaciones es utilizar redes de sensores para realizar mediciones y monitoreo del uso de agua, gas, luz para tomar decisiones en ahorros y cobros a los usuarios[30].

2.3. Arquitectura de software

Existen muchas definiciones para el termino arquitectura de software. esto es debido a su naturaleza, dado que su formulación viene basada a partir de su practica. En el Instituto de Ingeniería de Software tienen una sección donde se muestran una lista de definiciones de mas de 100 autores [2]. Por lo que llegar a una definición general aceptada es una tarea difícil.

Se escogió la definición que Bass coloca en Software architecture in practice [6], dado que este libro ha servido de manual para muchas investigaciones en cada una de sus versiones. Bass define Arquitectura de Software como el conjunto de estructuras que describen un sistema y que ayuda a entender como este se comportará. Estas estructuras son elementos arquitectónicos (componentes y conectores), las relaciones entre ellos y sus propiedades importantes.

La arquitectura de software es un conjunto de representaciones. Cada una de estas muestra el sistema de la forma en que cada interesado pueda observar los aspectos que le interesan del proyecto. Para que de esta forma los aspectos de calidad del software puedan alcanzar los limites mínimos esperados.

Existen diseños arquitectónicos que por resolver el mismo problema muchas veces que se vuelven estándares. Estos se conocen como estilos arquitectónicos y sus definiciones particulares se conocen como patrones arquitectónicos.

Estos dos términos han generado controversia a lo largo del estudio de arquitectura de software. Estos algunos autores los tratan como uno sólo término. Un estilo

arquitectónico define como se va a estructurar la arquitectura, definiendo el vocabulario de los componentes, conectores y el conjunto de restricciones y como pueden ser combinadas. Mientras que, un patrón es una solución genérica preestablecida para realizar la estructuración de la arquitectura. En un patrón se especifica dependiendo de este como van a ser estos componentes, conectores y restricciones.

Toda implementación de software sigue una arquitectura, así esta no sea conocida. Esto puede suceder por múltiples razones. Esto puede generar problemas dado que no se verifica que se cumplan los requerimientos de cada interesado o no se prevean errores puntuales, pudiéndose ver afectada de manera negativa la calidad del proyecto.

Los dos párrafos anteriores nos da a entender que no todas las arquitecturas son buenas, dependiendo del problema. Ya sea, porque no se ajusta a la naturaleza del proyecto, o a los requerimientos ya establecidos de las negociaciones con los interesados.

Uno de los puntos que Bass explica, en [6], que hay que tener en cuenta para saber si una arquitectura se ajusta a un problema o proyecto es que la arquitectura debe ser basada en una lista de requerimientos de atributos de calidad. Así mismo, que la arquitectura debe ser evaluada por su habilidad de satisfacer estos requerimientos. Esto se puede lograr utilizando patrones y tácticas conocidas. Bass muestra unos ejemplos que están listados a continuación.

- Si un sistema requiere alto desempeño, entonces se necesita prestarle atención al los comportamientos dependientes del tiempo de los elementos, al uso de los

recursos compartidos y la frecuencia y volumen de la comunicación interna de los elementos.

- Si la capacidad de modificación es portable, entonces es necesario prestarle atención al asignarle responsabilidades a los elementos para que la mayoría de los cambios en el sistema tenga el menor impacto posible.

- Si el sistema necesita ser muy seguro, entonces se deben tener mecanismos de protección de los mensajes entre elementos agregando nuevos elementos especializados para la seguridad en la arquitectura.

- Si el sistema debe ser escalable, entonces se necesita localizar el uso de recursos para poder facilitar la introducción de nuevas tecnologías.

- Si el sistema requiere que sus elementos sean reusables, entonces se debe evitar alto-acoplamiento para que al extraer un elemento se tenga el menor impacto dentro del sistema.

Los punto anteriores son netamente arquitectónicos. Sin embargo, tener una buena arquitectura no siempre es suficiente para alcanzar los atributos de calidad. También se debe tener precaución en las decisiones tomadas al momento de diseño, implementación y despliegue.

2.3.1. Atributos de calidad

Un software tiene muchas características que pueden afectar la calidad. Estas características deben seguir cierta naturaleza para que el sistema realice las tareas de la forma como los interesados quieren que se realicen. Estas características tienen que ser medidas de alguna forma, para saber en que grado de satisfacción se están cumpliendo lo pactado por los interesados. La forma de medir las características es a través de los atributos de calidad.

Los atributos de calidad son unidades medibles que sirven para identificar que tan bien se satisfacen los requerimientos de calidad establecidos en las negociaciones con los interesados.

Los requerimientos de atributos de calidad deben ser precisos y medibles. En [6] los especifican en diferentes partes.

1. Fuente de estímulo. Es alguna entidad (un humano, sistema o algún otro actor) que genera estímulo.

2. Estímulo. Es una condición que requiere respuesta cuando llega al sistema.

3. Ambiente. El estimulo ocurre bajo ciertas circunstancias que definen un ambiente. El sistema puede estar sobrecargado o en operación normal, o en algún estado relevante.

4. Artefacto. Un artefacto es estimulado. Esto puede ser una colección de sistemas, todo el sistema, o partes de este.

5. Respuesta. Es la actividad que se lleva a cabo como resultado de la llegada de un estimulo.

6. Medida de la respuesta. Cuando ocurre una respuesta, esta debe ser medible de alguna forma para que el requerimiento pueda ser probado.

Bass habla de diferentes escenarios que vienen siendo representados por características de calidad. En las siguientes secciones vamos a explicar estas características de calidad. En las sub-secciones siguientes se exponen como en [6] se explican cada característica de calidad.

2.3.2. Disponibilidad

La disponibilidad se refiere a la propiedad de un software para llevar a cabo las tareas cuando se necesite. Bass trata de agregarle un poco mas a la definición explicando que no solo sea el tiempo que este disponible el software sino que sea la capacidad del sistema tiene para enmascarar o reparar errores para que el tiempo donde el sistema este sin servicio no supere el umbral máximo para satisfacer los requerimientos.

2.3.3. Interoperabilidad

Interoperabilidad es el grado en el cual dos o mas sistemas pueden intercambiar de forma exitosa mensajes en un contexto en particular. Esto implica no solo el cambio de datos, sino ademas la correcta interpretación de estos.

2.3.4. Habilidad para ser modificado

La habilidad para ser modificado no solo es el grado en el cual un sub-sistema puede ser cambiado para cumplir un cambio en los requerimientos, sino ademas que afecte en lo mas mínimo el resto del sistema.

2.3.5. Desempeño

Desempeño la habilidad en el cual el sistema realiza las operaciones en el tiempo establecido en las negociaciones con los interesados. No todas las operaciones necesitan un tiempo de respuesta "rápido" pero en general deben tener un rango donde el tiempo de respuesta es optimo.

2.3.6. Seguridad

Es la habilidad de un sistema para proteger la información y los datos de personas no autorizadas que quieren tener acceso a estos, sin negar el acceso a personas autorizadas. Cuando una acción sucede con la intención de causar daño se conoce como ataque. Existen muchos ataques diferentes, dependiendo de como estos quieran afectar el sistema. Según el ataque existen diferentes categorías para la seguridad. Estas son confidencialidad, integridad y disponibilidad.

1. Confidencialidad es la propiedad que tiene un sistema para no exponer la información de agentes no autorizados.

2. Integridad es la propiedad de un sistema para mantener la información sin ser cambiada por personas no autorizadas.

3. Disponibilidad es la propiedad del sistema para mantener el sistema disponible para personas autorizadas. Por ejemplo, no permitir que exista un ataque de negación de servicios (DoS, por sus siglas en ingles Denial of Service).

Otras características para mantener la seguridad son autenticación, no repudio y autorización.

1. Autenticación es la propiedad de un sistema para verificar que los agentes que participan en una transacción son los que realmente dicen que son.

2. No repudio es la propiedad de un sistema que permite verificar que el agente que envió un mensaje no pueda mas adelante negar haberlo enviado y que el agente que recibió el mensaje no pueda negar mas adelante haberlo recibido.

3. Autorización es la propiedad de un sistema que limita las acciones a un agente a solo las que esta autorizado para realizar.

2.3.7. Capacidad de ser probado

La capacidad de ser probado se refiere a la facilidad que tiene el sistema para demostrar sus faltas a través de un proceso de prueba. La capacidad de ser probado, es específicamente, la probabilidad que el sistema falle en la próxima ves que se ejecute el proceso de prueba.

2.3.8. Usabilidad

Usabilidad es la habilidad de un sistema para que una persona que nunca ha utilizado el sistema pueda realizar una actividad sin ayuda externa. La usabilidad se describe en las siguientes áreas.

- Aprender características del sistema. Ayudar en lo más posible a un usuario que no ha utilizado una característica a aprender a usarla.
- Utilizar el sistema de una forma más eficiente. Ayudar al usuario a ser mas eficiente mientras utilice una característica del sistema.
- Minimizar el impacto de los errores.
- Adaptar el sistema a las necesidades del usuario. Un ejemplo seria en utilizar casillas de auto-completado en los formularios.
- Incrementar la satisfacción y confianza en el sistema. Ayudar al usuario a darle seguridad de que se están haciendo las cosas bien. Un ejemplo seria colocar un mensaje de carga cuando una acción larga se este ejecutando.

2.3.9. Variabilidad

Bass define variabilidad como una forma especial de capacidad de modificación. Se refiere a la capacidad de un sistema y sus materiales de especificación y soporte para soportar un conjunto de variables de una forma planificada. Su meta es poder mantener de forma fácil un producto en un periodo de tiempo.

2.3.10. Portabilidad

Portabilidad también es definido por Bass como una forma especial de capacidad de modificación. La portabilidad se refiere a que tan fácil es hacer que el producto se ejecute en una plataforma distinta para el cual fue construido.

2.3.11. Capacidad para ser desarrollado de forma distribuida

Los sistemas deben soportar ser desarrollados de forma distribuida. Los problemas que se deben solucionar es la coordinación de las actividades. Esta coordinación debe ser lo mas mínima posible para no agregar complejidad al momento de negociar la forma de comunicación entre los módulos de cada equipo.

2.3.12. Escalabilidad

Escalabilidad es la capacidad de un sistema para utilizar recursos adicionales de forma eficiente, mejorando alguna cualidad del sistema. Agregar dichos recursos no debe causar esfuerzo adicional ni debe afectar la funcionalidad del sistema.

2.3.13. Facilidad de despliegue

La facilidad de despliegue es la capacidad de un producto para ser ejecutado de forma sencilla en su ambiente y la facilidad en ser invocado.

2.3.14. Movilidad

Movilidad es la capacidad de un sistema para acomodarse a los problemas de movimientos y de gastos que tiene su plataforma. Algunos ejemplos serian como

lidiar con el tamaño, el tipo de pantalla, las entradas del dispositivo, el ancho de banda soportado por el dispositivo, la cantidad de batería, etc.

2.3.15. Capacidad de monitoreo

La capacidad de monitoreo es la capacidad que tiene un sistema para ser monitoreada mientras está en ejecución.

2.3.16. Seguridad externa

Han existido casos donde por alguna circunstancia el sistema ejecuta tareas que no necesariamente son errores que terminan realizando alguna actividad que terminan afectando su ambiente de manera negativa, donde incluso han habido consecuencias mortales. Por lo que la seguridad externa es una preocupación.

Seguridad externa es la capacidad de un software para no entrar en un estado que pueda ocasionar daño, lesiones o la muerte de personas, animales u objetos dentro del ambiente del software o recuperar su estado seguro y recuperar el daño si se entró a el estado dañino.

2.4. Modelo ISO/IEC 25010

El Modelo ISO/IEC es el modelo de calidad para sistemas y software que se presenta en las normas ISO/IEC 25000, descrito en [24].

Las normas ISO/IEC, conocida como SQuaRE (por una combinación de palabras en Inglés System and Software Quality Requirements and Evaluation). Son la evolu-

ción de las normas ISO/IEC 9126 y ISO/IEC 14598. Tienen como objetivo la creación de un marco de trabajo común para evaluar la calidad de un producto software.

El modelos ISO/IEC 25010 muestra un modelo jerárquico de características del software donde se puede observar como en el tope se encuentra la calidad general del producto software y luego un grupo de ocho características que a su vez tiene cada una un grupo de sub-características asociadas. El modelo de calidad del producto definido por la ISO/IEC 25010 se puede observar en la Figura 2.1.

Figura 2.1: Modelo de calidad para sistemas y software ISO/IEC 25010 [24]

En las sub-secciones siguientes se exponen como en [24] se explican cada característica y sus sub-características.

2.4.1. Adecuación funcional

Representa la capacidad del producto software para proporcionar funciones que satisfacen las necesidades declaradas e implícitas, cuando el producto se usa en las condiciones especificadas. Esta característica se subdivide a su vez en las siguientes sub-características:

- Completitud funcional. Grado en el cual el conjunto de funcionalidades cubre todas las tareas y los objetivos del usuario especificados.
- Corrección funcional. Capacidad del producto o sistema para proveer resultados correctos con el nivel de precisión requerido.
- Pertinencia funcional. Capacidad del producto software para proporcionar un conjunto apropiado de funciones para tareas y objetivos de usuario especificados.

2.4.2. Eficiencia de desempeño

Esta característica representa el desempeño relativo a la cantidad de recursos utilizados bajo determinadas condiciones. Esta característica se subdivide a su vez en las siguientes sub-características:

- Comportamiento temporal. Los tiempos de respuesta y procesamiento de un sistema cuando lleva a cabo sus funciones bajo condiciones determinadas en relación con un banco de pruebas establecido.

- Utilización de recursos. Las cantidades y tipos de recursos utilizados cuando el software lleva a cabo su función bajo condiciones determinadas.

- Capacidad. Grado en que los límites máximos de un parámetro de un producto o sistema software cumplen con los requisitos.

2.4.3. Compatibilidad

Capacidad de dos o más sistemas o componentes para intercambiar información y/o llevar a cabo sus funciones requeridas cuando comparten el mismo entorno hardware o software. Esta característica se subdivide a su vez en las siguientes sub-características:

- Coexistencia. Capacidad del producto para coexistir con otro software independiente, en un entorno común, compartiendo recursos comunes sin detrimento.

- Interoperabilidad. Capacidad de dos o más sistemas o componentes para intercambiar información y utilizar la información intercambiada.

2.4.4. Usabilidad

Capacidad del producto software para ser entendido, aprendido, usado y resultar atractivo para el usuario, cuando se usa bajo determinadas condiciones. Esta característica se subdivide a su vez en las siguientes sub-características:

- Capacidad para reconocer su adecuación. Capacidad del producto que permite al usuario entender si el software es adecuado para sus necesidades.

- Capacidad de aprendizaje. Capacidad del producto que permite al usuario aprender su aplicación.
- Capacidad para ser usado. Capacidad del producto que permite al usuario operarlo y controlarlo con facilidad.
- Protección contra errores de usuario. Capacidad del sistema para proteger a los usuarios de hacer errores.
- Estética de la interfaz de usuario. Capacidad de la interfaz de usuario de agradar y satisfacer la interacción con el usuario.
- Accesibilidad. Capacidad del producto que permite que sea utilizado por usuarios con determinadas características y discapacidades.

2.4.5. Fiabilidad

Capacidad de un sistema o componente para desempeñar las funciones especificadas, cuando se usa bajo unas condiciones y periodo de tiempo determinados. Esta característica se subdivide a su vez en las siguientes sub-características:

- Madurez. Capacidad del sistema para satisfacer las necesidades de fiabilidad en condiciones normales.
- Disponibilidad. Capacidad del sistema o componente de estar operativo y accesible para su uso cuando se requiere.

- Tolerancia a fallos. Capacidad del sistema o componente para operar según lo previsto en presencia de fallos hardware o software.

- Capacidad de recuperación. Capacidad del producto software para recuperar los datos directamente afectados y restablecer el estado deseado del sistema en caso de interrupción o fallo.

2.4.6. Seguridad

Capacidad de protección de la información y los datos de manera que personas o sistemas no autorizados no puedan leerlos o modificarlos. Esta característica se subdivide a su vez en las siguientes sub-características:

- Confidencialidad. Capacidad de protección contra el acceso de datos e información no autorizados, ya sea accidentales o deliberadas.

- Integridad. Capacidad del sistema o componente para prevenir accesos o modificaciones no autorizados a datos o programas.

- No repudio. Capacidad de demostrar las acciones o eventos que han tenido lugar, de manera que dichas acciones o eventos no puedan ser repudiados posteriormente.

- Responsabilidad. Capacidad de rastrear de forma inequívoca las acciones de una entidad.

- Autenticidad. Capacidad de demostrar la identidad de un sujeto o un recurso.

2.4.7. Mantenibilidad

Esta característica representa la capacidad del producto software para ser modificado efectiva y eficientemente, debido a necesidades evolutivas, correctivas o perfectivas. Esta característica se subdivide a su vez en las siguientes sub-características:

- Modularidad. Capacidad de un sistema o programa de ordenador (compuesto de componentes discretos) que permite que un cambio en un componente tenga un impacto mínimo en los demás.

- Reusabilidad. Capacidad de un activo que permite que sea utilizado en más de un sistema software o en la construcción de otros activos.

- Analizabilidad. Facilidad con la que se puede evaluar el impacto de un determinado cambio sobre el resto del software, diagnosticar las deficiencias o causas de fallos en el software, o identificar las partes a modificar.

- Capacidad para ser modificado. Capacidad del producto que permite que sea modificado de forma efectiva y eficiente sin introducir defectos o degradar el desempeño.

- Capacidad para ser probado. Facilidad con la que se pueden establecer criterios de prueba para un sistema o componente y con la que se pueden llevar a cabo las pruebas para determinar si se cumplen dichos criterios.

2.4.8. Portabilidad

Capacidad del producto o componente de ser transferido de forma efectiva y eficiente de un entorno hardware, software, operacional o de utilización a otro. Esta característica se subdivide a su vez en las siguientes sub-características:

- Adaptabilidad. Capacidad del producto que le permite ser adaptado de forma efectiva y eficiente a diferentes entornos determinados de hardware, software, operacionales o de uso.

- Capacidad para ser instalado. Facilidad con la que el producto se puede instalar y/o desinstalar de forma exitosa en un determinado entorno.

- Capacidad para ser reemplazado. Capacidad del producto para ser utilizado en lugar de otro producto software determinado con el mismo propósito y en el mismo entorno.

Capítulo 3.
Estado del Arte

A continuación se muestra la revisión bibliográfica realizada con respecto a metodologías para medir calidad en implementaciones de IoT. Primero empezamos mostrando que significa tener una implementación de calidad utilizando IoT. Para eso buscamos una lista de artículos que explicaran en diferentes ámbitos cuales son los retos y problemas que presentan, y cuales son las características de calidad que hay que tener en cuenta para sobrellevar estos. Luego se muestra un breve resumen de como se evalúa calidad de software actualmente. Por último, se muestran investigaciones que muestran como se ha medido calidad hasta el momento en soluciones IoT.

3.1. Calidad en IoT

El principal objetivo de esta revisión es poder identificar cuales características de calidad se ven mas afectadas con los retor y problemas propios del ámbito del IoT. De esta forma poder llegar a una conclusión de cuales son las características y sub-características que pueden estar presentes en un modelo de calidad para implementaciones IoT.

En [48] discuten como la intranet de las cosas deben evolucionar a aplicaciones IoT y cuales son los retos en realizarlo. El enfoque principal en las características de calidad son la compatibilidad, el rendimiento, la confianza, la seguridad y la portabilidad. Los retos con la portabilidad y el rendimiento es debido a la heterogeneidad de los datos y la comunicación, la capacidad de comunicación de las cosas, el poder computacional y almacenamiento en cada a parte del sistema, la disponibilidad de energía, flexibilidad en el manejo de diferentes tecnologías, la mobilidad e interoperabilidad. Los retos en la seguridad son el la autenticación de las cosas y la confidencialidad de los datos. Por ultimo, tiene en cuenta la portabilidad con la adaptabilidad y confianza como un todo.

En [30], los autores hablan de las tendencias en las arquitecturas del IoT y futuras aplicaciones. Ellos enfocan los retos presentes para Iot en la seguridad. Las sub-características donde ellos muestran mayor enfoque es en la privacidad, confidencialidad e integridad. Ademas, también mencionan a la interoperabilidad como reto principal.

[11] es una revisión de las aplicaciones actuales y futuras de IoT en smart-cities. En ellas revisan los aspectos principales de las arquitecturas y los requerimientos en este tipo de aplicaciones. En lo concerniente a las características de calidad, se habla mucho de interoperabilidad, comportamiento en el tiempo, trazabilidad, mobiliad y disponibilidad. También hablan de privacidad, flexibilidad, modulariad y extensibilidad a menor escala.

[12] es otra revisión del estado actual del IoT en industrias. Se muestran como mayores retos la estandarización y la seguridad. También hablan de escalabilidad de las aplicaciones y heterogeneidad en los sistemas.

En [16], se propone una interface de dispositivos, sensible y reconfigurable para redes de sensores en el dominio del IoT. El sistema propuesto se enfoca en el rendimiento, haciéndolo adecuado para aplicaciones con requerimientos de tiempo real y efectividad, para la toma de datos en alta velocidad en el dominio de IoT. Ademas del rendimeinto, que es la principal caracteristica de calidad, tambien se enfocan en interoperabilidad y en tolerancia a fallos como una sub-característica de confianza. Finalmente, le dedican algo de atención a la operabilidad del sistema, realizando simulaciones de circuitos utilizando una interfaz gráfica de usuario.

En [35], los autores presentan una revisión de técnicas y metodologías usadas para implementaciones IoT. En el muestran que aplicaciones se enfocan en una lista de 17 parámetros de aseguramiento de calidad. Los autores revisaron 20 artículos distintos y en una tabla mostraron que articulo cumplían con cada requerimientos de calidad. Ellos concluyen que varios artículos se enfocan en los atributos de rendimiento, como también en reusabilidad y portailidad. Las características de mantenibilidad y extensibilidad aparecían en la mayoría de los artículos y que la eficiencia y confianza aparece en algunos artículos.

En [27], realizan un resumen con el estado actual y posibles aplicaciones y arquitecturas de IoT en la salud. Lo que mencionan es que debido a que la información en

la salud es privada puede ser objeto de ataques, por lo que la seguridad es un aspecto clave para este tipo de aplicaciones. Las características de seguridad que mencionan son la confidencialidad, integridad, autenticación, autorización y no repudio. Otras características que ellos relaciona con la seguridad son la disponibilidad, no tener datos ambiguos (madurez de la aplicación), seguridad externa, tolerancia a fallos, capacidad de recuperación. Además, por lo retos que esta tiene que superar se agregan a la lista la movilidad, escalabilidad, capacidad y utilización de recursos.

En [3], hablan acerca del alcance potencial que puede tener el IoT, las tecnologías más utilizadas y algunas aplicaciones importantes en el dominio del IoT. También discuten acerca de los problemas de seguridad y privacidad con identificación por radiofrecuencia (RFID, por sus siglas en Inglés Radio-Frecuency Identification) y redes de sensores. Ellos hablan acerca de la importancia de la estandarización, privacidad, identificación, autenticación y autorización. Otro aspecto de calidad del que hablan es de la compatibilidad entre el mundo físico y virtual. Con esto se acarrean problemas de eficiencia como la velocidad de la comunicación, capacidad, heterogeneidad entre dispositivos y seguridad en general. Hablan de tener mecanismos para evitar la ambigüedad y regularización. También mencionan la disponibilidad

En [40], hablan de los diferentes protocolos de conexión en todas las diferentes capas de comunicación en implementaciones de IoT. En el artículo separan la contextualización, problemas y retos por cada una de las capas. En la capa física hablan de los protocolos de comunicación IEEE 802.15.4 y 6LoWPAN. Los retos están en

trabajar en las limitaciones de capacidad de los canales de comunicación, ahorro de energía y heterogeneidad en el trafico de red. En la capa de red hablan de la estandarización del protocolo IPv6, los retos vienen dados por la interoperabilidad entre aplicaciones distintas. Por último, en la capa de aplicación explican como funciona el protocolo de aplicaciones restringidas (CoAP, por sus siglas en Inglés Constrained Application Protocol). En los retos y oportunidades explican las limitaciones de coAP con respecto a escalabilidad, madurez de las redes, costos y eficiencia. Ademas, agregan en sus recomendaciones de investigación profundizar en la coexistencia entre los distintos estándares entre otras sugerencias.

En [22], muestran la arquitectura de un sistema de monitoreo y control de salud remoto. En ella muestran sus componentes y mencionan ademas cuales atributos de calidad deben garantizarse dentro de este esquema. Ellos hablan de que al introducir IoT en los sistemas de salud estos aumentaran en inteligencia, flexibilidad e interoperabilidad. Los requerimientos de calidad expuestos por ellos son confidencialidad, integridad, autenticidad, autorización, comportamiento temporal, capaidad, completitud funcional, accessibilidad, madurez, disponibilidad y tocan la escalabilidad.

En [18] muestran un sistema de control de tráfico utilizando estándares de Maquina a Maquina (M2M, por sus siglas en ingles Machine to Machine) y sistemas de administración para restringir el paso de los carros por ciertas vías y así poder reducir la congestión vehícular en la ciudad de Bologna. Su foco principal con respecto a calidad es en autenticación, pertinencia funcional, utilización de recursos, tolerancia

a fallos y capacidad de recuperación. Además, mencionan la interoperabilidad dado que trabajan con distintos sistemas.

En [8] describen los problemas y retos al integrar Bigdata con el concepto de IoT a trave de un proyecto conocido como SMARTCAMPUS. Ademas de describirlos muestran como pueden ser estos superados. Los requerimientos arquitectónicos que ellos presentan son la coexistencia debido a la heterogeneidad de los sensores y su comunicación, la portabilidad con los requerimientos de adaptabilidad y facilidad de instalación, escalabilidad y accesibilidad. Las características de calidad que aparecen en la interacción de varios sensores son la consistencia de los datos, transparencia y facilidad de configuración. Las características de calidad en la velocidad y volumen son la escalabilidad. En la parte de agregar valor las características que aparecen son la escalabilidad. Además, hablan de que el sistema debe responder a características de calidad de comportamiento temporal.

Internet of things beyond the hype: research innovation and deployment [43] describe los aspectos generales del IoT. Ahí explican cuales son los diferentes escenarios donde el IoT puede potencialmente tener gran impacto, si supera unos macro retos con respecto a su infraestructura y utilización. Estos macro retos pueden ser emparejados con un conjunto de características de calidad. El primer macro reto sera lidiar con el hecho de tener billones de dispositivos conectados. Con este reto se ven obligados a realizar los registros y descubrimientos de dispositivos de una manera escalable, ademas lidiar con interoperabilidad entre silos heterogéneos de sensores y

mejorar en la virtualización de sensores y optimización de las comunicaciones M2M para así reducir el consumo de energía. El segundo reto es como IoT maneja la robustez y confianza del software. Las características de calidad que mencionan con este reto son seguridad, tolerancia a fallos y capacidad de recuperación, confianza y coexistencia y rendimiento funcional y por ultimo adaptabilidad. El ultimo macro reto es razonamiento inteligente de los datos del IoT, en este no se apunta a ningún atributo de calidad del software sino mas como aprovechar los datos censados para la toma de decisiones.

Las aplicaciones aquí revisadas nos brindan una visión acerca de que características toman relevancia en la calidad de software para IoT. También, nos muestra que dependiendo del campo de acción unas características toman mas relevancia que otras.

3.2. Modelos y normas para medir calidad de software

Podemos encontrar muchas investigaciones acerca de calidad de software que dan como resultado un modelo de calidad. Sin embargo, no todos los modelos de calidad que encontramos pueden ser aplicados en el contexto de IoT. En esta sección revisaremos un conjunto de modelos generales y para el contexto de IoT para la medición de calidad de software.

Uno de los modelos mas conocidos es el presentado en el estandard de calidad ISO/IEC 25000. El modelo se esta representado en la norma ISO/IEC 25010 y es

el sucesor del modelo ISO9126. El modelo ISO/IEC25010 incluye ademas de la descripción de las características, las métricas y una guía de aplicación [38]. Existen varias aplicaciones que utilizan el modelo para medir calidad. Por ejemplo, en IPA-VEMNET, es una aplicación para ciudades inteligentes. Estas utilizan el modelo para certificación de la calidad de los servicios web que el pavimento (la cosa) utiliza para comunicarse con el sistema central [36]. Como es visto, ellos no utilizan el modelo para verificar calidad en toda la aplicación, solo en una parte de esta.

Otro modelo utilizado para verificar calidad de aplicaciones IoT es el ATAM [29]. El modelo ATAM da como resultado un conjunto de datos que dan indicios de donde se encuentra los riesgos que pueden pasar a ser desiciones arquitectonicas. Existen soluiones IoT que utilizan el modelo ATAM, como se puede observar en [47] y [4].

Los modelos mencionados anterior mente son de propósito general, pero estos modelos no encajan para verificar la calidad de una solución de IoT. Existen varias investigaciones acerca que calidad de servicio (QoS, por sus siglas en Inglés) en aplicaciones IoT, pero la mayoria se enfocan en el protocolo de red RFID, middleware, fiabilidad de los dispositivos, seguridad y costos.

En [39], los autores presentan un modelo de evaluación de QoS, que ayuda escoger los servicios a utilizar cuando un evento ocurre. A pesar de que puede ayudar a la medición de la calidad del software, su naturaleza no es para esto y ademas no alcanza a ver la aplicación como un todo.

El método utilizado en [26], aplica el modelo ***goal, question, metric*** (GQM)

para obtener un modelo para medición calidad. El modelo es valido, pero el resultado muestra un modelo subjetivo por la formulación de las preguntas.

En [32], se presenta un modelo para medición de QoS. Su modelo está basado en la infraestructura y retos del IoT. Ellos identificaron un grupo de factores que afectan la calidad. Estos factores son la fiabilidad, interoperabilidad, desempeño, escalabilidad y seguridad. Como resultado muestran un modelo jerárquico, donde cada factor tiene un grupo de características de software ligadas a este. El modelo deja por fuera características de software importantes para el aseguramiento de calidad en IoT como la capacidad y la utilización de memoria. Ademas, no presentar la forma de medición ni como aplicarlas.

El modelo presentado en [31] esta basado en características no convencionales de software, que a su vez ganan relevancia en IoT. La primera característica es la participación de diferentes dispositivos. El autor explica que una aplicación IoT consiste en componentes de software y hardware que deben ser tomados en consideración al momento del diseño e implementación. La siguiente característica es el modelo de colaboración con los dispositivos IoT, el autor explica que los dispositivos hardware deben ser parte del modelo de negocios de una implementacion de este tipo. Otras características son la movilidad y conectividad, monitoreo remoto de dispositivos, y la limitación de recursos en algunos componentes hardware de la solución. El autor utiliza estas características especiales para definir el modelo colocando como principales características de calidad la funcionalidad, la eficiencia en el desempeño, fiabilidad y

portabilidad. El modelo encaja con las características especiales presentadas, sin embargo, otras características de software que siguen teniendo importancia son dejadas por fuera, por lo que el modelo no es suficiente para verificar la calidad de la aplicación como un todo. En adición a esto, el modelo permite dar diferentes pesos a cada característica para calcular el valor general de calidad, permitiendo solo tres niveles, por lo que se le da tres valores distintos. Estos tres pesos distintos son insuficientes en ciertos contextos o se convierten innecesarios si se considera con el mismo peso cada característica de calidad.

Los modelos específicos para Iot no toman en consideración todas las características de calidad importantes en IoT. Por lo que se hace necesario proponer un modelo que incluya todas estas. Ademas, el modelo debe ser sensible al contexto ya que no todas las características tienen la misma importancia en aplicaciones de distintas naturalezas. Debido a como se presentan las características y sub-características, nos basamos en el modelo jerárquico presente en el modelo ISO/IEC 25010.

Capítulo 4.
Metodología para la Realización de la Investigación

La metodología a realizar debe ser capaz de cuantificar los atributos de calidad y además colocar un límite mínimo de selección que garantice calidad. Además, se tienen que definir las métricas y que información se necesita de una aplicación para poder obtener los resultados. Una vez definidas las métricas, se procede a realizar el algoritmo para encontrar las mediciones.

4.1. Selección de la metodología

Basándonos en la metodología de investigación en diseño de ciencias (DSRM, por sus siglas en inglés *desing science research methodology*) propuesto en [37], vamos a diseñar la metodología de medición de calidad de software para el contexto del Internet de las Cosas.

El método DSRM divide el proceso en los pasos descritos en la Figura 4.1. El primer paso es la identificación del problema, en este se definen las causas y la descripción d este, ademas la importancia que este tiene. En el segundo se definen el objetivo general y el grupo de objetivos específicos para obtener una solución al problema. El tercer paso es el diseño y desarrollo de la solución el cual debe dar como

resultado un artefacto que cumpla con los objetivos o que se utilice para cumplir estos. El cuarto paso es la demostración, donde se le muestra a el cliente como funciona el artefacto. El quinto paso es la evaluación, que debe decirnos si el diseño y desarrollo estuvieron correcto, de no ser así, se tiene que volver a este paso o inclusive a re-definir los objetivos. El ultimo paso seria la comunicación, que es dar a conocer el resultado obtenido mediante publicaciones académicas o profesionales.

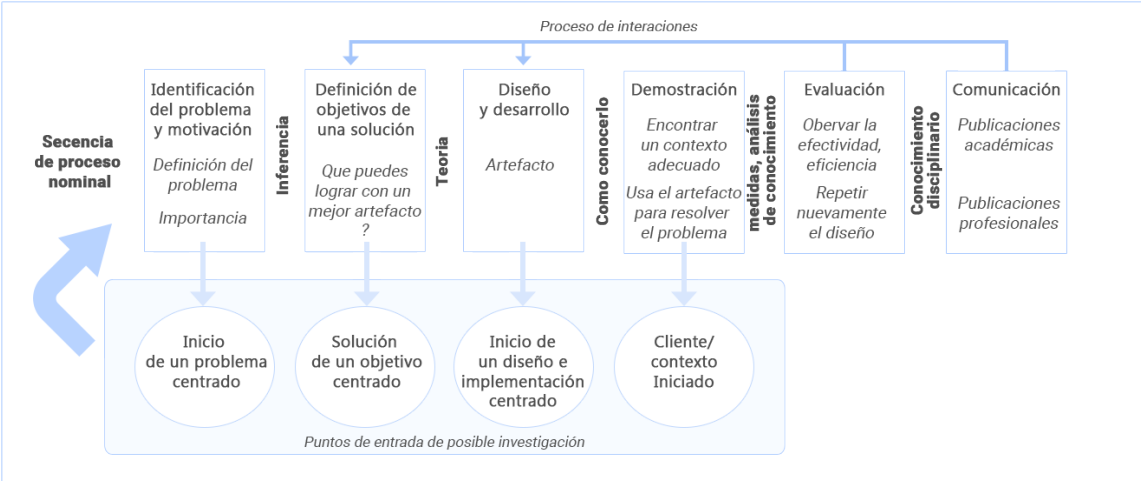

Figura 4.1: Modelo de proceso de DSRM [37].

4.2. Descripción de la metodología

En la primera parte de la investigación se realizará un análisis de la arquitectura de software y de las especificaciones de diseño software de aplicaciones del Internet de las cosas. Antes de empezar el análisis se deben buscar y seleccionar una lista de aplicaciones de Internet de las cosas y realizar dos grupos, un grupo de será utilizado para establecer la metodología de medición a utilizar y definir los criterios mínimos de selección y el segundo grupo será utilizado para pruebas. El objetivo del análisis

es buscar los atributos de calidad que le dieron mayor importancia en el momento de seleccionar la arquitectura de software, revisar si estos cumplen el propósito, para obtener el grupo de atributos de calidad tenidos en cuenta para aplicaciones de este tipo. Así mismo, listar cuales decisiones fueron tomadas por ser un problema de Internet de las Cosas o por el problema particular. Luego de obtener los atributos se prosigue a dar un rango de forma cuantitativa a cada atributo de calidad, a partir del estudio anterior, para tenerlos como base en el momento de seleccionar los criterios mínimos de selección.

En la segunda parte se realiza un cuadro comparativo entre los modelos o metodologías de medición de atributos de calidad que existen en la literatura. En este cuadro comparativo se debe tener en cuenta cuales atributos se miden, como se miden y que información se debe tener acerca del proyecto, para poder seleccionar una o varias metodologías que sean lo más simple posible y que califiquen la mayoría de las características de calidad que se deben tener en cuenta en el ámbito del Internet de las Cosas.

La tercera parte será, basándose en la información obtenida anteriormente, obtener el procedimiento de revisión y definir las métricas y que información se necesita de una aplicación para poder obtener los resultados. Ademas de esto se requiere obtener los pesos de cada característica de calidad en los diferentes contextos del IoT. Para esto se realizara un panel de expertos, con la ayuda del centro de excelencia y apropiación de Internet de las Cosas(CEA-IoT), el cual cuenta con el recurso humano

calificado para responder por el peso de las características de calidad en las distintas áreas.

Para la demostración, cuarta parte, se realizará una aplicación que reciba como entrada la información necesaria y obtendrá como resultado las mediciones de los atributos y una conclusión a partir de la calidad de la aplicación que se esté revisando.

Para la evaluación, quinta parte, se calificará al grupo de prueba seleccionado anteriormente aplicando la metodología obtenida. Es importante que este grupo las aplicaciones de Internet de las cosas se pueda decir, con base a la experiencia, si es de buena calidad o de mala calidad. Para así obtener cual fue el porcentaje de error que presenta la aplicación de la metodología propuesta.

Capítulo 5.
Modelo de Calidad Propuesto

Nuestro modelo esta basado en el modelo de calidad de la norma ISO/IEC 25000, mencionada anteriormente. Para encontrar los atributos de calidad que encajan en implementaicones de IoT. Revisamos 15 artículos para corroborar que características y sub-características de software están presentes en implementaciones de este tipo, para poder realizar nuestro modelo jerárquico.

En la Sección 5.1, listamos el estado actual de cada característica y sub-característica de software según la revisión realizada. En la Sección 5.2, se muestra el modelo jerárquico obtenido.

5.1. Identificación de los atributos de calidad

El modelo de calidad ISO/IEC 25000, no encaja en el dominio de IoT. Esto debido a que la naturaleza de las aplicaciones de IoT difieren con los softwares convencionales. Las razones principales son porque el hardware en las cosas y gateways son limitados, la heterogeneidad en las comunicaciones. Sin embargo, nuestra propuesta se basa en el modelo jerárquico que ISO/IEC 25000 presenta.

Basados en la revisión del estado del arte realizada, se tabulan a continuación

cada característica y sub-característica y las referencias donde son mencionados.

En la Tabla 5.1 aparecen las características de calidad y los artículos donde son mencionadas. Podemos observar que la usabilidad y la idoneidad funcional son pocos mencionados. Estas dos características no son tenidas en cuenta para la calidad en IoT debido a que el cliente de estos software no solamente es un humano sino que ademas son las cosas, que, su forma de utilizar el sistema es evaluado en la portabilidad. Ademas, la idoneidad funcional pierde criterio debido a que en IoT se preocupa mas porque el sistema sea confiable y mantenible.

Tabla 5.1: Características de calidad y los artículos donde son mencionadas

Característica	**Aparece**	**Característica**	**Aparece**
Idoneidad funcional	[12], [22], [18]	Fiabilidad	[48], [30], [11], [16], [27], [3], [40], [22], [18], [8], [43]
Eficiencia del rendimiento	[48], [11], [12], [16], [35], [27], [3], [40], [40], [22], [18], [8], [43]	Seguridad	[48], [30], [11], [12], [27], [3], [22], [18], [43]
Compatibilidad	[48], [30], [11], [12], [16], [3], [40], [22], [18], [8], [43]	Mantenibilidad	[11], [12], [35], [27], [40], [22], [8], [43]
Usabilidad	[16], [22], [8]	Portabilidad	[48], [11], [12], [35], [8], [43]

En la Tabla 5.2, se corrobora lo mencionado anteriormente, viendo pocas menciones en las sub-características para idoneidad funcional.

Tabla 5.2: Sub-características de idoneidad funcional y los artículos donde son mencionados

Sub-característica	**Aparece**
Completud funcional	[12], [22]
Exactitud funcional	[18]
Adecuación funcional	

En la Tabla 5.3, se observa que las características de eficiencia del rendimiento son mencionados en la mayoría de los artículos revisados. Dejando ver que esta es una característica importante.

Tabla 5.3: Sub-características de Eficiencia del rendimiento y los artículos donde son mencionados

Sub-característica	**Aparece**
Comportamiento en el tiempo	[11], [12], [16], [35], [3], [22], [8]
Utilización de recursos	[48], [35], [27], [18]
Capacidad	[35], [27], [3], [40], [22]

En la Tabla 5.4 nos damos cuenta que la interoperabilidad es mencionada por muchos artículos como una sub-característica importante en la calidad del software en IoT. Incluso, en algunas investigaciones es mencionado como una característica principal. Debido a que nos basamos en el modelo ISO/IEC 25010 esta es tomada como sub-característica de compatibilidad.

Tabla 5.4: Sub-características de compatibilidad y los artículos donde son mencionados

Sub-característica	**Aparece**
Co-existencia	[11], [27], [40], [8], [43]
Interoperabilidad	[48], [30], [11], [12], [16], [3], [40], [22], [18], [43]

En la Tabla 5.5 podemos observar que la usabiliad tiene sub-características que no son mencionadas en ningún artículo revisado. Asimismo, las sub-características mencionadas tienen relación con otras sub-características.

Tabla 5.5: Sub-características de usabilidad y los artículos donde son mencionados

Sub-característica	**Aparece**
Capacidad para reconocer su adecuación	
Capacidad de aprendizaje	
Operabilidad	[16]
Proteccion contra errores de usuario	
Accesibilidad	[22], [8]

En la Tabla 5.6 podemos observar que sus sub-características tienen algunas menciones. A pesar de no ser tantas menciones como otras sub-caracteristicas, tienen una importancia considerable.

Tabla 5.6: Sub-características de fiabilidad y los artículos donde son mencionados

Sub-característica	**Aparece**
Madurez	[27], [3], [40], [22], [8], [43]
Disponibilidad	[11], [27], [3], [22]
Tolerancia a fallos	[16], [27], [18], [43]
Capacidad de recuperación	[27], [18], [43]
Conectividad	[30], [12], [8], [43]

En la Tabla 5.7 podemos observar que la sub-característica de confidencialidad y autenticidad juegan un rol importante en la característica de seguridad. Aun así, todas las sub-características tienen importancia.

Tabla 5.7: Sub-características de seguridad y los artículos donde son mencionados

Sub-característica	**Aparece**
Confidencialidad	[48], [30], [11], [12], [27], [3], [22]
Integridad	[30], [27], [22]
No repudio	[30], [27], [22]
Responsabilidad	[30], [27], [3], [22]
Authenticidad	[48], [30], [27], [3], [22], [18]

En la Tabla 5.8 podemos observar que la sub-característica mas importante es la escalabilidad, esto puede ser debido a que en este tipo de aplicaciones la cantidad de recursos necesarios puede variar debido a la cantidad de dispositivos que pueden ser conectados. También tiene importancia la reusabiliad y la modularidad.

Tabla 5.8: Sub-características de mantenibilidad y los artículos donde son mencionados

Sub-característica	**Aparece**
Modularidad	[11], [12], [22]
Reusabilidad	[35], [11], [12], [22]
Analizabilidad	[11]
Capacidad para ser modificado	[11]
Capacidad para ser probado	
Escalabilidad	[11], [12], [35], [27], [40], [22], [8], [43]

En la Tabla 5.9 podemos observar que la sub-característica con mas menciones es la adaptabilidad. Una de las razones de esto es por la heterogeneidad de lo dispositivos y los canales de comunicación. A pesar que la conformidad no tiene muchas menciones, debido a la cantidad de estándares de comunicación es importante tenerla

en cuenta en el modelo de calidad.

Tabla 5.9: Sub-características de portabilidad y los artículos donde son mencionados

Sub-característica	**Aparece**
Adaptabilidad	[48], [11], [8], [43]
Capacidad para ser instalado	[12], [8]
Capacidad para ser reemplazado	
Conformidad	[12], [40]

5.2. Modelo jerárquico de calidad

Las características de calidad utilizadas en nuestro modelo están basadas en los retos y amenazas para IoT y los factores de calidad analizados en trabajos previos. Aunque IoT tiene diferentes factores que pueden afectar su calidad, solo los relacionados con el software son considerados. Cada característica tiene un conjunto de sub-características. En la Figura 5.1 se encuentra el modelo de calidad de software.

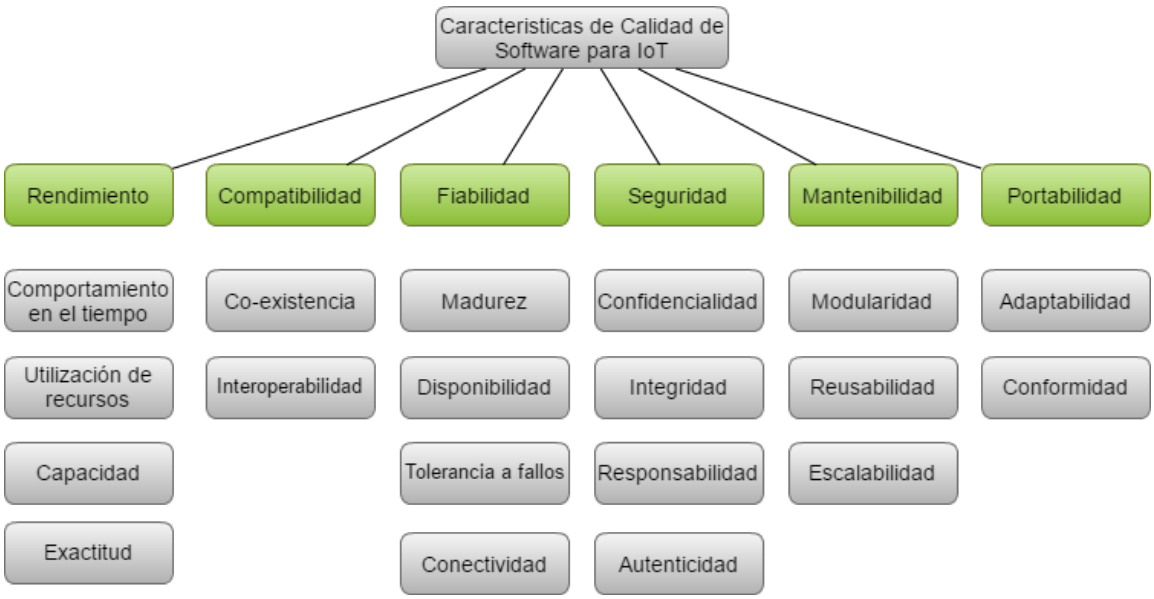

Figura 5.1: Modelo jerárquico de calidad propuesto

5.3. Métricas para las características de calidad del modelo

La medición de la calidad de proyectos o softwares es muy importante porque sin medición de la calidad no se pueden obtener las metas propuestas. Midiendo, se puede controlar y mejorar la calidad de nuestros proyectos. Ademas, tener una buena herramienta de medición es importante porque solo se tienen que encontrar cuales son los factores que afectan negativamente la calidad [9].

El modelo propuesto tiene un conjunto de ecuaciones para calcular la calidad individualmente de cada sub-característica. El modelo tiene una ecuación que permite, a partir del resultado de cada sub-característica, obtener un valor de calidad para cada característica. Por último, se tiene una ecuación para calcular un valor general de calidad, a partir del resultado de cada característica. La forma general está descrita en la Ecuación 5.1.

$$Q = \frac{\alpha_1 Q_1 + \alpha_2 Q_2 + \alpha_3 Q_3 + \alpha_4 Q_4 + \alpha_5 Q_5 + \alpha_6 Q_6}{\alpha_1 + \alpha_2 + a_3 + \alpha_4 + a_5 + \alpha_6}. \tag{5.1}$$

Donde Q_1 el el valor de calidad de eficiencia de rendimiento, Q_2 es el valor de calidad de compatibilidad, Q_3 es el valor de calidad de fiabilidad, Q_4 es el valor de calidad de seguridad, Q_5 es el valor de calidad de mantenibilidad, Q_6 es el valor de calidad de portabilidad, y cada α_i es el peso en la calidad general que se le da a cada característica dependiendo del contexto de la aplicación .

En las siguientes sub-secciones se definen las métricas para cada características de

calidad, divididas entre cada sub-característica y la forma como son implementadas.

Métrica para eficiencia de rendimiento

La métrica para para eficiencia de rendimiento están dadas por la Ecuación 5.2.

$$Q_1 = \frac{\alpha_{11}SQ_{11} + \alpha_{12}SQ_{12} + \alpha_{13}SQ_{13} + \alpha_{14}SQ_{14}}{\alpha_{11} + \alpha_{12} + \alpha_{13} + \alpha_{14}}. \tag{5.2}$$

Donde SQ_{11} representa el resultado para comportamiento en el tiempo, SQ_{12} el resultado de utilización de recursos, SQ_{13} el resultado para capacidad, y SQ_{14} el resultado para exactitud.

La métrica para comportamiento en el tiempo esta basado en la respuesta de tiempos de la aplicación. La métrica para comportamiento en el tiempo (SQ_{11}) es dado por la Ecuación 5.3.

$$SQ_{11} = \frac{M_{11}}{N_{11}}. \tag{5.3}$$

Donde N_{11} es el número de módulos que deben satisfacer los requerimientos de respuesta en tiempos específicos, y M_{11} es el número de módulos que dan respuesta promedio en el tiempo esperado.

La métrica para utilización de recursos (SQ_{12}) es dada por la Ecuación 5.4.

$$SQ_{12} = \frac{\sum_{i=1}^{n} \frac{ru_1 + ru_2 + ru_3}{3}}{N_{12}} \tag{5.4}$$

Donde N_{12} es el numero de módulos en la aplicación. ru_1 es el porcentaje de

utilización de memoria, ru_2 es el porcentaje de utilización de procesamiento, ru_3 es el porcentaje de utilización de recursos de red.

La métrica para capacidad (SQ_{13}) esta dada por la Ecuación 5.5.

$$SQ_{13} = \frac{M_{13}}{N_{13}}. \tag{5.5}$$

Donde N_{13} es el número de módulos del software y M_{13} es el número de módulos que satisfacen los requerimientos de capacidad (los parámetros siempre mantengan por dentro de los limites).

La métrica para exactitud (SQ_{14}) esta dada por la Ecuación 5.6.

$$SQ_{14} = \frac{M_{14}}{N_{14}}. \tag{5.6}$$

Donde N_{14} es el número de funciones del software y M_{14} es el número de funciones que satisfacen los requerimientos de exactitud (Siempre de el mismo resultado para el mismo set de datos de entrada).

Métrica para compatibilidad

La métrica para para compatibilidad están dadas por la Ecuación 5.7.

$$Q_2 = \frac{\alpha_{21} SQ_{21} + \alpha_{22} SQ_{22}}{\alpha_{21} + \alpha_{22}}. \tag{5.7}$$

Donde SQ_{21} representa el resultado de co-existencia, y SQ_{22} el resultado de interoperabilidad.

La métrica para co-existencia (SQ_{21}) esta dada por la Ecuación 5.8.

$$SQ_{21} = \frac{\sum_{i=1}^{n} \frac{N_i}{M_i}}{n} \tag{5.8}$$

Donde n es el numero de recursos compartidos, M_i es el numero de módulos que utilizan el recurso i, y N_i son los módulos que tienen acceso al recurso en el tiempo que sea necesario.

La métrica para interoperabilidad (SQ_{22}) esta dada por la Ecuación 5.9.

$$SQ_{22} = \frac{N_{22}}{M_{22}}. \tag{5.9}$$

Donde M_{22} es el número de canales de comunicación necesarios en el sistema, N_{22} es el número de canales de comunicación que satisfacen el requerimiento de interoperabilidad (que la información transmitida pueda ser interpretada por el receptor).

Métrica para fiabilidad

La métrica para fiabilidad están dadas por la Ecuación 5.10.

$$Q_3 = \frac{\alpha_{31} SQ_{31} + \alpha_{32} SQ_{32} + \alpha_{33} SQ_{33} + \alpha_{34} SQ_{34}}{\alpha_{31} + \alpha_{32} + a_{33} + \alpha_{34}}. \tag{5.10}$$

Donde SQ_{31} representa el resultado para madurez, SQ_{32} representa el resultado para disponibilidad, SQ_{33} representa el resultado para tolerancia a fallos, y SQ_{34} el resultado para conectividad.

La métrica para madurez (SQ_{31}) esta dada por la Ecuación 5.11.

$$SQ_{31} = \frac{N_{31}}{M_{31}}. \tag{5.11}$$

Donde M_{31} es el número de módulos en los que está dividido el software y N_{31} es el número de módulos que satisfacen el requerimiento de fiabilidad en circunstancias normales.

La métrica para disponibilidad (SQ_{32}) esta dada por la Ecuación 5.12.

$$SQ_{32} = \frac{N_{32}}{N_{32} + M_{32}}. \tag{5.12}$$

Donde N_{32} se refiere al tiempo promedio entre fallos y M_{32} es el tiempo promedio en el que son arreglados.

La métrica para tolerancia a fallos (SQ_{33}) esta dada por la Ecuación 5.13.

$$SQ_{33} = \frac{\sum_{i=1}^{n} D(i)}{n} \tag{5.13}$$

Donde n es el número de módulos, y $D(i)$ es el grado en el que el módulo i opera a pesar de fallos de hardware o software.

La métrica para conectividad (SQ_{34}) esta dada por la Ecuación 5.14.

$$SQ_{34} = \frac{\sum_{i=1}^{n} D(i)}{n} \tag{5.14}$$

Donde n es el número de interfaces a otros proyectos y servicios, y $D(n)$ es el grado en el cual el modulo i logra conectarse satisfactoriamente hacia el otro servicio o software.

Métrica para seguridad

La métrica para seguridad está dadas por la Ecuación 5.15.

$$Q_4 = \frac{\alpha_{41}SQ_{41} + \alpha_{42}SQ_{42} + \alpha_{43}SQ_{43} + \alpha_{44}SQ_{44}}{\alpha_{41} + \alpha_{42} + \alpha_{43} + \alpha_{44}}. \tag{5.15}$$

Donde SQ_{41} representa el resultado para confidencialidad, SQ_{42} el resultado para integridad, SQ_{43} el resultado para responsabilidad, y SQ_{44} el resultado para autenticidad.

La métrica para confidencialidad (SQ_{41}) esta dada por la Ecuación 5.16.

$$SQ_{41} = \frac{N_{41}}{M_{41}}. \tag{5.16}$$

Donde M_{41} es el número de módulos en el sistema y N_{41} es el número de módulos que satisfacen el requerimiento de confidencialidad (mecanismos que permitan que la información solo puede ser vista por personal autorizado).

La métrica para integridad (SQ_{42}) esta dada por la Ecuación 5.17.

$$SQ_{42} = \frac{N_{42}}{M_{42}}. \tag{5.17}$$

Donde M_{42} es el número de transacciones realizadas en el sistema y N_{42} es el número de transacciones que no se vieron modificadas durante su transmisión.

La métrica para responsabilidad (SQ_{43}) esta dada por la Ecuación 5.18.

$$SQ_{43} = \frac{N_{43}}{M_{43}}. \tag{5.18}$$

Donde M_{43} es el número de acciones tomadas en el sistema y N_{43} es el número de acciones que se puede verificar el autor.

La métrica para autenticidad (SQ_{44}) esta dada por la Ecuación 5.19.

$$SQ_{44} = \frac{N_{44}}{M_{44}}. \tag{5.19}$$

Donde M_{44} es el número de usuarios que intentan autenticarse en el sistema y N_{44} es el número de usuarios que no son ó falsos positivos (Usuario que ingreso con una identidad que no es la suya) ó falsos negativos (Usuario que intentó ingresar con credenciales del sistema, pero este lo rechazó).

Métrica para mantenibilidad

La métrica para mantenibilidad está dadas por la Ecuación 5.20.

$$Q_5 = \frac{\alpha_{51} SQ_{51} + \alpha_{52} SQ_{52} + \alpha_{53} SQ_{53}}{\alpha_{51} + \alpha_{52} + \alpha_{53}}. \tag{5.20}$$

Donde SQ_{51} representa el resultado para modularidad, SQ_{52} el resultado para reusabilidad, y SQ_{53} el resultado para escalabilidad.

La métrica para modularidad (SQ_{51}) esta dada por la Ecuación 5.21.

$$SQ_{51} = \frac{\frac{\sum_{j=1}^{M} \sum_{i=1}^{M} N(i,j)}{M}}{M}. \tag{5.21}$$

Donde M es el número de módulos y $N(i, j)$ es el grado en el cual el modulo i se e afectado si se elimina el modulo j.

La métrica para reusabilidad (SQ_{52}) esta dada por la Ecuación 5.22.

$$SQ_{52} = \frac{N_{52}}{M_{52}}. \tag{5.22}$$

Donde M_{52} es el número de módulos del sistema y N_{52} es el número de módulos que pueden ser utilizados fuera del sistema, sin la necesidad de otro modulo.

La métrica para escalabilidad (SQ_{53}) esta dada por la Ecuación 5.23.

$$SQ_{53} = \frac{N_{53}}{M_{53}}. \tag{5.23}$$

Donde M_{53} es el número de módulos del sistema y N_{53} es el número de módulos que el sistema permite agregarle nuevos recursos.

Métrica para portabilidad

La métrica para portabilidad está dadas por la Ecuación 5.24.

$$Q_6 = \frac{\alpha_{61} SQ_{61} + \alpha_{62} SQ_{62}}{\alpha_{61} + \alpha_{62}}. \tag{5.24}$$

Donde SQ_{61} representa el resultado para adaptabilidad, and SQ_{62} el resultado para conformidad.

La métrica para adaptabilidad (SQ_{61}) esta dada por la Ecuación 5.25.

$$SQ_{61} = \frac{N_{61}}{M_{61}}. \tag{5.25}$$

Donde M_{61} es el número de módulos en el sistema y N_{61} es el número de módulos que pueden ser adaptados de forma efectiva y eficiente en diferentes entornos.

La métrica para conformidad (SQ_{62}) esta dada por la Ecuación 5.26.

$$SQ_{62} = \frac{N_{62}}{M_{62}}. \tag{5.26}$$

Donde M_{62} es el número de módulos, servicios o sub-programas que deben seguir un estandard o aluna ley y N_{62} es el número de módulos que satisfacen el estandard o ley.

Capítulo 6.
Análisis de resultados

Para darles valores a los coeficientes α_i de la Ecuación 5.1 se realizó un panel de expertos con los integrantes del centro de excelencia y apropiación de IoT (CEA-Iot).

6.1. Obtención de valores para los coeficientes por línea de trabajo

A cada miembro del panel se les pidió que dieran un valor de 0 a 10, que según su concepto que nivel de importancia tenia cada característica y sub-característica en cada uno de sus líneas de trabajo del centro. Los resultados se utilizan en las ecuaciones con un valor de 0 a 1 por lo que se normalizan los valores dividiéndolos entre 10. Los resultados se presentaran por cada una de las lineas de trabajo en cada una de las siguientes secciones.

6.1.1. Salud

Para la linea de salud se obtuvieron los valores promedios para cada característica que se observan el la Tabla 6.1. para las sub-características se observan en la Tabla 6.2. En los resultados se puede observar que la seguridad y la fiabilidad son importantes en aplicaciones de salud. Esto puede ser debido a que la información de un paciente

debe ser protegida y ademas al confiar la salud a estos dispositivos, estos deben ser confiables.

Tabla 6.1: Promedios para cada característica en Salud

Característica	**Valor Promedio**
Eficiencia de rendimiento	8,0
Compatibilidad	7,0
Fiabilidad	8,8
Seguridad	8,9
Mantenibilidad	7,9
Portabilidad	6,9

Tabla 6.2: Promedios para cada sub-característica en Salud

Característica	**Valor**	**Característica**	**Valor**	**Característica**	**Valor**
Comportamineto en el tiempo	8,6	Disponibilidad	8,5	Modularidad	7,4
Utilizacion de recursos	7,4	Tolerancia a fallos	8,3	Reusabilidad	8
Capacidad	7,7	Conectividad	8,7	Escalabilidad	7,3
Exactitud	7,1	Confidencialidad	8,9	Adaptabilidad	6,9
Co-existencia	7,9	Integridad	8,7	Conformidad	6,9
Interoperabilidad	8,3	Autenticidad	8,4		
Madurez	7,3	Responsabilidad	8,3		

6.1.2. Logística

Para la linea de logística se obtuvieron los valores promedios para cada característica que se observan el la Tabla 6.3. para las sub-características se observan en la Tabla 6.4. En logística tienen mas relevancia la fiabilidad y el rendimiento, esto se debe a que para no romper la cadena estas aplicaciones deben responder a los tiempos precisos de manera confiable.

Tabla 6.3: Promedios para cada característica en Logística

Característica	**Valor Promedio**
Eficiencia de rendimiento	8,4
Compatibilidad	7,2
Fiabilidad	8,2
Seguridad	7,9
Mantenibilidad	7,9
Portabilidad	6,7

Tabla 6.4: Promedios para cada sub-característica en Logística

Característica	**Valor**	**Característica**	**Valor**	**Característica**	**Valor**
Comportamineto en el tiempo	8,3	Disponibilidad	8,2	Modularidad	7,3
Utilizacion de recursos	8,1	Tolerancia a fallos	8,2	Reusabilidad	7,8
Capacidad	8	Conectividad	8,7	Escalabilidad	8,3
Exactitud	8,6	Confidencialidad	8,0	Adaptabilidad	7,9
Co-existencia	8,0	Integridad	8,5	Conformidad	6,7
Interoperabilidad	8,2	Autenticidad	8,0		
Madurez	7,5	Responsabilidad	8,2		

6.1.3. Industria

Para la linea de Industria se obtuvieron los valores promedios para cada característica que se observan el la Tabla 6.5. para las sub-características se observan en la Tabla 6.6. En Industria debido al el hermetismo de las empresas, su información la quieren siempre correcta y solo para ellos, por esta razón la seguridad y fiabilidad tienen mayor peso.

Tabla 6.5: Promedios para cada característica en Industria

Característica	**Valor Promedio**
Eficiencia de rendimiento	8,6
Compatibilidad	8,1
Fiabilidad	9,0
Seguridad	9,1
Mantenibilidad	8,4
Portabilidad	7,6

Tabla 6.6: Promedios para cada sub-característica en Industria

Característica	**Valor**	**Característica**	**Valor**	**Característica**	**Valor**
Comportamineto en el tiempo	8,5	Disponibilidad	8,2	Modularidad	7,8
Utilizacion de recursos	8,0	Tolerancia a fallos	8,8	Reusabilidad	7,5
Capacidad	8,2	Conectividad	8,5	Escalabilidad	8,7
Exactitud	8,5	Confidencialidad	8,5	Adaptabilidad	8,0
Co-existencia	7,7	Integridad	8,3	Conformidad	6,9
Interoperabilidad	8,3	Autenticidad	7,8		
Madurez	7,8	Responsabilidad	7,7		

6.1.4. Vestibles

Para la linea de Vestibles se obtuvieron los valores promedios para cada característica que se observan el la Tabla 6.7. para las sub-características se observan en la Tabla 6.8. La portabilidad gana relevancia en vestibles dado que los dispositivos son de cambio constante, por lo que las aplicaciones realizadas deben funcionar en un número considerado de dispositivos.

Tabla 6.7: Promedios para cada característica en Vestibles

Característica	**Valor Promedio**
Eficiencia de rendimiento	7,9
Compatibilidad	8,0
Fiabilidad	8,1
Seguridad	7,6
Mantenibilidad	7,4
Portabilidad	8,6

Tabla 6.8: Promedios para cada sub-característica en Vestibles

Característica	**Valor**	**Característica**	**Valor**	**Característica**	**Valor**
Comportamineto en el tiempo	7,3	Disponibilidad	6,3	Modularidad	7,3
Utilizacion de recursos	7,7	Tolerancia a fallos	7	Reusabilidad	7,7
Capacidad	6,3	Conectividad	6,8	Escalabilidad	6,8
Exactitud	6,5	Confidencialidad	5,7	Adaptabilidad	7,0
Co-existencia	6,7	Integridad	6,0	Conformidad	6,3
Interoperabilidad	6,7	Autenticidad	5,6		
Madurez	6,2	Responsabilidad	6,3		

6.1.5. Seguridad

Para la linea de seguridad se obtuvieron los valores promedios para cada característica que se observan el la Tabla 6.9. para las sub-características se observan en la Tabla 6.10. En seguridad ya sea industrial o en general deben ser precisas y correctas por esta razón as características de eficiencia y fiabilidad tienen mayor relevancia. Ademas, para evitar en ciertos casos ataques terroristas o de ese estilo, la información debe estas segura.

Tabla 6.9: Promedios para cada característica en Seguridad

Característica	**Valor Promedio**
Eficiencia de rendimiento	8,9
Compatibilidad	8,3
Fiabilidad	9,0
Seguridad	9,0
Mantenibilidad	7,5
Portabilidad	7,1

Tabla 6.10: Promedios para cada sub-característica en Seguridad

Característica	**Valor**	**Característica**	**Valor**	**Característica**	**Valor**
Comportamineto en el tiempo	8,8	Disponibilidad	8,7	Modularidad	7,0
Utilizacion de recursos	8,2	Tolerancia a fallos	9	Reusabilidad	7,5
Capacidad	8,7	Conectividad	7,8	Escalabilidad	8,0
Exactitud	9	Confidencialidad	8,3	Adaptabilidad	8,0
Co-existencia	8,5	Integridad	8,2	Conformidad	6,7
Interoperabilidad	8,7	Autenticidad	8,3		
Madurez	8,7	Responsabilidad	8,2		

6.1.6. Agroindustria y Medio Ambiente

Para la linea de seguridad se obtuvieron los valores promedios para cada característica que se observan el la Tabla 6.11. para las sub-características se observan en la Tabla 6.12. En la agroindustria y medio ambiente por lo general el software es limitado, por lo que un uso eficiente de sus recursos toma importancia. A esto se le agrega que estas aplicaciones por su localidad deben tener un alto grado de madurez, tolerancia a fallos y disponibilidad.

Tabla 6.11: Promedios para cada característica en Agroindustria y Medio Ambiente

Característica	**Valor Promedio**
Eficiencia de rendimiento	8,6
Compatibilidad	8,0
Fiabilidad	7,9
Seguridad	7,0
Mantenibilidad	8,1
Portabilidad	7,9

Tabla 6.12: Promedios para cada sub-característica en Agroindustria y Medio Ambiente

Característica	**Valor**	**Característica**	**Valor**	**Característica**	**Valor**
Comportamineto en el tiempo	8,5	Disponibilidad	6,8	Modularidad	7,3
Utilizacion de recursos	8,3	Tolerancia a fallos	7,2	Reusabilidad	7,8
Capacidad	8,0	Conectividad	5,3	Escalabilidad	8,0
Exactitud	8,3	Confidencialidad	6,3	Adaptabilidad	6,9
Co-existencia	7	Integridad	7,3	Conformidad	7,0
Interoperabilidad	7	Autenticidad	6		
Madurez	6,3	Responsabilidad	6,5		

6.1.7. Gobierno

Para la linea de gobierno se obtuvieron los valores promedios para cada característica que se observan el la Tabla 6.13. para las sub-características se observan en la Tabla 6.14. En las aplicaciones de gobierno, donde son de uso publico es importante mantener la información segura.

Tabla 6.13: Promedios para cada característica en Gobierno

Característica	**Valor Promedio**
Eficiencia de rendimiento	8,3
Compatibilidad	8,1
Fiabilidad	8,9
Seguridad	9,3
Mantenibilidad	7,9
Portabilidad	7,1

Tabla 6.14: Promedios para cada sub-característica en Gobierno

Característica	**Valor**	**Característica**	**Valor**	**Característica**	**Valor**
Comportamineto en el tiempo	7,3	Disponibilidad	6,7	Modularidad	6,3
Utilizacion de recursos	6,5	Tolerancia a fallos	6,8	Reusabilidad	7,0
Capacidad	7,2	Conectividad	7,0	Escalabilidad	7,8
Exactitud	7,5	Confidencialidad	7,3	Adaptabilidad	6,9
Co-existencia	6,5	Integridad	7,2	Conformidad	6,7
Interoperabilidad	6,3	Autenticidad	7,2		
Madurez	6,7	Responsabilidad	6,5		

6.2. Algoritmo de aplicación de la metodología

Para aplicar las métricas se debe seguir un algoritmo, donde se recopila la información necesaria para poder aplicar la métrica y obtener los valores de calidad para cada característica y el valor general. El flujo general del algoritmo se puede observar en la Figura 6.1.

El primer paso es seleccionar la línea de trabajo en la que se enmarca la implementación. Esto nos dará la lista de coeficientes de importancias para cada característica y sub-característica de calidad para aplicar a la ecuación. Para un proyecto con características especiales o por requerimientos propios, los valores de los coeficientes

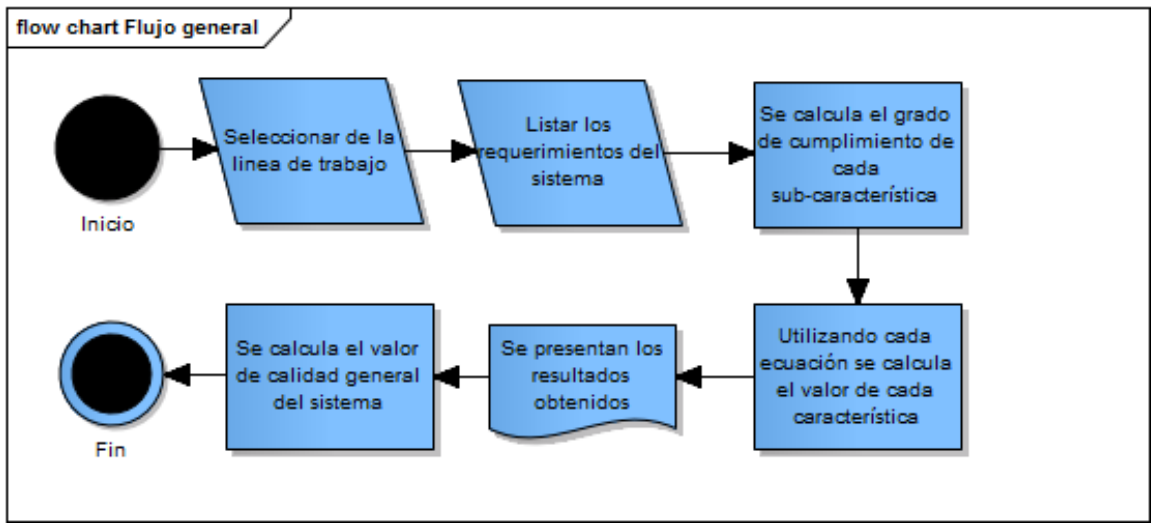

Figura 6.1: Flujo general del algoritmo para aplicar las métricas

pueden variar.

El siguiente paso es revisar con los interesados cuales son los requerimientos del software de la implementación. Estos deben estar lo más detallado posible, debido a que generalizar o agrupar requerimientos puede guiar a obtener valores de calidad que no estén acorde al diseño o a la arquitectura del software de la aplicación. Ademas de obtener los requerimientos, se obtienen para cada uno cuales son las sub-características de calidad relacionadas.

Para cada relación entre los requerimientos del software y las sub-características se calcula con que grado el requerimiento cumple con la sub-característica de calidad. Estos valores están limitados entre 0 y 1, siendo 0 que el requerimiento no cumple con la sub-característica y 1 que lo cumple al 100 % de satisfacción.

Luego se calcula utilizando las ecuaciones de la Sección 5.3 y los coeficientes de importancia para obtener los niveles de calidad en cada una de las características de

calidad del software y el valor de calidad general.

Finalmente se obtiene los valores de calidad para cada una de las características y el valor de calidad general. Se presentan para sacar conclusiones y tomar decisiones en cuanto a el diseño e implementación de código de la aplicación de software del proyecto IoT.

Para analizar los valores se tiene en cuenta que mientras mas se acerque a uno el valor general, esto implica que se tiene mejor calidad general o que las características de mayor peso tienen buena calidad. Un mínimo aceptable para que una aplicación sea de buena calidad es cuando su valor general es mayor a 0.7, para que sea de excelente calidad debe obtener un valor mayor a 0,9. Esto también afecta a las características con su conjunto de sub-característica.

6.3. Comparación y Validación de nuestro modelo

En esta sección se compara nuestra propuesta con los trabajos más similares al nuestro. El proceso consistió en evaluar el mismo sistema bajo los diferentes modelos. En la segunda parte, validamos nuestro modelo en un sistema real implementado en el CEA IoT para la optimización de la irrigación para cultivos de palma de aceite.

6.3.1. Comparación con otros modelos

La métrica fue aplicada a la aplicación presentada en [31]. El primer paso es enmarcar en una de las lineas de trabajo la aplicación. Dado el contexto de aplicación para vestibles los valores de los pesos α están dados en las Tablas 6.16 y 6.15. El

segundo paso es verificar la lista de requerimientos con los interesados. La lista de requerimientos se presenta en [31], con la explicación de si se cumple o no el requerimiento y por que razones. El tercer paso es calcular el grado de cumplimiento para cada sub-característica, el resultado de este paso puede ser visto en la Tabla 6.15. Luego, en el cuarto paso se utilizan las ecuaciones para calcular el valor a cada característica a partir del valor de sus sub-características. El resultado puede ser visto en la Tabla 6.16.

Tabla 6.15: pesos de los α para las sub-características para vestibles obtenidos en el panel de experto y resultados para el caso de estudio

Características	α	valor	**Características**	α	valor
Comportamiento temporal	α_{11} 0,73	0,75	Confidencialidad	α_{41} 0,57	1
Utilización de recursos	α_{12} 0,77	0,62	Integridad	α_{42} 0,6	1
Capacidad	α_{13} 0,87	0,62	Responsabilidad	α_{43} 0,63	1
Exactitud	α_{14} 0,65	0,87	Autenticidad	α_{44} 0,56	1
Co-existencia	α_{21} 0,67	1	Modularidad	α_{51} 0,73	1
Interoperabilidad	α_{22} 0,67	0	Reusabilidad	α_{52} 0,77	1
Madurez	α_{31} 0,62	0,75	Escalabilidad	α_{53} 0,68	0
Disponibilidad	α_{32} 0,63	1	Adaptabilidad	α_{61} 0,7	0,87
Tolerancia a fallos	α_{33} 0,72	0,75	Conformidad	α_{62} 0,63	0,87
Conectividad	α_{34} 0,68	0,75			

Tabla 6.16: pesos de los α para las características para vestibles obtenidos en el panel de experto y resultados para el caso de estudio

Características	α	valor
Q_1 Eficiencia de rendimiento	α_1 0,79	0,7
Q_2 Compatibilidad	α_2 0,80	0,5
Q_3 Fiabilidad	α_3 0,81	0,81
Q_4 Seguridad	α_4 0,76	0,76
Q_5 Mantenibilidad	α_5 0,74	0,69
Q_6 Portabilidad	α_6 0,86	0,87

El ultimo paso es utilizar la ecuación para calcular el valor general de calidad.Este fue de 0,72/1 = 0,72. El resultado general para la misma aplicación en [31] fue de 0,72/0,8 = 0,9. En su modelo, cada sub-característica tiene el mismo peso para calcular el valor de cada característica. Ademas, el autor le da el mismo peso a cada característica para calcular el valor general, lo que varia con el nuestro dado que según el contexto cada característica tiene un valor distinto. Otra diferencias es el valor que se obtiene en en la característica de eficiencia. El valor obtenidos por el autor con su métrica es de 0,975 sobre 1; con nuestro modelo el resultado es de 0,7. La razón principal para esta diferencia es que en su modelo no tienen en cuenta la capacidad ni la exactitud como características de eficiencia de rendimiento.

Debido a que en el modelo propuesto en [32] no se pudo realizar una comparación debido a que ellos no presentan las métricas, ni el proceso para ser aplicadas. En la Tabla 6.17 se muestran otras diferencias entre el modelo comparado, el modelo presentado en [32] y el nuestro.

Tabla 6.17: Comparación de nuestro modelo con otros

Ítem de evaluación	**Kiruthika et al.[32]**	**Kim[31].**	**Nuestro modelo**
¿Están definidas las características de calidad ?	Yes	Yes	Yes
¿Se definen las métricas y existe un proceso bien definido para ser aplicadas?	No	Yes	Yes
¿Las metricas se adaptan al contexto?	No	No	Yes

6.3.2. Validación con una implementación IoT: Sistema de irrigación para cultivos de palma de aceite

Los resultados aquí mostrados están consignados en el artículo titulado Ïrrigation system for oil palm in Colombia - An Internet of Things approach", el cual (al momento de escribir este documentos) será presentado en el Workshop on Engineering Applications (WEA 2017).

El sistema desarrollado es capaz de crear calendarios de riego basados en el balance hidrológico del suelo. El balance hidrológico del suelo es un modelo que calcula la cantidad de agua almacenada en un lugar, calculada a partir de los flujos de entrada y salida del agua. Los valores necesarios son obtenidos de un conjunto de sensores y de información de canales de meteorología para calcular la cantidad de agua a irrigar necesaria para mantener el nivel del agua a niveles deseados. La Figura 6.2 muestra el diseño de la aplicación.

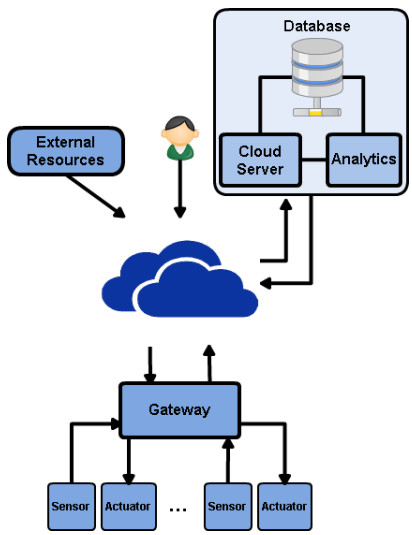

Figura 6.2: Diseño del Sistema de Irrigación

En las Tablas 6.19 y 6.18 se encuentran los resultados para cada característica y sub-característica de calidad el sistema evaluado. Se puede observar en los resultados que la aplicación presenta calidad excelente en compatibilidad, mientras que en seguridad y mantenibilidad se encuentra sus puntos débiles. Esto da como resultado un valor de calidad general de 0,767 que logra mantenerse en niveles considerables de calidad. Para mejorar este valor se podría mejorar en las sub-carcateristicas de calidad de confidencialidad, utilizando los mecanismos de encriptación en todos los módulos. Ademas, se puede mejorar la modularidad utilizando mecanismos de comunicación estandarizados, mejorando también en la conectividad.

Tabla 6.18: Pesos de los α para las sub-características para agro-industria obtenidos en el panel de experto y resultados para el sistema de riego

Características	α	valor	**Características**	α	valor
Comportamiento temporal	α_{11} 0,85	1	Confidencialidad	α_{41} 0,63	0,6
Utilización de recursos	α_{12} 0,83	0,8	Integridad	α_{42} 0,73	1
Capacidad	α_{13} 0,8	0,6	Responsabilidad	α_{43} 0,65	1
Exactitud	α_{14} 0,83	0,6	Autenticidad	α_{44} 0,6	1
Co-existencia	α_{21} 0,7	1	Modularidad	α_{51} 0,73	0,2
Interoperabilidad	α_{22} 0,7	0,6	Reusabilidad	α_{52} 0,78	0,9
Madurez	α_{31} 0,63	1	Escalabilidad	α_{53} 0,8	1
Disponibilidad	α_{32} 0,68	1	Adaptabilidad	α_{61} 0,69	0,6
Tolerancia a fallos	α_{33} 0,72	0,7	Conformidad	α_{62} 0,7	1
Conectividad	α_{34} 0,53	0,6			

Tabla 6.19: Pesos de los α para las características para agro-industria obtenidos en el panel de experto y resultados para el sistema de riego

Características	α	valor
Q_1 Eficiencia de rendimiento	α_1 0,86	0,75
Q_2 Compatibilidad	α_2 0,80	1
Q_3 Fiabilidad	α_3 0,79	0,83
Q_4 Seguridad	α_4 0,76	0,7
Q_5 Mantenibilidad	α_5 0,81	0,71
Q_6 Portabilidad	α_6 0,79	0,8

Finalmente, en la Tabla 6.20 se muestra el valor general de calidad tomando cada juego de pesos para cada línea de trabajo. El valor resultante para todas las líneas de trabajo están en niveles aceptables, por lo que una implementación con la arquitectura utilizada tendría buena calidad en cualquier linea de trabajo.

Tabla 6.20: Valor de calidad general para cada una de las líneas, calculado a partir de los pesos obtenidos del panel de expertos

Línea de trabajo	**Valor de calidad**
Agro-industria y Medio Ambiente	0,767
Salud	0,762
Logística	0,761
Industria	0,757
Vestibles	0,763
Seguridad	0,763
Gobierno	0,769

Capítulo 7.
Conclusiones

A lo largo de esta tesis se siguieron una serie de pasos para poder proponer una metodología para la evaluación de las características, que afectan en la calidad del software para implementaciones dentro del contexto de IoT. Es esta sección se explicaran cuales fueron los resultados obtenidos durante todo el proceso.

Lo primero que se realizó fue una revisión del estado actual de los retos y amenazas que se presentan al momento de realizar una aplicación IoT. A partir de esta revisión se obtuvieron los atributos de calidad mas relevantes en implementaciones de este tipo. Se tabuló la lista de características y en cuales investigaciones fueron mencionadas.

Lo segundo que se realizó fue una revisión del estado actual de las metodologías y modelos para medir calidad tanto para software en general como para aplicaciones de IoT. Se logró identificar que las metodologías y modelos generales no se pueden aplicar a un aplicación IoT, debido a la naturaleza del software. Los modelos actuales para la medición de la calidad de software para IoT dejan por fuera características de

calidad que son importantes en este contexto. Ademas, se vio que por las diferentes lineas de trabajo en IoT no todas las características tienen el mismo pero. Por lo que se decidió proponer un modelo que incluyera todas las características importantes en IoT y que tuviera una forma de adaptarse a cada linea de trabajo.

Se propuso un modelo jerárquico basado en el modelo de la norma ISO/IEC 2501n. Este modelo toma en cuenta las características mas relevantes en la calidad de aplicaciones de IoT. Se propuso una metodología y una métrica a partir del modelo. La metodología agrega una forma para cambiar los pesos de los atributos dependiendo de la linea de trabajo de la solución y se propuso ciertos valores para cada una de las lineas, que pueden ser ajustados dependiendo de requerimientos puntuales de la solución a evaluar. Estos valores fueron obtenidos por la realización de un panel de expertos a los cuales se les preguntó cual es el peso que tiene cada característica y sub-característica en cada una de las lineas de trabajo. El modelo propuesto siempre estará propenso a verificación, debido a que los pesos para cada linea pueden ser modificados y el conjunto de características y sub-características pueden cambiar debido a los cambios en los retos y amenazas para IoT, la evolución de los dispositivos y tecnologías y los problemas externos.

Por ultimo se validó la metodología. Para la validación se realizó una prueba a una aplicación que había sido evaluada por otra metodología mostrando así la di-

ferencia entre los resultados de la nuestra y la otra. Las diferencias mas evidentes aparecen debido a que nuestra metodología le da pesos distintos a las características y sub-características. Otra diferencia que aparece es debido a la forma como están divididas y el nivel de importancia que tienen unas características en un modelo y el otro.

En conclusión,el modelo propuesto cumple con los objetivos trazados al inicio de la investigación. El modelo no solo se enfoca en los factores que afectan a la calidad del tipo de software que se debe realizar para IoT, sino que, ademas, es capaz de amoldarse a los diferentes escenarios que nos presenta este mundo de aplicaciones.

Bibliografía

[1] Líneas de trabajo. http://www.cea-iot.org/lineas-de-trabajo/, 2016. [Online; accessed 219-June-2017].

[2] Defining Software Architecture. http://www.sei.cmu.edu/architecture/index.cfm, 2017. [Online; accessed 29-June-2017].

[3] S. Agrawal and M. L. Das. Internet of things—a paradigm shift of future internet applications. In *Engineering (NUiCONE), 2011 Nirma University International Conference on*, pages 1–7. IEEE, 2011.

[4] W. Aman and E. Snekkenes. Managing security trade-offs in the internet of things using adaptive security. In *2015 10th International Conference for Internet Technology and Secured Transactions (ICITST)*, pages 362–368. IEEE, 2015.

[5] S. Amendola, R. Lodato, S. Manzari, C. Occhiuzzi, and G. Marrocco. Rfid technology for iot-based personal healthcare in smart spaces. *IEEE Internet of Things Journal*, 1(2):144–152, 2014.

[6] L. Bass et al. *Software architecture in practice*. Addison-Wesley Professional, 2012.

[7] L. Catarinucci, D. De Donno, L. Mainetti, L. Palano, L. Patrono, M. L. Stefanizzi, and L. Tarricone. An iot-aware architecture for smart healthcare systems. *IEEE Internet of Things Journal*, 2(6):515–526, 2015.

[8] C. Cecchinel, M. Jimenez, S. Mosser, and M. Riveill. An architecture to support the collection of big data in the internet of things. In *Services (SERVICES), 2014 IEEE World Congress on*, pages 442–449. IEEE, 2014.

[9] V. Chauhan, D. L. Gupta, and S. Dixit. Role of software metrics to improve software quality. *Complexity*, 2(3):6, 2014.

[10] M. G. Cimino, N. Celandroni, E. Ferro, D. La Rosa, F. Palumbo, and G. Vaglini. Wireless communication, identification and sensing technologies enabling integrated logistics: a study in the harbor environment. *arXiv preprint arXiv:1510.06175*, 2015.

[11] W. M. da Silva, A. Alvaro, G. H. Tomas, R. A. Afonso, K. L. Dias, and V. C. Garcia. Smart cities software architectures: a survey. In *Proceedings of the 28th Annual ACM Symposium on Applied Computing*, pages 1722–1727. ACM, 2013.

[12] L. Da Xu, W. He, and S. Li. Internet of things in industries: A survey. *IEEE Transactions on Industrial Informatics*, 10(4):2233–2243, 2014.

[13] D. Evans. The internet of things: How the next evolution of the internet is changing everything. *CISCO white paper*, 1(2011):1–11, 2011.

[14] H. M. Fardoun, A. A. AL-Ghamdi, and A. P. Cipres. Improving surgery operations by means of cloud systems and distributed user interfaces. In *Proceedings of the 2014 Workshop on Distributed User Interfaces and Multimodal Interaction*, pages 31–36. ACM, 2014.

[15] P. Ferreira, R. Martinho, and D. Domingos. Iot-aware business processes for logistics: limitations of current approaches. In *INForum*, pages 611–622, 2010.

[16] H. FIRDOUSE and N. SRINIVAS. Implementation of a reconfigurable smart sensor interface for industrial wsn in iot environment. 2015.

[17] E. Fleisch et al. What is the internet of things? an economic perspective. *Economics, Management, and Financial Markets*, (2):125–157, 2010.

[18] L. Foschini, T. Taleb, A. Corradi, and D. Bottazzi. M2m-based metropolitan platform for ims-enabled road traffic management in iot. *IEEE Communications Magazine*, 49(11), 2011.

[19] N. Gershenfeld. *When things start to think*. Macmillan, 1999.

[20] P. Gope and T. Hwang. Bsn-care: a secure iot-based modern healthcare system using body sensor network. *IEEE Sensors Journal*, 16(5):1368–1376, 2016.

[21] J. Gubbi, R. Buyya, S. Marusic, and M. Palaniswami. Internet of things (iot): A vision, architectural elements, and future directions. *Future generation computer systems*, 29(7):1645–1660, 2013.

[22] M. Hassanalieragh, A. Page, T. Soyata, G. Sharma, M. Aktas, G. Mateos, B. Kantarci, and S. Andreescu. Health monitoring and management using internet-of-things (iot) sensing with cloud-based processing: Opportunities and challenges. In *Services Computing (SCC), 2015 IEEE International Conference on*, pages 285–292. IEEE, 2015.

[23] M. Herbert and B. Halecker. Management of innovation in the industrial internet of things. In *XXVI ISPIM Conference-Shaping the frontiers of Innovation Management in Budapest*. ISPIM, 2015.

[24] I. IEC. Iso/iec 25000 software engineering software product quality requirements and evaluation (square) guide to square. *Systems Engineering*, 41, 2005.

[25] Intel. Optimizing manufacturing with the internet of things, 2014.

[26] S. Islam and P. Falcarin. Measuring security requirements for software security. In *Cybernetic Intelligent Systems (CIS), 2011 IEEE 10th International Conference on*, pages 70–75. IEEE, 2011.

[27] S. R. Islam, D. Kwak, M. H. Kabir, M. Hossain, and K.-S. Kwak. The internet of things for health care: a comprehensive survey. *IEEE Access*, 3:678–708, 2015.

[28] K. Karimi and G. Atkinson. What the internet of things (iot) needs to become a reality. *White Paper, FreeScale and ARM*, pages 1–16, 2013.

[29] R. Kazman, M. Klein, M. Barbacci, T. Longstaff, H. Lipson, and J. Carriere. The architecture tradeoff analysis method. In *Engineering of Complex Computer Systems, 1998. ICECCS'98. Proceedings. Fourth IEEE International Conference on*, pages 68–78. IEEE, 1998.

[30] R. Khan, S. U. Khan, R. Zaheer, and S. Khan. Future internet: the internet of things architecture, possible applications and key challenges. In *Frontiers of Information Technology (FIT), 2012 10th International Conference on*, pages 257–260. IEEE, 2012.

[31] M. Kim. A quality model for evaluating iot applications. *International Journal of Computer and Electrical Engineering*, 8(1):66, 2016.

[32] J. Kiruthika and S. Khaddaj. Software quality issues and challenges of internet of things. In *2015 14th International Symposium on Distributed Computing and Applications for Business Engineering and Science (DCABES)*, pages 176–179. IEEE, 2015.

[33] Kontron. Adding IoT to water works wonders for Malaysian rice growers. Technical report, 2015.

[34] E. P. Management. Ams suite improves plant profitability by more than $ 1.5 million annually, 2011.

[35] Q. M. Marwah and M. Sirshar. Software quality assurance in internet of things. *International Journal of Computer Applications*, 109(9):16–24, 2015.

[36] J. R. Oviedo, M. Rodríguez, and M. Piattini. Certification of ipavement applications for smart cities a case study. In *Evaluation of Novel Approaches to Software Engineering (ENASE), 2015 International Conference on*, pages 244–249. IEEE, 2015.

[37] K. Peffers, T. Tuunanen, M. A. Rothenberger, and S. Chatterjee. A design science research methodology for information systems research. *Journal of management information systems*, 24(3):45–77, 2007.

[38] P. A. Roa, C. Morales, and P. Gutiérrez. Norma iso/iec 25000. *Tecnología Investigación y Academia*, 3(2):27–33, 2016.

[39] F. Shaoshuai, S. Wenxiao, W. Nan, and L. Yan. Modm-based evaluation model of service quality in the internet of things. *Procedia Environmental Sciences*, 11:63–69, 2011.

[40] Z. Sheng, S. Yang, Y. Yu, A. V. Vasilakos, J. A. McCann, and K. K. Leung. A survey on the ietf protocol suite for the internet of things: Standards, challenges, and opportunities. *IEEE Wireless Communications*, 20(6):91–98, 2013.

[41] J. K. Siror, S. Huanye, and W. Dong. Rfid based model for an intelligent port. *Computers in industry*, 62(8):795–810, 2011.

[42] L. M. R. Tarouco, L. M. Bertholdo, L. Z. Granville, L. M. R. Arbiza, F. Carbone, M. Marotta, and J. J. C. de Santanna. Internet of things in healthcare: Interoperatibility and security issues. In *Communications (ICC), 2012 IEEE International Conference on*, pages 6121–6125. IEEE, 2012.

[43] O. Vermesan, P. Friess, P. Guillemin, R. Giaffreda, H. Grindvoll, M. Eisenhauer, M. Serrano, K. Moessner, M. Spirito, L. Blystad, et al. Internet of things beyond the hype: research innovation and deployment. *IERC Cluster SRIA*, 2015.

[44] M. K. Watfa, U. Suleman, and Z. Arafat. Rfid system implementation in jebel ali port. In *Consumer Communications and Networking Conference (CCNC), 2013 IEEE*, pages 950–955. IEEE, 2013.

[45] J. Wei. How wearables intersect with the cloud and the internet of things: Considerations for the developers of wearables. *IEEE Consumer Electronics Magazine*, 3(3):53–56, 2014.

[46] B. Xu, L. Da Xu, H. Cai, C. Xie, J. Hu, and F. Bu. Ubiquitous data accessing method in iot-based information system for emergency medical services. *IEEE Transactions on Industrial Informatics*, 10(2):1578–1586, 2014.

[47] J. Zhang, B. Iannucci, M. Hennessy, K. Gopal, S. Xiao, S. Kumar, D. Pfeffer, B. Aljedia, Y. Ren, M. Griss, et al. Sensor data as a service–a federated platform for mobile data-centric service development and sharing. In *Services Computing (SCC), 2013 IEEE International Conference on*, pages 446–453. IEEE, 2013.

[48] M. Zorzi, A. Gluhak, S. Lange, and A. Bassi. From today's intranet of things to a future internet of things: a wireless-and mobility-related view. *IEEE Wireless Communications*, 17(6):44–51, 2010.

Anexo 1: Resultados del panel de expertos

El objetivo principal del panel de expertos fue obtener un valor que represente la importancia de las características y sub características del modelo de calidad propuesto, en varios de los diferentes ambientes que se presentan en IoT que son listados en las líneas de trabajo de CEA-IoT.

- Salud
- Logística
- Industria
- Vestibles
- Seguridad
- Agroindustria y medio ambiente
- Gobierno

Metodología

La encuesta se aplicó a los miembros del Centro de Excelencia y Apropiación de Internet de las Cosas(CEA-IoT). La cual contó de 175 preguntas. En la encuesta se preguntaba cuál es el valor de importancia que el encuestado piensa que cada característica (6) y sub característica (19) en cada una de las líneas de trabajo (7). El valor de importancia era un número del 0 al 10, donde 0 es menor valor de importancia y 10 el mayor valor de importancia. La encuesta fue respondida por 17 personas, obteniendo los resultados mostrados en la siguiente sección.

Resultados

Los resultados aquí presentados están divididos por líneas de trabajo, presentado el promedio (Prom) y la desviación estándar (DS).

Salud

Tabla 1: Promedio y desviación estandard de las características de salud

Categoría	**Promedio**	**Desviación estandard**
Eficiencia de rendimiento	8,0	2,4
Compatibilidad	7,0	2,5
Confianza	8,8	1,8
Seguridad	8,9	2,3
Mantenibilidad	7,9	1,8
Portabilidad	6,9	2,5

Tabla 2: Promedio y desviación estandard de las sub características de salud

Características	**Prom**	**DS**	**Características**	**Prom**	**DS**
Comportamiento temporal	8,6	1,4	Confidencialidad	8,9	2,3
Utilización de recursos	7,4	1,9	Integridad	8,7	2,2
Capacidad	7,7	2,1	Responsabilidad	8,3	2,6
Exactitud	7,1	2,7	Autenticidad	8,4	2,3
Co-existencia	7,9	1,1	Modularidad	7,4	1,9
Interoperabilidad	8,3	2,2	Reusabilidad	8,0	1,5
Madurez	7,3	2,2	Escalabilidad	7,3	1,9
Disponibilidad	8,5	1,8	Adaptabilidad	6,9	2,9
Tolerancia a fallos	8,3	2,4	Conformidad	6,9	3,5
Conectividad	8,7	1,4			

Logística

Tabla 3: Promedio y desviación estandard de las características de logística

Categoría	**Promedio**	**Desviación estandard**
Eficiencia de rendimiento	8,4	1,9
Compatibilidad	7,2	2,6
Confianza	8,2	1,9
Seguridad	7,9	2,4
Mantenibilidad	7,9	1,6
Portabilidad	6,8	2,3

Tabla 4: Promedio y desviación estandard de las sub características de logística

Características	**Prom**	**DS**	**Características**	**Prom**	**DS**
Comportamiento temporal	8,3	2,0	Confidencialidad	8,0	2,2
Utilización de recursos	8,1	1,3	Integridad	8,5	2,5
Capacidad	8,0	1,3	Responsabilidad	8,2	2,4
Exactitud	8,6	1,5	Autenticidad	8,0	2,2
Co-existencia	8,0	0,9	Modularidad	7,3	1,5
Interoperabilidad	8,2	1,2	Reusabilidad	7,8	1,0
Madurez	7,5	2,0	Escalabilidad	8,3	1,9
Disponibilidad	8,2	1,2	Adaptabilidad	7,9	1,6
Tolerancia a fallos	8,2	2,1	Conformidad	6,7	2,0
Conectividad	8,7	1,5			

Industria

Tabla 5: Promedio y desviación estandard de las características de industria

Categoría	**Promedio**	**Desviación estandard**
Eficiencia de rendimiento	8,6	1,5
Compatibilidad	8,1	1,9
Confianza	9,0	1,5
Seguridad	9,1	1,5
Mantenibilidad	8,4	1,7
Portabilidad	7,6	1,6

Tabla 6: Promedio y desviación estandard de las sub características de industria

Características	**Prom**	**DS**	**Características**	**Prom**	**DS**
Comportamiento temporal	8,5	1,5	Confidencialidad	8,5	2,8
Utilización de recursos	8,0	1,3	Integridad	8,3	2,7
Capacidad	8,2	1,7	Responsabilidad	7,7	2,9
Exactitud	8,5	1,8	Autenticidad	7,8	2,8
Co-existencia	7,7	2,0	Modularidad	7,8	1,2
Interoperabilidad	8,3	2,4	Reusabilidad	7,5	1,2
Madurez	7,8	1,2	Escalabilidad	8,7	1,8
Disponibilidad	8,2	2,2	Adaptabilidad	8,0	1,7
Tolerancia a fallos	8,8	1,6	Conformidad	6,9	1,6
Conectividad	8,5	1,0			

Vestibles

Tabla 7: Promedio y desviación estandard de las características de vestibles

Categoría	**Promedio**	**Desviación estandard**
Eficiencia de rendimiento	7,9	2,0
Compatibilidad	8,0	1,6
Confianza	8,1	1,8
Seguridad	7,6	2,1
Mantenibilidad	7,4	2,4
Portabilidad	8,6	2,3

Tabla 8: Promedio y desviación estandard de las sub características de vestibles

Características	**Prom**	**DS**	**Características**	**Prom**	**DS**
Comportamiento temporal	7,3	1,4	Confidencialidad	5,7	3,3
Utilización de recursos	7,7	1,5	Integridad	6,0	3,3
Capacidad	6,3	2,3	Responsabilidad	6,3	2,6
Exactitud	6,5	2,3	Autenticidad	5,7	3,3
Co-existencia	6,7	2,2	Modularidad	7,3	1,5
Interoperabilidad	6,7	2,4	Reusabilidad	7,7	1,8
Madurez	6,2	2,5	Escalabilidad	6,8	1,0
Disponibilidad	6,3	1,9	Adaptabilidad	7,0	2,9
Tolerancia a fallos	7,0	2,0	Conformidad	6,3	2,5
Conectividad	6,8	2,0			

Seguridad

Tabla 9: Promedio y desviación estandard de las características de seguridad

Categoría	**Promedio**	**Desviación estandard**
Eficiencia de rendimiento	8,9	1,6
Compatibilidad	8,3	1,8
Confianza	9,0	1,5
Seguridad	9,0	1,5
Mantenibilidad	7,5	1,4
Portabilidad	7,1	2,0

Tabla 10: Promedio y desviación estandard de las sub características de seguridad

Características	**Prom**	**DS**	**Características**	**Prom**	**DS**
Comportamiento temporal	8,8	1,5	Confidencialidad	8,3	2,7
Utilización de recursos	8,2	1,3	Integridad	8,2	3,1
Capacidad	8,7	0,8	Responsabilidad	8,2	2,7
Exactitud	9,0	1,1	Autenticidad	8,3	2,3
Co-existencia	8,5	1,4	Modularidad	7,0	1,9
Interoperabilidad	8,7	1,9	Reusabilidad	7,5	1,5
Madurez	8,7	1,4	Escalabilidad	8,0	1,7
Disponibilidad	8,7	1,4	Adaptabilidad	8,0	1,4
Tolerancia a fallos	9,0	2,0	Conformidad	6,7	1,7
Conectividad	7,8	2,1			

Agroindustria y Medio Ambiente

Tabla 11: Promedio y desviación estandard de las características de agroindustria y medio ambiente

Categoría	**Promedio**	**Desviación estandard**
Eficiencia de rendimiento	8,7	1,6
Compatibilidad	8,0	2,1
Confianza	7,9	2,0
Seguridad	7,0	2,7
Mantenibilidad	8,1	1,4
Portabilidad	7,9	1,5

Tabla 12: Promedio y desviación estandard de las sub características de agroindustria y medio ambiente

Características	**Prom**	**DS**	**Características**	**Prom**	**DS**
Comportamiento temporal	8,5	1,4	Confidencialidad	6,3	3,1
Utilización de recursos	8,3	0,8	Integridad	7,3	3,1
Capacidad	8,0	2,1	Responsabilidad	6,5	2,9
Exactitud	8,3	1,5	Autenticidad	6,0	3,2
Co-existencia	7,0	1,8	Modularidad	7,3	1,4
Interoperabilidad	7,0	2,4	Reusabilidad	7,8	1,3
Madurez	6,3	2,5	Escalabilidad	8,0	1,8
Disponibilidad	6,8	1,9	Adaptabilidad	6,9	1,3
Tolerancia a fallos	7,2	2,6	Conformidad	7,0	1,7
Conectividad	5,3	3,5			

Gobierno

Tabla 13: Promedio y desviación estandard de las características de gobierno

Categoría	**Promedio**	**Desviación estandard**
Eficiencia de rendimiento	8,4	2,0
Compatibilidad	8,1	2,4
Confianza	8,9	2,0
Seguridad	9,3	1,5
Mantenibilidad	7,9	2,1
Portabilidad	7,1	2,6

Tabla 14: Promedio y desviación estandard de las sub características de gobierno

Características	**Prom**	**DS**	**Características**	**Prom**	**DS**
Comportamiento temporal	7,3	2,2	Confidencialidad	7,3	4,2
Utilización de recursos	6,5	2,6	Integridad	7,2	4,4
Capacidad	7,2	2,6	Responsabilidad	6.5	4,0
Exactitud	7,5	2,1	Autenticidad	7,2	4,1
Co-existencia	6.5	2,9	Modularidad	6,3	2,3
Interoperabilidad	6,3	3,2	Reusabilidad	7,0	1,9
Madurez	6,7	2,5	Escalabilidad	7,8	1,9
Disponibilidad	6,7	3,4	Adaptabilidad	6,9	2,3
Tolerancia a fallos	6,8	3,3	Conformidad	6,7	2,4
Conectividad	7,0	2,7			

Conclusiones

Se calculó la desviación estandard para cada Característica y sub característica para validar las encuestas realizadas. Las desviaciones estandard no son muy grandes. Ademas, Se verificó que los valores estén en el rango permisible($\mu \pm \sigma$, donde μ es la media y σ es la desviación estandard). Por lo que podemos concluir que los resultados están acordes a pesar de que la encuesta no se hizo de manera colaborativa o en consenso.