MAESTRÍA EN GERENCIA DE SISTEMAS Y TECNOLOGÍA EMPRESARIAL

FORTALECIMIENTO DE LA FÁBRICA DE SOFTWARE IN HOUSE EN LA EMPRESA BANCO ECUADOR

AUTOR

Ing. Carlos Andrés Fernández Paredes

AÑO

2020 - 2021

Resumen

El trabajo desarrollado muestra un análisis de la situación actual de la fábrica de software in house de Banco Ecuador, la cual pertenece a la industria bancaria. Banco Ecuador cuenta con un extenso catálogo de aplicaciones cuyo principal objetivo es brindar un servicio de calidad y permitir a los usuarios la gestión de sus cuentas sin la necesidad de acudir físicamente a sus instalaciones. Para cumplir este objetivo, la organización cuenta con herramientas tecnológicas y marcos de trabajo acorde a las necesidades.

Dentro de sus operaciones se han encontrado problemas relacionados a la seguridad en el desarrollo de software, al bajo porcentaje de efectividad de los equipos asignados a proyectos de software, mala planificación, tecnología obsoleta en algunos casos y una nula gestión de documentos los cuales impiden que se desarrollen las cosas conforme a lo planificado.

Después del análisis realizado se proponen ciertas iniciativas que ayudarán a solventar esta problemática. Estas iniciativas están definidas bajo las arquitecturas que recomienda TOGAF y, además se fundamentan en las mejores prácticas sugeridas por marcos de trabajo ágiles como Scrum y DevOps y referentes en cuanto a desarrollo de software como CMMI Dev 1.3 y OWASP.

Finalmente se diseña un mapa de ruta y una planificación que ayudarán a implementar los proyectos planteados.

Abstract

The work developed shows an analysis of the current situation of Banco Ecuador's in-house software factory, which belongs to the banking industry. Banco Ecuador has an extensive catalog of applications whose main objective is to provide a quality service and allow users to manage their accounts so there is no need to physically go to its facilities. To reach this goal, the organization has technological tools and frameworks according to the needs.

Within its operations, problems related to security in software development have been found, the low percentage of effectiveness of the teams assigned to software

projects, poor planning, obsolete technology in some cases and no document management which prevent things from going according to plan.

After the analysis carried out, a few initiatives are proposed that will certainly help solve this problem. These initiatives are defined under the architectures recommended by TOGAF and are also based on the best practices suggested by agile frameworks such as Scrum and DevOps and references in terms of software development such as CMMI Dev 1.3 and OWASP.

Finally, a road map and planning were designed to assist with the implementation of the proposed projects.

Índice del contenido

Figuras

Tablas

1. Fase Preliminar

1.1. Contexto

Según la publicación de octubre del 2019 de “Innovation in Retail Banking” Figura 1, las proyecciones para los siguientes 5 años se enfocan en estrategias de Open Banking, migración de clientes de canales físicos a digitales, entre otros, evidenciando la preocupación por el fortalecimiento del área tecnológica.

Figura 1. Prioridades estratégicas bancarias para el futuro. Tomado de (Marous, J., 2019).

Este enfoque ha sido tomado por algunas instituciones bancarias nacionales e internacionales, y sus estrategias de transformación digital están direccionadas a ofrecer productos y servicios basados en el desarrollo o fortalecimiento de tecnologías. Se puede evidenciar esta tendencia en el reporte “Digital Banking Report Research January 2020” Figura 2, en donde, las primeras cuatro estrategias están relacionadas a la aplicación de productos digitales.

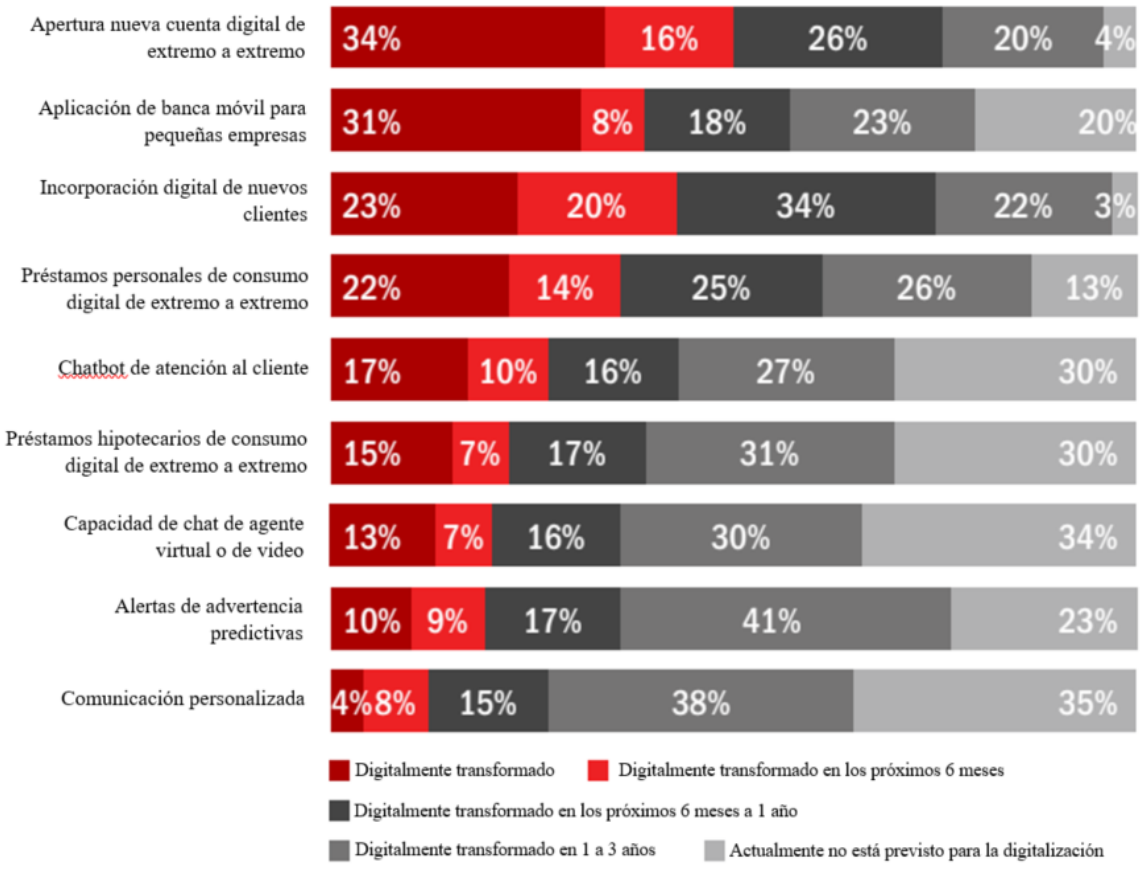


Figura 2. Estado de las estrategias de transformación digital. Tomado de (Marous, J., 2019)

Banco Ecuador es una entidad financiera líder en el mercado con presencia nacional e internacional, con más de 100 años de experiencia. Cuenta con casi 4

millones de clientes de los cuales más de 2.4 millones son usuarios afiliados a canales digitales y más de un millón son activos.

Banco Ecuador ha establecido una primera versión de una fábrica de *software in house* con no más de 2 años en actividades. La forma en la que está organizada es en tribus, cada tribu tiene un equipo de trabajo bajo el esquema propuesto por *Scrum*.

Actualmente tiene algunas opciones de canales digitales, que son de ayuda a los usuarios evitando la presencia física en sus agencias para interactuar con los servicios que ofrece el banco. Los principales canales digitales con los que cuenta son:

- Banca Web.
- Banca Móvil.
- Apertura de cuentas.
- Billetera Móvil.
- Solicitud y acreditación de préstamos.
- Solicitud de tarjetas de crédito.

Para poder mantener estos productos y servicios en producción, mejorarlos y crear nuevos, el banco cuenta con un equipo tecnológico conformado por personal interno de alrededor de 200 personas y personal externo distribuido en más de 8 diferentes proveedores.

Los problemas actuales que posee la empresa entorno a la fábrica de *software in house* son:

- El 35% de servicios de autogestión se lo hace a través de canales digitales, el 65% restante se lo hace en agencias o canales físicos.
- El 80% de proyectos no tiene establecido un estándar de código seguro.
- No se tiene un repositorio de componentes comunes para los equipos de desarrollo.

- El porcentaje de efectividad de los equipos de desarrollo en un Sprint es del 52%.
- No se tiene una pronta respuesta a solicitudes levantadas a proveedores.
- Existe mucha replanificación en medio del desarrollo de un *Sprint*.
- Las historias de usuario no se encuentran documentadas a detalle.
- El *backlog* que se tiene es de menos de 2 meses.
- Los tiempos de respuesta de información entre departamentos son de más de 2 días.
- La información que se maneja en los departamentos no es centralizada.

1.2. *Stakeholders* y expectativas de valor

Tabla 1
Stakeholders y expectativas de valor

Stakehloder	**Poder decisión**	**Nivel interés**	**Expectativa de valor**
Gerencia General	Alto	Medio	Potenciar los canales digitales disponibles y sus planes de mantenimiento y liberación de nuevas funcionalidades. Aumentar la satisfacción del cliente.
Gerencia Financiera	Alto	Alto	Poder reducir el gasto operativo y de personal.
Gerencia Seguridad	Alto	Alto	Desarrollo de software seguro. Análisis de probabilidad de ocurrencia e impacto sobre los riesgos del software desarrollado.
Gerencia de TI	Alto	Alto	Ambiente común de desarrollo. Fortalecer el monitoreo y control del nivel del servicio.
Gerencia de Canales	Medio	Medio	Alta disponibilidad de canales digitales. Fortalecer el proceso para crear nuevos productos o servicios competitivos y potenciar los existentes.
Gerencia de Gestión de Proyectos	Medio	Medio	Proyectos ejecutados en los tiempos establecidos. Mejorar el orden en la gestión de proyectos.

Gerente de Marketing	Medio	Medio	Respaldar las campañas publicitarias con el desarrollo de software de calidad.
Cliente	Bajo	Alto	Contar con un producto funcional e intuitivo. Alta disponibilidad del servicio utilizado.

1.3. Organización impactada

Las áreas afectadas están divididas en cuatro grandes grupos: impacto mayor, impacto menor, impacto extendido y comunidades afectadas con base en el análisis de stakeholder, su poder de decisión, nivel de interés y las expectativas de valor esperadas. Se encuentran distribuidas de la siguiente manera Figura 3.

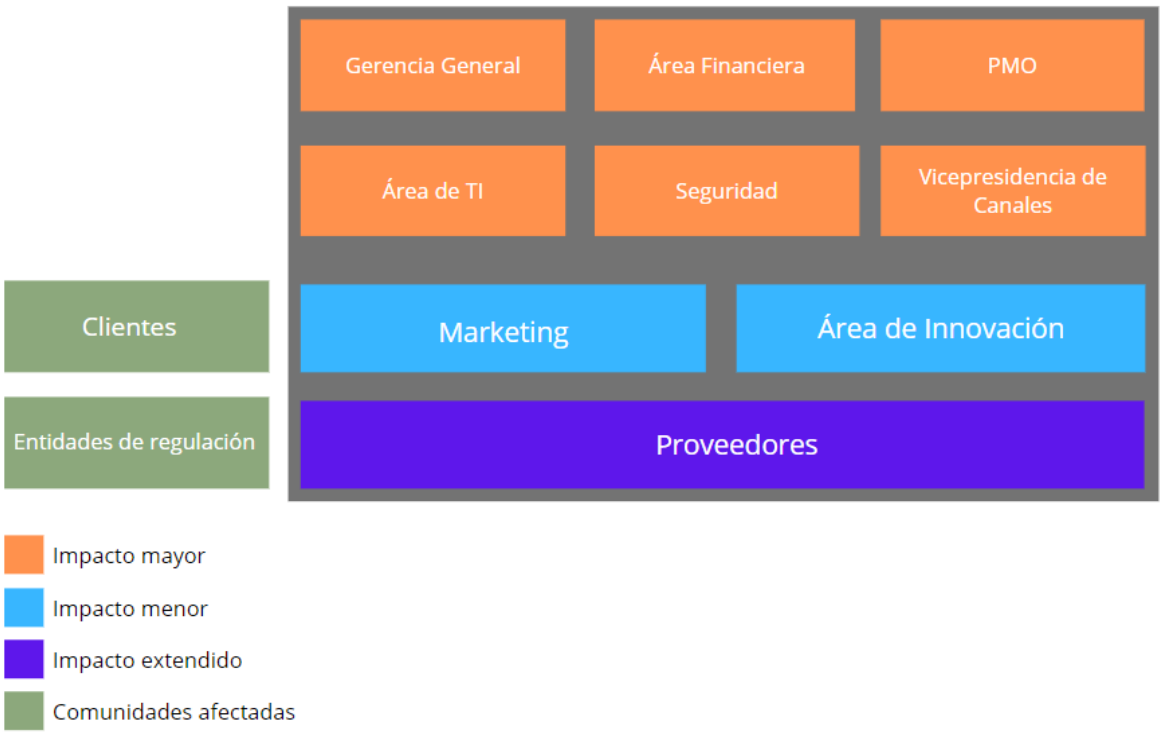


Figura 3. Organización impactada. Para el análisis de la organización impactada, se agruparon las áreas.

1.3.1. BMM

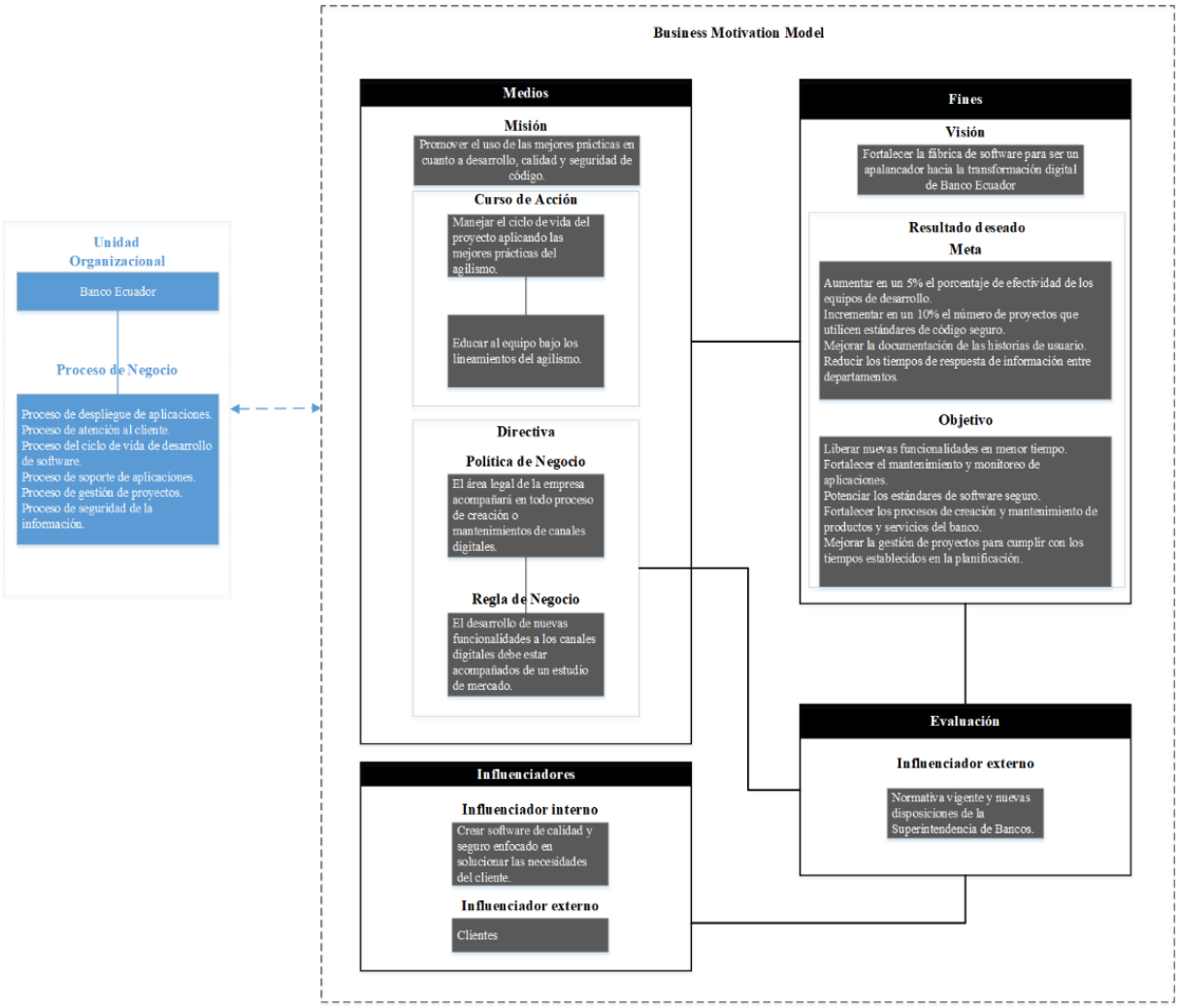


Figura 4. Business Model Motivation de Banco Ecuador

1.4. Marcos de referencia complementarios

Tabla 2
Marcos de referencia complementarios

Área	Referente	Descripción
Ciclo de vida de aplicaciones	CMMI Dev 1.3	Es un modelo que consta de buenas prácticas para mejorar y evaluar los procesos para el desarrollo, mantenimiento y puesta en producción de productos o servicios de *software*.
Agilismo	*Scrum*	Marco de trabajo simple que promueve la colaboración en los equipos para lograr desarrollar productos complejos.
Despliegue de aplicaciones	DevOps	Metodología de desarrollo de *software* centrada en la comunicación, colaboración e integración entre desarrolladores de *software* y profesionales de negocio. Ayuda a las organizaciones a producir productos y servicios de *software* más rápido, con mejor calidad y a un coste menor.
Seguridad para desarrollo de aplicaciones	OWASP	Buenas prácticas enfocadas en la seguridad de aplicaciones. Se dedica a la búsqueda y lucha contra las causas del *software* inseguro.

1.5. Equipo de arquitectura

El equipo de arquitectura estará definido de la siguiente manera:

Comité de Arquitectura Empresarial

Será el responsable de dar los lineamientos para el plan de migración y asegurar su ejecución en beneficio de la empresa para cumplir los objetivos estratégicos. Es la máxima autoridad en cuanto a la toma de decisiones sobre el análisis de este capstone.

Gerente Técnico (Patrocinador)

Se encargará de que todo el ejercicio que se aplique en la organización cuente con los recursos necesarios y las aprobaciones requeridas para la implementación.

Líder de arquitectura

Este rol tiene a cargo las siguientes responsabilidades:

- Establecer un grupo de excelencia que asegure el cumplimiento de los beneficios ofrecidos en el desarrollo de este ejercicio y coordinar todas sus actividades.
- Direccionar el fortalecimiento de la capacidad de arquitectura en la ejecución de las iteraciones y fases de este ejercicio.

Arquitectos de dominio

Este rol está conformado por el Arquitecto de Negocio, Arquitecto de Aplicaciones, Arquitecto de Información y Arquitecto de Tecnología. Serán los encargados de definir para todos los dominios las especificaciones e interacciones entre los diferentes componentes de su dominio.

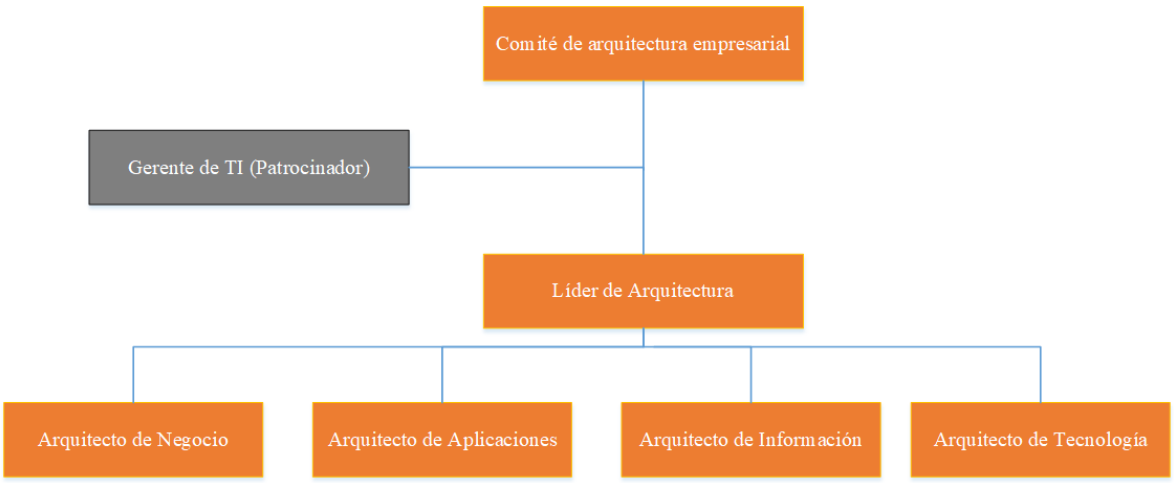


Figura 5. Equipo de arquitectura de Banco Ecuador

1.6. Catálogo de principios

Basados en los principios definidos en TOGAF, se especifican los siguientes para Banco Ecuador (The Open Group. 2018, p. 201-212)

Principios de negocio

Tabla 3
Gestión de la Información de la empresa

Nombre	**Gestión de la Información de la empresa**
Reference	BP01
Declaración	Las decisiones de gestión de la información se toman para proporcionar el máximo beneficio a toda la empresa.
Razón fundamental	Este principio se enfoca en la generación de valor a largo plazo a partir de las decisiones tomadas desde una perspectiva de toda la empresa. Para maximizar los beneficios se requiere que las decisiones de gestión de la información se alineen con las prioridades de toda la empresa.
Implicaciones	Para lograr este principio se requerirá cambios en la forma en la que se planifica y administra la información, si bien se tienen sistemas que gestionan la información hay que tener en cuenta que no se puede dejar toda la responsabilidad a la tecnología. Las prioridades de desarrollo de aplicaciones deben ser establecidas por toda la empresa y deben ser para toda la empresa. A medida que surgen necesidades se deben ajustar las prioridades, estas decisiones deben ser tomadas por un grupo de dueños de procesos de toda la empresa.

Nota. Tomado de (The Open Group. 2018, p. 201-212)

Tabla 4
Continuidad del negocio

Nombre	**Continuidad del negocio**
Reference	BP02
Declaración	Las operaciones empresariales se mantendrán a pesar de las interrupciones de los sistemas.
Razón fundamental	A medida que los sistemas ofrecen más servicios, las empresas se vuelven más dependientes de ellos, por lo tanto, se debe considerar la confiabilidad de los sistemas en su diseño, uso y disponibilidad. Los servicios ofrecidos por la empresa no se deben interrumpir o detener a pesar de la indisponibilidad de los sistemas, fallas a nivel de hardware, eventos externos o corrupción de datos.
Implicaciones	Las aplicaciones deben ser evaluadas tomando en cuenta la criticidad e impacto en la misión de la empresa para poder determinar el nivel de continuidad que requiere y el plan de recuperación correspondiente. Al momento del diseño de las aplicaciones se debe tomar en cuenta la recuperabilidad, la redundancia y la mantenibilidad. Se debe hacer periódicamente pruebas de vulnerabilidad. Asegurar la continuidad del negocio mediante capacidades redundantes o alternativas.

Nota. Tomado de (The Open Group. 2018, p. 201-212)

Tabla 5
Responsabilidades de TI

Nombre	**Responsabilidades de TI**
Reference	BP03
Declaración	El área de TI de la empresa es la responsable de poseer e implementar procesos de infraestructura necesarios que permitan que las soluciones cumplan con los requisitos de funcionalidad, nivel de servicio, costo y tiempo de entrega definidos por los usuarios.
Razón fundamental	Alinear de manera eficaz las expectativas con las capacidades y los costos para que todos los proyectos sean rentables.
Implicaciones	Se debe crear un proceso para priorizar proyectos. Se deben crear modelos de datos, aplicaciones y tecnología para permitir soluciones de calidad integradas y maximizar los resultados.

Nota. Tomado de (The Open Group. 2018, p. 201-212)

Principios de datos

Tabla 6
Definición de datos y vocabulario común

Nombre	Definición de datos y vocabulario común
Reference	DP01
Declaración	Los datos se deben definir de forma coherente en toda la empresa y las definiciones deben ser comprensibles y estar disponibles para todos los usuarios.
Razón fundamental	Los datos que se utilizarán en el desarrollo de aplicaciones deben tener una definición común en toda la empresa para permitir el intercambio de datos. Es necesario conectar sistemas para el intercambio de datos.
Implicaciones	La empresa debe establecer el vocabulario común inicial para el negocio. Las definiciones se utilizarán de manera uniforme en toda la empresa. El administrador de datos debe coordinar lo necesario en caso de que se requiera una nueva definición de datos.

Nota. Tomado de (The Open Group. 2018, p. 201-212)

Tabla 7
Seguridad de los datos

Nombre	**Seguridad de los datos**
Reference	DP02
Declaración	Los datos deberán estar protegidos contra el uso y la divulgación no autorizada.
Razón fundamental	El intercambio y divulgación de información debe estar restringida con base en la legislación pertinente. La información debe estar clasificada y restringida con base en su criticidad y contenido sensible.
Implicaciones	Los propietarios y usuarios finales de los datos deben determinar si las modificaciones sobre los mismos dan como resultado un mayor nivel de clasificación. Políticas y procedimientos para la clasificación y revisión de los datos. Se debe usar un sistema para almacenar los datos clasificados. Se debe implementar reglas de seguridad de los datos para restringir el acceso a vista, revisión y edición.

Nota. Tomado de (The Open Group. 2018, p. 201-212)

Principios de Aplicación

Tabla 8
Independencia tecnológica

Nombre	**Independencia tecnológica**
Reference	AP01
Declaración	Las aplicaciones pueden operar en cualquier plataforma tecnológica.
Razón fundamental	Las aplicaciones desarrolladas por terceros sin pensar en los requisitos del usuario se convierten en un impulsor y no en un apalancador para el logro de los objetivos empresariales. Potenciar la independencia de aplicaciones de la tecnología subyacente, permitiendo que las aplicaciones se desarrollen, actualicen, mantengan y operen de la manera más rentable y oportuna. Garantizar que el desarrollo de software no dependa de hardware o software de sistemas operativos específicos.
Implicaciones	Creación de APIs para la interoperabilidad entre aplicaciones y entornos operativos desarrollados bajo una arquitectura empresarial.

Nota. Tomado de (The Open Group. 2018, p. 201-212)

Tabla 9
Fácil uso

Nombre	**Fácil uso**
Reference	AP02
Declaración	Las aplicaciones son fáciles de usar para el cliente.
Razón fundamental	La facilidad de uso es un incentivo positivo para el uso de las aplicaciones por parte de los usuarios. Es necesario involucrar a los usuarios a trabajar de la mano del equipo técnico para evitar el desarrollo de sistemas aislados, de esta manera el riesgo de utilizar un sistema de forma incorrecta es bajo.
Implicaciones	Se debe desarrollar un estándar común para el look and feel de las aplicaciones y asegurar que cumplan los requisitos ergonómicos. Además, se deben desarrollar criterios de pruebas de usabilidad. Los servicios que ofrecen las aplicaciones no deben verse limitados por ubicación del usuario, idioma, capacitación en sistemas o la capacidad física.

Nota. Tomado de (The Open Group. 2018, p. 201-212)

Principios de Tecnología

Tabla 10
Cambio basado en requisitos

Nombre	**Cambio basado en requisitos**
Reference	TP01
Declaración	Los cambios en aplicaciones y tecnología deben hacerse solo en respuesta a las necesidades comerciales.
Razón fundamental	Este principio establecerá una cultura en la que el entorno de la información cambie en respuesta a las necesidades del negocio, en lugar de que el negocio cambie en respuesta a los cambios de TI. De esta manera se minimizarán los efectos no deseados en el negocio debido a cambios de TI.
Implicaciones	Los cambios en la implementación deben hacerse utilizando la arquitectura empresarial. No financiar una mejora técnica o el desarrollo de un sistema si no se tiene una necesidad comercial documentada. Asegurarse de que el proceso de documentación de requisitos no obstaculice el flujo para satisfacer las necesidades comerciales legítimas. El principal propósito de este principio es mantenernos enfocados en los negocios y no en las tecnologías.

Nota. Tomado de (The Open Group. 2018, p. 201-212)

Tabla 11
Interoperabilidad

Nombre	**Interoperabilidad**
Reference	TP02
Declaración	El software y hardware deben alinearse a estándares que promuevan la interoperabilidad de datos, aplicaciones y tecnología.
Razón fundamental	Garantizar la coherencia a través de estándares, mejorando la capacidad de administrar sistemas y mejorar la satisfacción del usuario y a su vez proteger las inversiones de TI. Garantizar el soporte de múltiples proveedores para los productos de la organización y facilitar la integración de la cadena de suministro.
Implicaciones	Se deben seguir los estándares de interoperabilidad y los estándares de la industria a menos que exista una razón comercial para no aplicar una solución estándar. Establecer un proceso para la definición, revisión y mantenimiento de estándares. Documentar e identificar las plataformas de TI existentes.

Nota. Tomado de (The Open Group. 2018, p. 201-212)

2. Visionamiento Arquitectónico

El presente capítulo muestra un análisis de dos casos de éxito que serán tomados como referentes debido a la similitud con el concern. A partir de lo cual se obtiene el resultado del visionamiento, el análisis de brechas y una aproximación a alto nivel de los escenarios de solución.

2.1. Casos de éxito

Existen excelentes referentes de fábricas de *software* a nivel mundial. Para el análisis del visionamiento se tomó como referencia el caso de éxito de Netflix debido a la similitud con la industria bancaria en cuanto a disponibilidad, omnicanalidad, alto nivel de transaccionalidad, seguridad e integridad de información del usuario y a BBVA como referente de fábrica de software bajo un marco de trabajo ágil como *Scrum*.

2.1.1. Netflix

Netflix empezó este camino en 2008 haciendo una migración de sus aplicaciones hacia la nube y adoptando la arquitectura de microservicios (Izrailevsky Y, 2016). Las estrategias adoptadas se traducen en las siguientes características, agrupadas bajo las metas y prácticas genéricas de CMMI Dev 1.3 propuestas para este análisis:

- (CM) Gestión de configuración (Software Engineering Insitute, 2010, p. 243-255).
    - Control de hardware y equipamiento.
    - Configuración de herramientas de desarrollo.
    - Herramientas de pruebas y scripts de pruebas.
- (IPM) Gestión integrada del proyecto (Software Engineering Insitute, 2010, p. 267-285).
    - Correcto establecimiento del entorno de trabajo.
- (PMC) Monitorización y control del proyecto (Software Engineering Insitute, 2010, p. 393-402).
    - Manejo de alertas para prevenir problemas.
    - Monitoreo de cada servicio.
    - Aislamiento de problemas para no afectar la ejecución de otros servicios.

- (PP) Planificación del proyecto (Software Engineering Insitute, 2010, p. 403-424).
    - Identificación de los riesgos del proyecto.
    - Interés en el fortalecimiento de habilidades profesionales del personal.
- (RD) Desarrollo de requisitos (Software Engineering Insitute, 2010, p. 455-472).
    - Entender las verdaderas necesidades del cliente.
- (TS) Solución técnica (Software Engineering Insitute, 2010, p. 509-530).
    - Modificaciones de código más fáciles.
    - Despliegues más rápidos.
    - Análisis de nuevas tecnologías para obtener una ventaja competitiva.
    - Desarrollo de código seguro.
- (VAL) Validación (Software Engineering Insitute, 2010, p. 531-540).
    - Adopción de la ingeniería del caos.
    - Automatización de pruebas unitarias y de integración.
- (VER) Verificación (Software Engineering Insitute, 2010, p. 541-552).
    - Adopción de revisión entre pares.

2.1.2. BBVA

BBVA inició este viaje para convertirse en una organización ágil en el año 2014 y hasta el día de hoy este proceso continúa. Toda la empresa opera bajo la metodología ágil a través del marco de trabajo *Scrum*, el cual ayuda fomentando la autonomía de los equipos y adoptando los roles y responsabilidades de cada miembro (BBVA en 2014, p. 33).

La adopción de esta filosofía ayudó a que el incremento de productividad se triplique, a mejorar las habilidades de liderazgo de sus colaboradores y a contagiar esta cultura a más de 16000 empleados de BBVA.

2.2. Resultado del visionamiento

Con base en los casos de éxito descritos, a las mejores prácticas de los marcos de trabajo de referencia y considerando el tamaño del Banco Ecuador se puede fortalecer la fábrica de *software in house* de la siguiente manera:

- Se fortalecerá la metodología ágil de desarrollo de software *Scrum* respetando los roles y responsabilidades de cada miembro del equipo y participando de manera activa de todas las ceremonias. Según (Menzinsky A y López G. 2019) el marco técnico de *Scrum* está formado por
    - Roles:
        - El equipo *Scrum*.
        - El dueño del producto o *product owner*.
        - El *Scrum Master*.
    - Artefactos:
        - Pila del producto o *sprint backlog*.
        - Pila del *sprint* o *sprint backlog*.
        - Incremento.
    - Eventos:
        - *Sprint*.
        - Reunión de planificación del *sprint*.
        - *Scrum* diario.
        - Revisión del *sprint*.
        - Retrospectiva del *sprint*.
- Se aplicarán las siguientes metas y prácticas genéricas (Software Engineering Insitute, 2010) para fortalecer el ciclo de vida de la gestión del desarrollo de software:
    - (CM) Gestión de configuración.
    - (IPM) Gestión integrada del proyecto.
    - (PMC) Monitorización y control del proyecto.
    - (PP) Planificación del proyecto.

- (RD) Desarrollo de requisitos.
- (REQM) Gestión de requisitos.
- (TS) Solución técnica.
- (VAL) Validación.
- (VER) Verificación.

Dentro de CMMI Dev 1.3 se integrarán prácticas de DevOps con los siguientes objetivos:

- Fortalecer la cultura de colaboración entre el equipo técnico y el de negocio.
- Fortalecer el proceso de integración y despliegue continuo mediante el uso de herramientas tecnológicas.

- Para desarrollar aplicaciones seguras es necesario aplicar los siguientes controles de (Kang et al. 2005. *The Open Web Application Security*) con base en el tamaño de la empresa y en el tipo de aplicaciones que está desarrollando:
  - Metodología de desarrollo.
  - Estándares de codificación.
  - Control de código fuente.
  - Arquitectura de seguridad.
  - Control de auditoría.

El diagrama conceptual de la solución propuesta se detalla en el Anexo 1.

2.3. Análisis de brechas

Para poder encontrar un estado actual, un referente y un estado esperado, es necesario hacer un análisis de brechas. Los referentes que se usarán se muestran en la siguiente tabla 12:

Tabla 12
Referentes

Por analizar	**Referentes**
Ciclo de vida de aplicaciones	CMMI Dev 1.3
Agilismo	Scrum DevOps
Seguridad para desarrollo de aplicaciones	OWASP

La rúbrica para la valoración se muestra en la siguiente tabla 13:

Tabla 13
Rúbrica

Puntaje	Observaciones
0	Falta de cualquier capacidad básica.
1	Se logra de manera inicial o intuitiva sin apoyo tecnológico.
2	Logra su propósito, pero sin seguir los estándares del referente. Cuenta con documentación parcial y algunos indicadores de gestión.
3	Logra su propósito de manera más organizada. Tiene documentación formal e indicadores de gestión. Se apoya de herramientas tecnológicas.
4	Logra su propósito, está bien definido y mide su rendimiento. Es guiado por buenas prácticas y adopta tecnologías referentes.
5	Logra su propósito, está bien definido, se mide su rendimiento para mejorar el desempeño y se busca la mejora continua.

2.3.1. Ciclo de desarrollo de aplicaciones

Tabla 14
Planificación del proyecto

	Actual	Objetivo	Referente	Brechas a cerrar
Gestión de configuración	1	2	5	Generar líneas base de código. Documentar elementos de trabajo que son dependientes entre sí. Estandarizar planes y procedimientos de pruebas.
Gestión integrada del proyecto	1	3	5	Fortalecer el proceso de onboarding a nivel tecnológico. Fortalecer DoR, DoD de las tareas para saber su inicio y fin. Implementar informes periódicos del estado del equipo.
Monitorización y control del proyecto	2	3	5	Establecer medidas a nivel del personal para conocer la adquisición de conocimiento y de las habilidades.
Planificación del proyecto	2	3	5	Fortalecer las estimaciones de los atributos de los productos de trabajo y de las tareas. Tomar en cuenta los recursos de infraestructura para la planificación del proyecto.

Nota. Brechas encontradas correspondientes a la planificación del proyecto.

Tabla 15
Gestión de requisitos

	Actual	Objetivo	Referente	Brechas a cerrar
Desarrollo de requisitos	2	3	5	Fortalecer el desarrollo de requisitos técnicos necesarios para el diseño del producto final. Fortalecer la validación de requisitos.
Gestión de requisitos	2	3	5	Establecer criterios objetivos para la evaluación y aceptación de los requisitos. Realizar una evaluación de los cambios de requisitos desde el punto de vista de los stakeholders relevantes. Establecer una correcta trazabilidad bidireccional de los requisitos.

Nota. Brechas encontradas correspondientes a la gestión de requisitos.

Tabla 16
Solución técnica

	Actual	Objetivo	Referente	Brechas a cerrar
Solución técnica	2	3	5	Identificar las tecnologías que están en uso actualmente y las nuevas tecnologías de producto en cuanto a ventajas competitivas. Encontrar componentes de solución reutilizables. Definir formalmente el comportamiento e interacción de los componentes usando un lenguaje de descripción de la arquitectura. Establecer los criterios de evaluación para los componentes de los productos en pos de decidir si se deberían desarrollar, comprar o reutilizar. Usar métodos eficaces para la implementación de los componentes del producto.

Nota. Brechas encontradas correspondientes a la solución técnica.

Tabla 17
Control de calidad y soporte

	Actual	Objetivo	Referente	Brechas a cerrar
Validación	2	3	5	Fortalecer el proceso de selección de productos para la evaluación. Hacer un seguimiento periódico de los procedimientos y criterios de evaluación.
Verificación	2	3	5	Fortalecer el proceso de selección de productos para la verificación. Fortalecer los métodos de verificación.
Soporte de aplicaciones	1	3	5	Revisión de contratos de nivel de servicio. Fortalecer el proceso de aceptación y cierre de mejoras o modificaciones de los productos o servicios.

Nota. Brechas encontradas correspondientes al control de calidad y el soporte de aplicaciones.

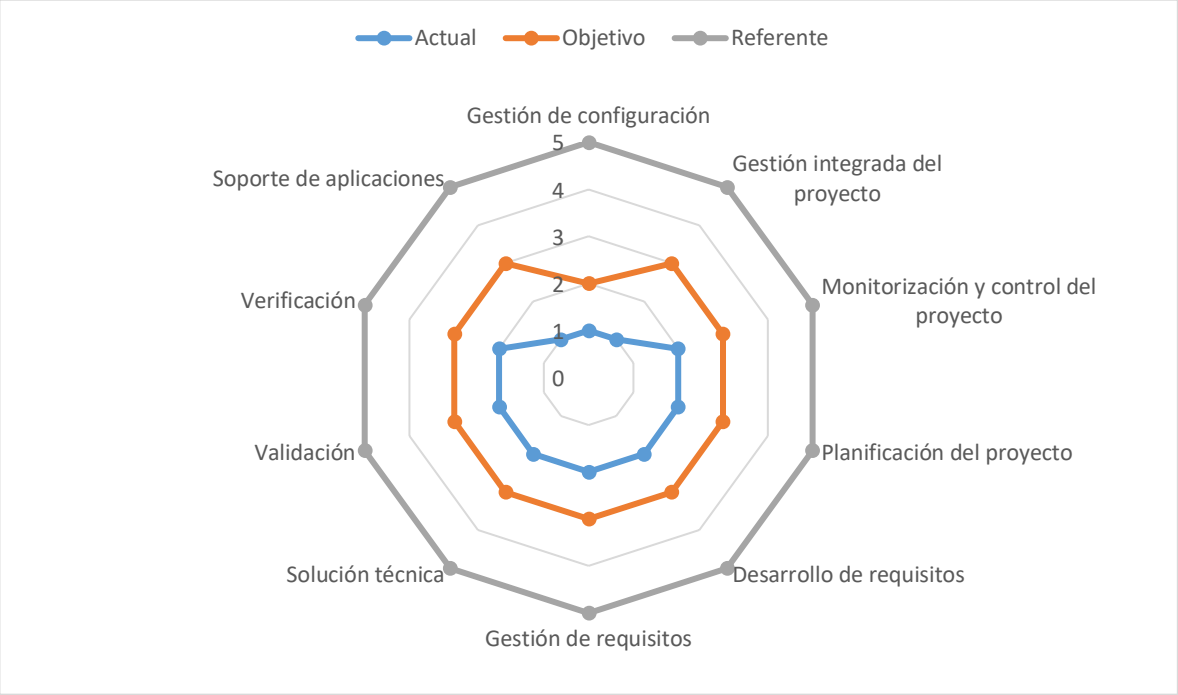


Figura 6. Análisis de brechas del Ciclo de desarrollo de aplicaciones

2.3.2. Agilismo

Tabla 18
Agilismo

	Actual	Objetivo	Referente	Brechas por cerrar
Definición de Rol de dueño del producto	2	3	5	Mejorar el contacto con el equipo de desarrollo
Definición de backlog para un sprint	2	3	5	Actualizar el estado de las tareas diariamente
Asistencia a ceremonias	2	4	5	Fortalecer el compromiso del equipo con la metodología. Explotar el rol de Scrum Master dentro del equipo.
Definición de terminado	2	5	5	Definir características que determinen la finalización de una tarea. Respetar las definiciones acordadas. Hacer visible estas definiciones.
Definición de product backlog	1	3	5	Establecer los ítems que deben ser parte del product backlog. Fortalecer la definición de los ítems.
Manejo de iteraciones	3	3	5	
Distribución del equipo de trabajo	3	4	5	Reducir el tamaño de los equipos de trabajo cuando exceda a lo recomendado por Scrum. Ubicar en un mismo lugar a todo el equipo.

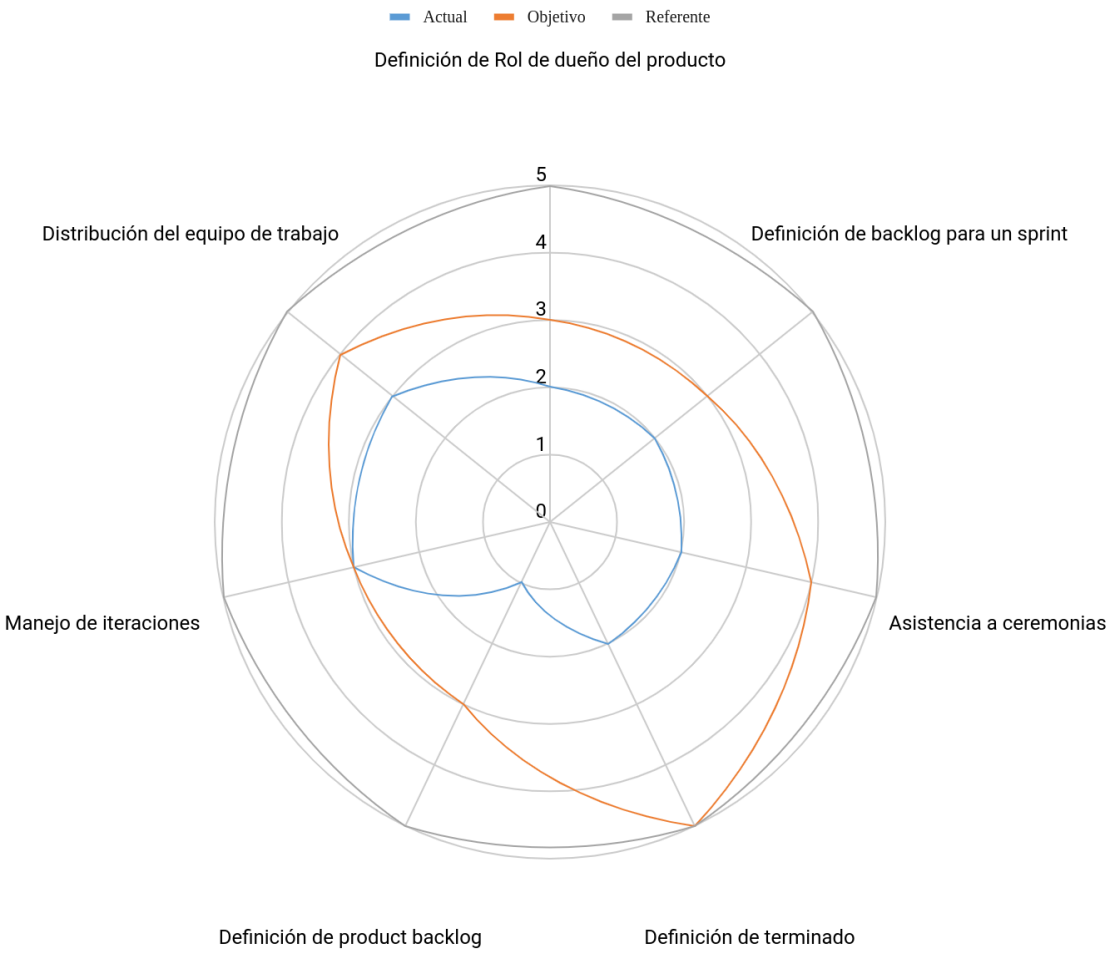


Figura 7. Análisis de brechas del agilismo

2.3.3. Seguridad para desarrollo de aplicaciones

Tabla 19
Seguridad para desarrollo de aplicaciones

	Actual	Objetivo	Referente	Brechas por cerrar
Metodología de desarrollo	2	3	5	Fortalecer el testeo y documentación del código. Reducir la tasa actual de errores.
Estándares de codificación	3	4	5	Medir el rendimiento del código al aplicar los estándares definidos.
Control de código fuente	3	4	5	Ejecutar set de pruebas automatizadas a bloques de códigos versionados.
Arquitectura de seguridad	3	4	5	Aplicar pruebas de seguridad en ambientes de desarrollo, pruebas y producción
Control de auditoría	4	5	5	Fortalecer la encriptación de datos en logs de Auditoría. Mayor restricción de acceso a registros de log.

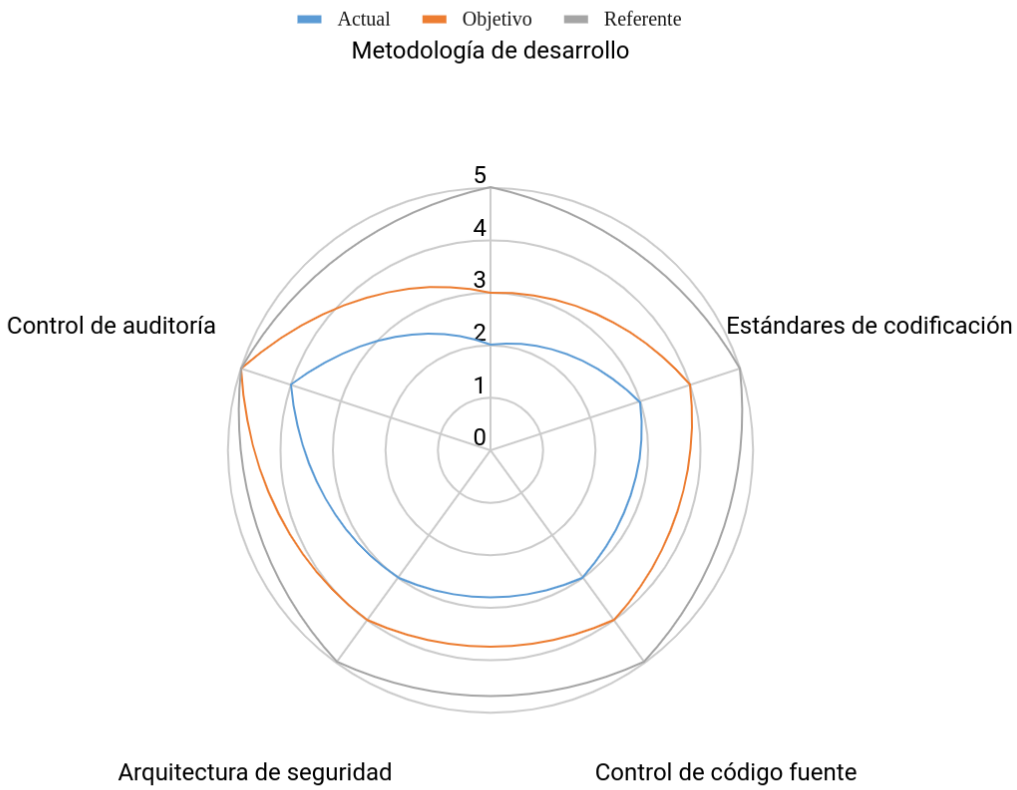


Figura 8. Análisis de brechas de Seguridad para desarrollo de aplicaciones

2.4. Definición de arquitectura objetivo

2.4.1. Arquitectura de negocio

La arquitectura de negocio objetivo se enfocará en definir los procesos necesarios para fortalecer la fábrica de software en Banco Ecuador dividiendo la empresa en dos categorías:

- Ciclo de desarrollo de software.
- Talento Humano.

En la figura 9 se muestra la arquitectura de negocio objetivo integrando CMMI Dev 1.3 y Scrum.

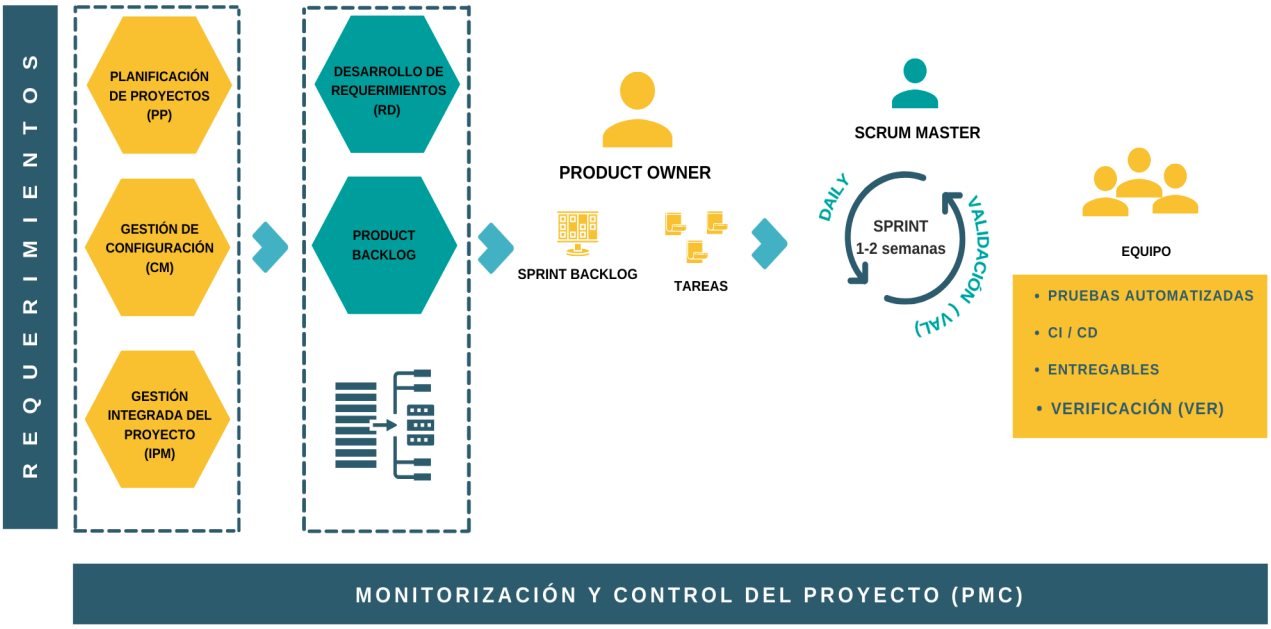


Figura 9. Arquitectura de negocio objetivo. Se tiene una integración entre las mejores prácticas de CMMI Dev 1.3 y *Scrum*

2.4.2. Arquitectura de datos

Los datos deberán ser reclasificados con base en su nivel de confidencialidad para definir el grupo de usuarios que tendrá acceso, y serán almacenados en Confluence, una solución de Atlassian.

Confluence permitirá almacenar de manera segura la información de Banco Ecuador, las principales ventajas de esta herramienta son:

- Edición en tiempo real.
- Creación de notificaciones.
- Agregar comentarios sobre la información almacenada.
- Permisos.

2.4.3. Arquitectura de aplicaciones

Las aplicaciones que deberán ser implementadas están clasificadas en tres diferentes grupos:

- Aplicaciones colaborativas:
  - Confluence.
  - *Teams*.
  - *Suite* de google.
- Aplicaciones de gestión de proyectos:
  - Jira.
  - Trello.
- Aplicaciones para el ciclo de vida del desarrollo de *software*:
  - Swagger.
  - Postman.
  - Jenkins.
  - Docker.
  - Kubernet.
  - Azure.
  - Azure Monitor para monitoreo de aplicaciones.

2.4.4. Arquitectura de infraestructura base

Las aplicaciones serán desplegadas en la nube. Banco Ecuador cuenta con soporte de Microsoft y además, con licencias activas en la nube de Microsoft Azure. Por lo tanto la infraestructura necesaria está disponible.

3. Arquitectura de Negocio

3.1. Arquitectura actual

La arquitectura actual de negocio de Banco Ecuador está basada en gran porcentaje en *Scrum*, los equipos cuentan con los roles que conforman este marco de trabajo, sin embargo, sus responsabilidades no están definidas al 100% y los entregables no siempre están enfocados en generar valor a los clientes. En la Figura 10 se muestra la arquitectura actual de negocio.

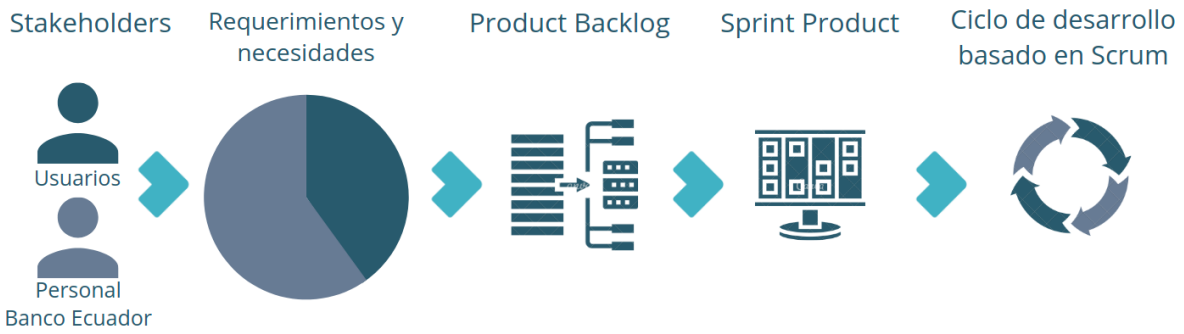


Figura 10. Arquitectura actual de negocio

3.2. Unidad organizativa

Actualmente el área correspondiente a la fábrica de software está conformada por dos grandes grupos:

- Equipo directivo.
  - Vicepresidente de Tecnología.
  - *Chief Digital Officer*.
- Equipo funcional.
  - Líderes de agilidad, experiencia de usuario y arquitectura de software.
  - *Product Owner*.

- *Scrum Master*.
- Equipo *Scrum*.

La Figura 11 muestra la estructura actual de la unidad organizativa del área de tecnología de Banco Ecuador.

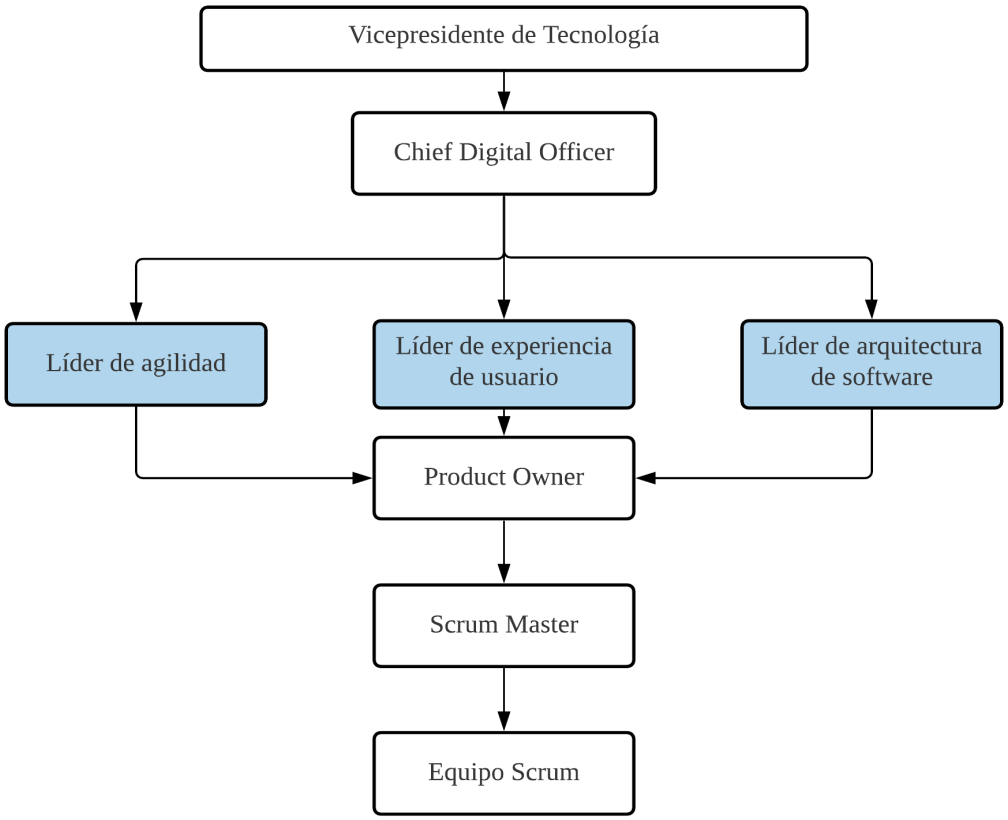


Figura 11. Unidad Organizativa del área de Tecnología

El equipo Scrum está formado por los siguientes roles:

- Arquitecto de software.
- DevOps
- Especialista en experiencia de usuario.
- Especialista de interfaces de usuario.
- Desarrollador *backend*.
- Desarrollador *frontend*.
- Especialista en control de calidad.

3.2.1. Perfiles del equipo de trabajo

***Product Owner*:** Es quien toma las decisiones del cliente. Su mayor responsabilidad es la de generar valor al producto. Además, debe contar con las siguientes características:

- Poder decidir cómo será el resultado final y el orden en el que se van construyendo los sucesivos incrementos.
- Conocer el plan del producto.
- Responsable de fechas y funcionalidades del producto.
- Conocer el entorno del negocio del cliente.
- Conocer el marco de trabajo de *Scrum*.
- Tener conocimiento en desarrollo y administración del *backlog* del producto.

***Scrum Master*:** Es quien hace que se cumplan las reglas del marco de trabajo de *Scrum*, asegurando que estas sean entendidas por toda la empresa y se trabaje conforme a ellas. Es el responsable de:

- Asesorar y formar al equipo para trabajar de forma autoorganizada.
- Revisar y validar el *backlog* del producto.
- Es el moderador en todas las ceremonias.
- Ayuda a resolver impedimentos que puedan entorpecer la ejecución de las tareas.
- Gestión, diseño y mejora continua de las prácticas de *Scrum* en la empresa.

Equipo de desarrollo: Conformado por un grupo de profesionales encargados de realizar el incremento en cada sprint. Cumplen con las siguientes características:

- Todos conocen y comprenden la visión del *Product Owner*.
- Se encargan del desarrollo de los ítems del *backlog* del producto.
- Participan en las decisiones del equipo.
- Comparten el objetivo y la responsabilidad del logro de cada sprint.
- Conocen de *Scrum*.

3.3. Roles y responsabilidades

A continuación, se muestra una tabla con los procesos elegidos para el planteamiento de la arquitectura de negocio objetivo, distribuidos en procesos de CMMI Dev 1.3, *Scrum* y OWASP.

Tabla 20
Roles y responsabilidades bajo los procesos de CMMI Dev 1.3

Procesos	Vicepresi-dencia de Tecnología	Chief Digital Officer	Product Owner	Scrum Master	Equipo de desarrollo
Gestión de configuración		A	R	R	R
Gestión integrada del proyecto		A	R	R	
Monitorización y control del proyecto	A	A	A	A	R
Planificación del proyecto			R	R	
Desarrollo de requisitos			R	A	R
Gestión de requisitos			R	A	R
Solución técnica		A	A		R
Validación			A		R
Verificación			A		R

Tabla 21
Roles y responsabilidades bajo los procesos de Scrum

Procesos	Vicepresi-dencia de Tecnología	Chief Digital Officer	Product Owner	Scrum Master	Equipo de desarrollo
Definición de Rol de dueño del producto			R		
Definición de backlog para un sprint			A	R	R
Asistencia a ceremonias	R	R	R	R	R
Definición de terminado			A	R	R
Definición de product backlog			R		
Manejo de iteraciones			R	R	
Distribución del equipo de trabajo			R		
Metodología de desarrollo			I	I	R
Estándares de codificación			I	I	R
Control de código fuente			R		R
Arquitectura de seguridad		I			R
Control de auditoría	I	I	R	R	R

Tabla 22
Roles y responsabilidades bajo los procesos de OWASP

Procesos	Vicepresi-dencia de Tecnología	*Chief Digital Officer*	*Product Owner*	*Scrum Master*	Equipo de desarrollo
Metodología de desarrollo			I	I	R
Estándares de codificación			I	I	R
Control de código fuente			R		R
Arquitectura de seguridad		I			R
Control de auditoría	I	I	R	R	R

3.4. Análisis de brechas

En la tabla 23 se muestra el análisis de brechas obtenido entre la arquitectura de negocio objetivo detallada en el capítulo de Visionamiento Arquitectónico y la arquitectura actual.

Tabla 23
Análisis de brechas

Brecha	Procesos asociados	Problema
BN1	● Planificación del proyecto. ● Gestión de Requisitos. ● Definición de backlog del producto.	Los requerimientos del proyecto son priorizados en mayor parte enfocados en las necesidades del grupo de stakeholders pertenecientes a Banco Ecuador, dejando de lado las necesidades de clientes de los canales digitales del portafolio de aplicaciones.
BN2	● Definición de backlog para un sprint. ● Definición de terminado.	Las historias de usuario descritas en el sprint backlog no cuentan con el nivel de detalle recomendado por Scrum.
BN3	● Validación. ● Verificación.	El porcentaje de automatización de pruebas a nivel de Banco Ecuador es inferior al 30%.
BN4	● Control de auditoría.	Las herramientas de monitoreo no centralizan todos los componentes que conforman los productos de Banco Ecuador.
BN5	● Monitorización y control del proyecto. ● Control de auditoría.	No se tiene un estándar de manejo de logs en todas las aplicaciones del portafolio de aplicaciones de Banco Ecuador.
BN6	● Verificación.	No se tiene implementado un proceso para la revisión por pares del código fuente.
BN7	● Verificación.	No se tiene un estándar de procedimientos y criterios de verificación.

3.4.1. Iniciativas para cerrar brechas

Una vez analizada la arquitectura objetivo y la arquitectura actual de negocio de Banco Ecuador se hizo el análisis de brechas encontrando algunos problemas, los cuales podrán ser cerrados ejecutando la siguiente lista de iniciativas:

- Priorizar el *backlog* del producto con base en las necesidades de usuarios finales. Para ejecutar esta iniciativa *Scrum* recomienda redactar bien las historias de usuario, refinarlas y armar un mapa de historias. Con este mapa se podrá tener una visibilidad amplia de las principales características del producto y de las dependencias entre historias.
- El *Scrum Master* debe tener más participación en cuanto a la autogestión del equipo para que todos respeten, asistan y participen de manera activa en las ceremonias de este marco de trabajo. Así, la ceremonia de refinamiento de historias de usuario ayudará de la siguiente manera:
    - Mejorar la redacción.
    - Identificar los impedimentos antes de priorizar una historia de usuario.
    - Encontrar todos los componentes involucrados en el desarrollo de una historia de usuario.
    - Identificar dependencias de otras historias de usuario.
- Basados en la cultura DevOps, lo primero que se debe hacer es crear una estrategia para implementar las pruebas automáticas dentro del flujo de despliegue. En la Figura 12 se muestra el flujo que involucra el despliegue de código probado.

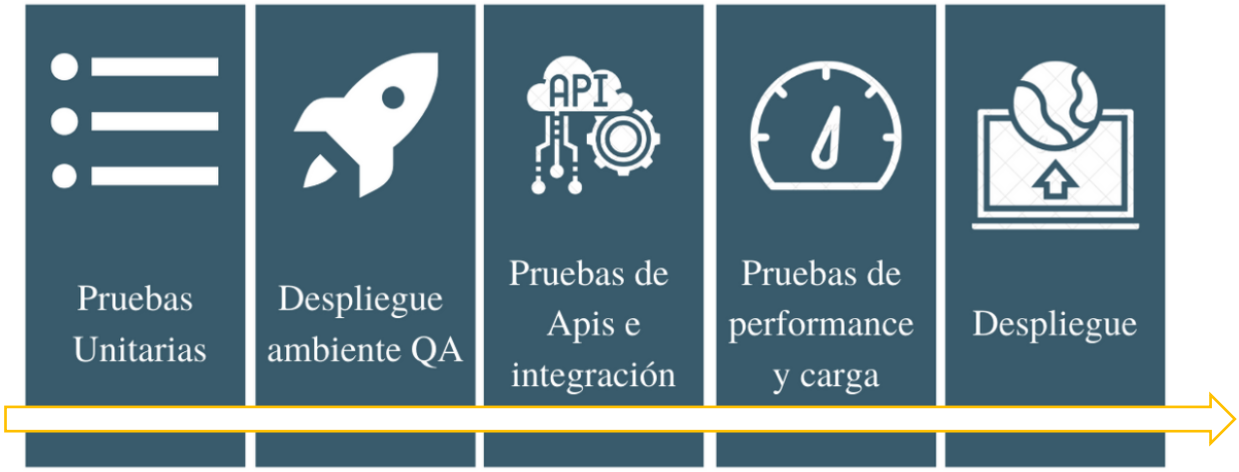


Figura 12. Flujo de despliegue de código

- Estandarizar la escritura de logs considerando las siguientes características:
  - Niveles de logs.
  - Almacenar los logs de los diferentes productos en la misma herramienta.
  - Cifrar la información sensible de los usuarios.
  - Separar los logs de Auditoría, Depuración, Errores, Logs en diferentes ficheros. La guía de desarrollo de OWASP sugiere la siguiente arquitectura descrita en la Figura 13.

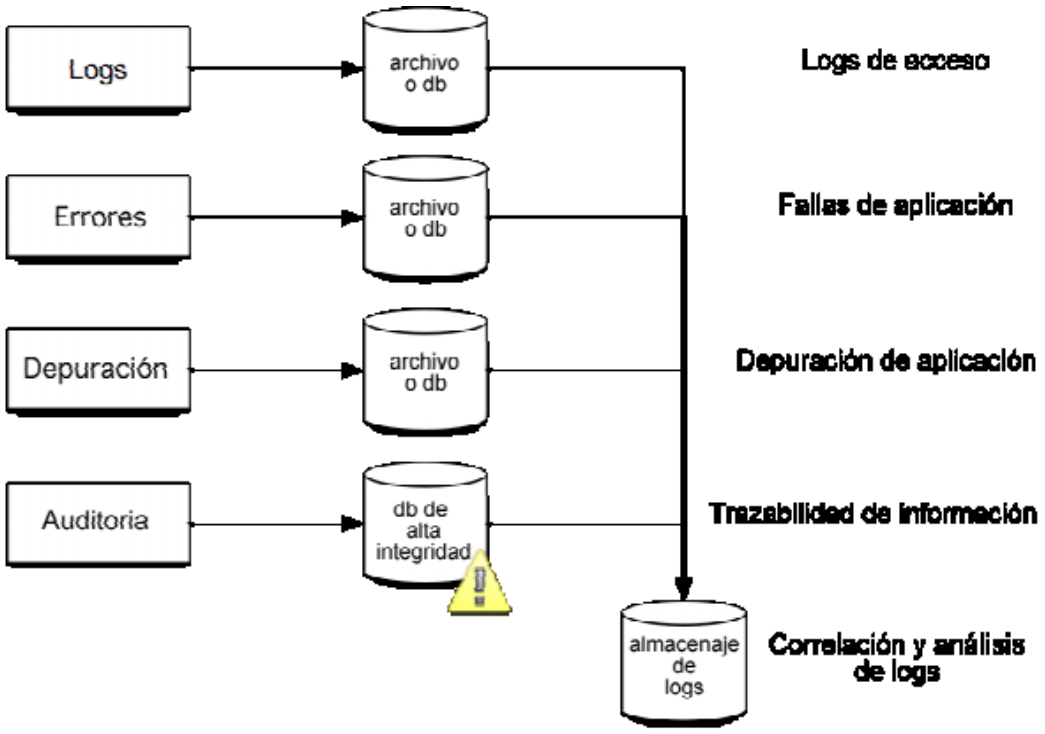


Figura 13. Arquitectura de logs. (Kang et al. 2005. The Open Web Application Security)

- Establecer un proceso para la revisión por pares del código fuente, para lo cual basados en CMMI Dev 1.3 se debe empezar identificando al personal que formará parte de la revisión entre pares de los productos de trabajo. Además, se deben generar los siguientes componentes para la ejecución exitosa de la revisión.
    - Listas de comprobación.
    - Criterios de revisión.
    - Calendario de la revisión.
- Establecer criterios de verificación para asegurar que los productos de trabajo cumplan sus requisitos.

4. Arquitectura de Sistemas / Información

4.1. Arquitectura de Aplicaciones actual

Actualmente Banco Ecuador cuenta con un catálogo de aplicaciones extenso. Cada aplicación cumple con sus funciones, pero existe poca relación entre ellas.

En la figura 14 se muestra una aproximación a nivel general de las categorías bajo las cuales están agrupadas las diferentes aplicaciones. Por motivos de seguridad no se detallan nombres de las herramientas con las que cuentan.

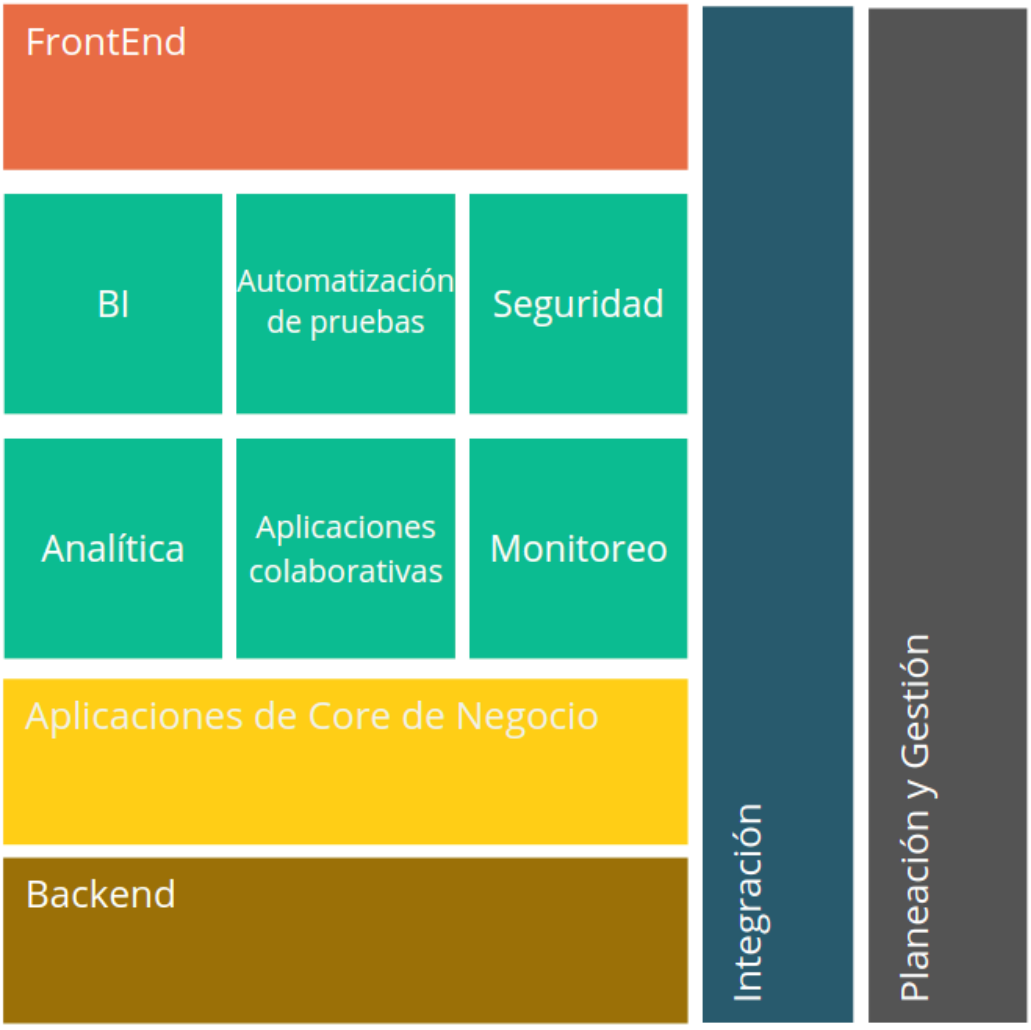


Figura 14. Arquitectura de aplicaciones actual

4.2. Análisis de brechas

En relación con el objetivo de fortalecer la fábrica de *software inhouse* de Banco Ecuador, se realizó el análisis de brechas enfocados en las siguientes categorías:

- Integración.
- Seguridad.
- Automatización de pruebas.
- Analítica de datos.
- Monitoreo de aplicaciones.
- Aplicaciones colaborativas.

La tabla 24 explica las dimensiones aplicadas sobre las categorías con el fin de encontrar el nivel de madurez y capacidad de cada una.

Tabla 24
Rúbrica para análisis de brechas

Dimensión	Descripción
Vigencia tecnológica	Nivel de incorporación de componentes tecnológicos modernos y aceptados por el mercado de soluciones de software y cumplimiento de estándares internacionales para usabilidad, robustez, portabilidad
Interoperabilidad	Capacidad de la aplicación para intercambiar información de manera automática con otras aplicaciones
Confiabilidad	Esquema de garantía que ofrece la aplicación para dar resultados precisos, un buen desempeño y consistencia de los datos
Arquitectura	Esquema de administración del conocimiento asociado a la aplicación, modelos arquitectónicos, diseños, guías de uso, administración actualizados.
Soporte	Esquema de apoyo/soporte con el que cuenta la aplicación para atender inquietudes de los usuarios, incidentes y requerimientos de operación
Seguridad	Nivel de protección con que cuenta la aplicación para contrarrestar los riesgos de vulnerabilidad que afecten la disponibilidad, integridad y confidencialidad de la información y los servicios tecnológicos que asiste
Flexibilidad/Mantenimiento	Capacidad de la aplicación para adaptarla a nuevos requerimientos en tiempos y costos razonables a los alcances del cambio, con la asistencia a versionamiento, documentación y marco de trabajo para los cambios
Gobernabilidad	Esquema de supervisión ejecutiva que se le da a la aplicación y aseguramiento de su alineamiento estratégico con las prioridades del negocio

En efecto de este análisis se puede observar cuales son las brechas existentes en las categorías mencionadas anteriormente. En la figura 15 se muestra el estado actual, el esperado y el referente en cuanto a madurez y capacidad en cada caso. Los análisis individuales se encuentran al final del archivo en el Anexo 2.

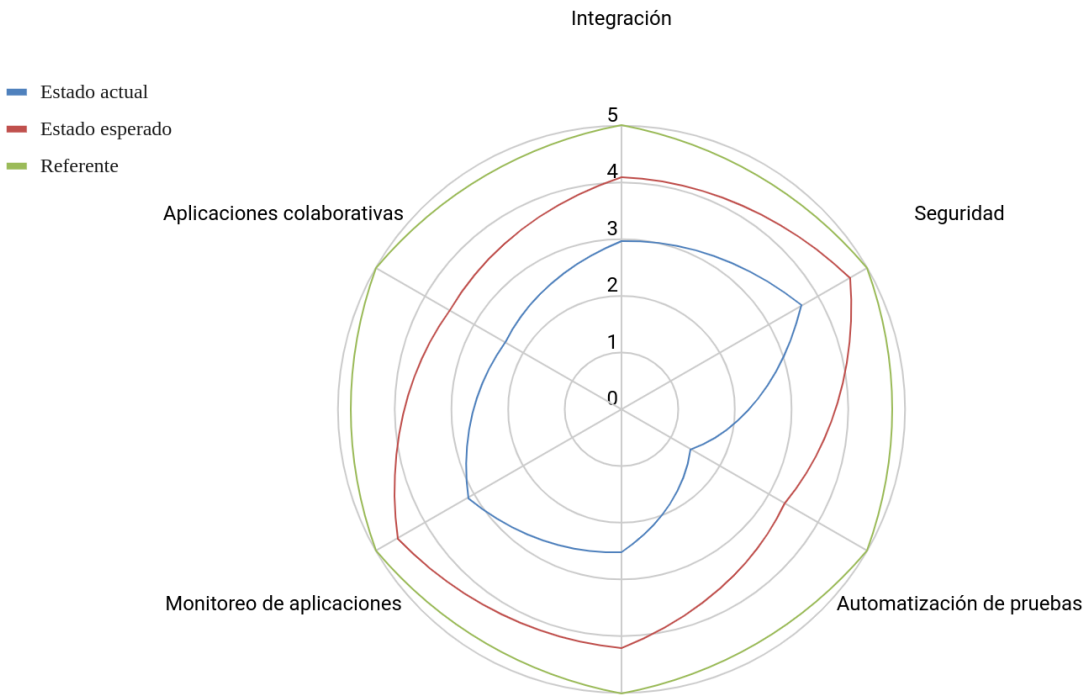


Figura 15. Resultado estado de madurez y capacidad

Como resultado de este ejercicio se encontraron los siguientes problemas asociados a sus categorías detallados en las siguientes tablas.

Tabla 25
Análisis de brechas de Integración

Brecha	Categoría asociada	Problema
BA1	Integración	La característica de unidireccionalidad en las herramientas de integración impide que otras herramientas puedan recopilar información para monitoreo de despliegue de las aplicaciones de Banco Ecuador.
BA2	Integración	El mantenimiento es provisto por el mismo equipo técnico del Banco Ecuador, no se cuenta con un proveedor externo que dé soporte a problemas debido al tipo de licencia del producto.

Tabla 26
Análisis de brechas de Seguridad

Brecha	Categoría asociada	Problema
BA3	Seguridad	Las herramientas de seguridad no están integradas al monitoreo de aplicaciones lo que dificulta la presentación de métricas en torno a este tema.
BA4	Seguridad	El desarrollo de nuevas características de seguridad implica un alto costo en tiempo y recursos debido a su poca flexibilidad.

Tabla 27
Automatización de pruebas

Brecha	Categoría asociada	Problema
BA5	Automatización de pruebas	No se cuenta con una herramienta que ayude a automatizar las pruebas. Todos los intentos de automatización con los que cuenta Banco Ecuador no están estandarizados ni siguen una documentación específica.
BA6	Automatización de pruebas	Debido a las limitantes que se tienen en las pruebas automáticas no existe interoperabilidad con otras aplicaciones y los datos obtenidos no son compartidos ni generan reportes legibles para el grupo de stakeholders no técnicos.
BA7	Automatización de pruebas	Existen casos que no han sido cubiertos de manera exitosa en la ejecución de pruebas automatizadas, lo que ocasionó problemas en versiones liberadas en producción.
BA8	Automatización de pruebas	No se tiene documentación que sirva como guía para la implementación de pruebas.
BA9	Automatización de pruebas	Las pruebas automatizadas escritas en código no cumplen con estándares de seguridad.

Tabla 28
Analítica de datos

Brecha	Categoría asociada	Problema
BA10	Analítica de datos	Algunas métricas e indicadores son realizados en excel.
BA11	Analítica de datos	No se tiene un correcto mantenimiento de los modelos establecidos para la generación de métricas. En muchos de los casos, cada cambio en el modelo implica un nuevo desarrollo debido a la falta de documentación de las implementaciones actuales.

Tabla 29
Monitoreo de aplicaciones

Brecha	Categoría asociada	Problema
BA12	Monitoreo de aplicaciones	Las herramientas utilizadas para el monitoreo están aisladas entre sí. No se comparte información y existen demasiados tableros con información repetida.

Tabla 30
Aplicaciones colaborativas

Brecha	Categoría asociada	Problema
BA13	Aplicaciones colaborativas	Se tienen diferentes aplicaciones que cumplen la misma función, sin embargo no se explota al 100% sus capacidades.

4.2.1. Iniciativas para cerrar brechas

Una vez hecho el análisis de brechas, se plantean las siguientes iniciativas con el fin de cerrar los problemas detallados anteriormente:

- Se implementará la herramienta Jenkins para configuración de flujo de despliegue e integración continua. Se elige esta herramienta debido a las siguientes características:
    - Curva de aprendizaje bajo.
    - Amplia comunidad para brindar soporte.
    - Panel de reportes del estado de los despliegues.
    - Interoperabilidad.
    - Herramienta de software libre.
- Se migrarán el 50% de aplicaciones en arquitecturas *on-premise* hacia la nube. El principal apalancador en esta iniciativa será Docker.
  Docker es una tecnología que utiliza el kernel y las funciones de Linux para sesgar los procesos, de modo que puedan ejecutarse de manera independiente. Las aplicaciones serán migradas hacia estos contenedores con el fin de darles independencia, ejecutar varios procesos y aplicaciones por separado y brindarles seguridad.
- Implementar herramientas de seguridad líderes en el mercado. Un ejemplo de esto es Akamai, esta plataforma ofrece:

    - Investigación de amenazas: Obtención de datos en tiempo real, generación de informes de seguridad, investigaciones recientes sobre ciberseguridad.
    - Investigación del rendimiento: Brinda información sobre rendimiento y entrega, ofrece mejora en la experiencia online gracias a los datos recopilados desde su plataforma.
- Se plantean un grupo de herramientas que pueden ser utilizadas para cumplir con estándares de seguridad en cuanto a la ejecución y creación de pruebas automáticas. Estas herramientas se pueden ver en la Figura 3.
- Es necesario crear guiones documentados para la ejecución de pruebas de calidad, debido al gran ecosistema de aplicaciones que se tiene en Banco Ecuador. Además, de esta manera se tiene una vista panorámica de los casos de prueba que han sido cubiertos.
- Se deben aplicar los mismos estándares de código seguro, de calidad y de escritura de código en las pruebas que involucran desarrollos.
- Los reportes e indicadores no pueden seguir desarrollados en Excel, se recomienda utilizar las siguientes herramientas:
    - Google Analytics para recopilación y analítica de datos a nivel de experiencia de usuario y patrones de comportamiento de usuarios.
    - PowerBI para seguimiento de métricas asociadas a la estrategia empresarial.
- Depurar tableros de monitoreo actuales para evitar duplicidad de información.
- Migrar a una herramienta única que ayude a centralizar la información recopilada y evite el excesivo trabajo al tener que revisar tableros de monitoreo de diferentes herramientas. Esto ayudará a evitar errores humanos por mala interpretación de diferentes alertas sobre un mismo problema.

4.2.2. Catálogo de aplicaciones objetivo

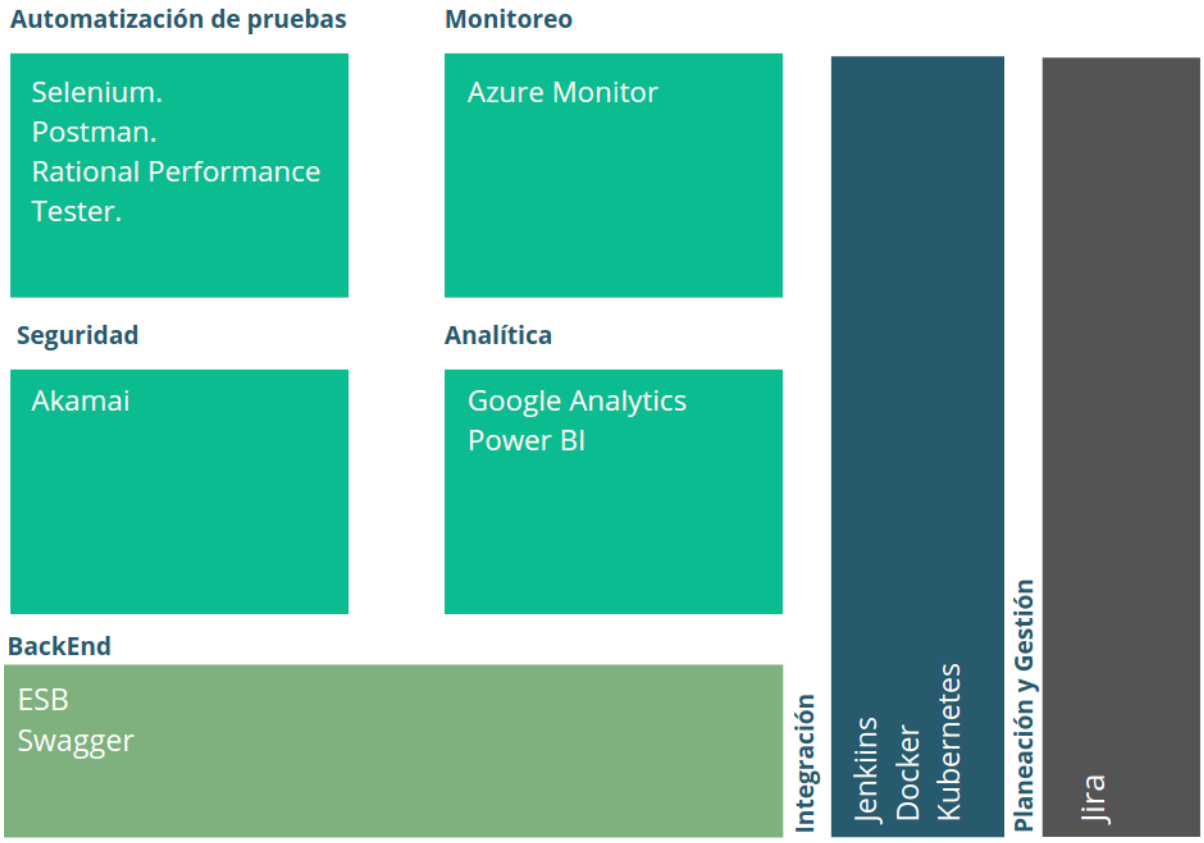


Figura 16. Catálogo de aplicaciones objetivo

4.3. Arquitectura de Datos actual

Respecto a la arquitectura de datos manejada por Banco Ecuador se tiene una estructura compleja y con una interoperabilidad prácticamente nula. Cada documento generado o información obtenida de la ejecución de herramientas tecnológicas internas, maneja su propio espacio de destino de almacenamiento. En la figura 17 se muestra una descripción de los diversos repositorios de datos que manejan y la información que almacena en cada caso.

Figura 17. Arquitectura de datos actual

4.4. Análisis de brechas

Debido a que la información que se maneja está distribuida en diferentes repositorios y sin interacción entre ellos, se requiere establecer un repositorio centralizado que cumpla con las características descritas en la arquitectura de datos objetivo desarrollada en la fase de visión.

Teniendo en cuenta que la información está repetida en algunos repositorios de datos, es necesario que se analice si la información actual es consistente y correcta. Además, es necesario destacar que el repositorio de código se encuentra bien organizado y almacena lo debido.

4.4.1. Iniciativas para cerrar brechas

Se establecerá un repositorio único de la mano de Confluence, una herramienta de Atlassian. Para una correcta implementación se debe recopilar toda la información actual y la nueva información que se generará al implementar las iniciativas de la arquitectura de aplicaciones que involucren la creación de nuevos documentos, se la clasificará y se asignará roles y responsables para cada caso.

La tabla 31 muestra una matriz de documentación versus responsables que deberá tener la documentación actual con base en la arquitectura de datos actual de Banco Ecuador.

Tabla 31
Documentación versus responsables

Documentación	Vicepresidencia de Tecnología	Chief Digital Officer	Product Owner	Scrum Master	Equipo de desarrollo
Métricas/indicadores	X	X	X	X	
Presentaciones	X	X	X	X	
Minutas			X	X	X
Historias de usuario			X	X	X
Diagramas de arquitectura de aplicaciones		X	X	X	X
Documentación de control de calidad				X	X
Flujos de aplicaciones		X		X	X
Artefactos para desarrollo					X
Requerimientos de usuario			X	X	X

5. Arquitectura Tecnológica

Banco Ecuador cuenta con una infraestructura tecnológica grande, distribuida en las diferentes tribus que conforman la fábrica de *software in house*. Se tomará una tribu dedicada al producto de banca *web* como referente.

Los equipos con los que cuenta en ambientes de desarrollo, pruebas, producción y el equipo *Scrum* se detallan a continuación.

5.1. Arquitectura de Tecnología actual

Los equipos con los que cuenta Banco Ecuador en ambientes no productivos están detallados en las siguientes tablas.

Tabla 32
Arquitectura de tecnología actual de ambiente de Desarrollo

Servidor	Número de equipos	Características	Detalle de uso
SonarQube	1	Sistema Operativo: Linux RAM: 4GB Disco duro: 128GB Procesadores: 2	Análisis de calidad en código
Identidad	1	Sistema Operativo: Linux RAM: 4GB Disco duro: 128GB Procesadores: 2	Provee autorización de manera centralizada para el inicio de sesión
Balanceador	1	Sistema Operativo: Linux RAM: 6GB Disco duro: 128GB Procesadores: 2	Distribuye la carga hacia servidores de despliegue de *frontend* y *backend*
Aplicaciones	2	Sistema Operativo: Linux RAM: 4GB Disco duro: 128GB Procesadores: 2	Servidor de aplicaciones para componentes de *FrontEnd*
Aplicaciones	2	Sistema Operativo: Linux RAM: 4GB Disco duro: 128GB Procesadores: 2	Servidor de aplicaciones para componentes de *BackEnd*

Tabla 33
Arquitectura de tecnología actual de ambiente de Pruebas

Servidor	Número de equipos	Características	Detalle de uso
Identidad	1	Sistema Operativo: Linux RAM: 4GB Disco duro: 128GB Procesadores: 2	Provee autorización de manera centralizada para el inicio de sesión
Balanceador	1	Sistema Operativo: Linux RAM: 6GB Disco duro: 128GB Procesadores: 2	Distribuye la carga hacia servidores de despliegue de *frontend* y *backend*
Aplicaciones	2	Sistema Operativo: Linux RAM: 4GB Disco duro: 128GB Procesadores: 2	Servidor de aplicaciones para componentes de *FrontEnd*
Aplicaciones	2	Sistema Operativo: Linux RAM: 4GB Disco duro: 128GB Procesadores: 2	Servidor de aplicaciones para componentes de *BackEnd*

Tabla 34
Arquitectura de tecnología actual de ambiente de Producción

Servidor	Número de equipos	Características	Detalle de uso
Identidad	2	Sistema Operativo: Linux RAM: 16GB Disco duro: 128GB Procesadores: 4	Provee autorización de manera centralizada para el inicio de sesión
Balanceador	2	Sistema Operativo: Linux RAM: 16GB Disco duro: 128GB Procesadores: 4	Distribuye la carga hacia servidores de despliegue de *frontend* y *backend*
Aplicaciones	3	Sistema Operativo: Linux RAM: 16GB Disco duro: 128GB Procesadores: 4	Servidor de aplicaciones para componentes de *FrontEnd*
Aplicaciones	3	Sistema Operativo: Linux RAM: 16GB Disco duro: 128GB Procesadores: 4	Servidor de aplicaciones para componentes de *BackEnd*
DataPower	1	Sistema Operativo: Linux RAM: 8GB Disco duro: 128GB Procesadores: 4	Provee seguridad e integración en una única pasarela multicanal

Tabla 35
Arquitectura de tecnología actual del equipo *Scrum*

Servidor	Número de equipos	Características	Detalle de uso
Dell		Sistema Operativo: Windows RAM: 16GB Disco duro SSD: 1TB Procesador: Intel i7 8va Generación	Equipo destinado para gestón
Dell		Sistema Operativo: Windows RAM: 16GB Disco duro SSD: 1TB Procesador: Intel i7 8va Generación	Equipo destinado para gestón
Dell		Sistema Operativo: Linux RAM: 16GB Disco duro SSD: 1TB Procesador: Intel i7 8va Generación	Equipo destinado para desarrollo de *software*
MacBook Pro		Sistema Operativo: macOS RAM: 8GB Disco duro SSD: 512GB Procesador: Intel i5	Equipo destinado para diseño de experiencia de usuario y diseño de interfaces

5.2. Análisis de brechas

En este caso no es conveniente realizar un análisis de brechas debido a que el fortalecimiento de la fábrica de *software in house* para Banco Ecuador viene de la mano de la migración de la arquitectura *on-premise* hacia la nube.

5.3. Arquitectura de infraestructura objetivo

Banco Ecuador necesita una infraestructura en la nube que cubra los siguientes frentes:

- Seguridad.
- Escalabilidad.
- Descentralización de sus aplicaciones.
- Administración de infraestructura.
- Monitoreo de aplicaciones.

Con base en estas características se recomienda que la migración se la haga hacia la nube Azure de Microsoft. Esta plataforma cubre todos los frentes mencionados de la siguiente manera.

5.3.1. Seguridad

Recomienda seguir la guía de seguridad de DevOps aprovechando las instrucciones dadas y la automatización para proteger las aplicaciones en la nube, en lugar de empezar de cero. Azure cuenta con un kit de herramientas para proteger DevOps orientado a (Microsoft, 2021):

- Escritura de código seguro ayudado de una herramienta llamada Security IntelliSense.

- Comprobación de seguridad en todo el flujo de integración y despliegue continuo.
- Rastreo en el manejo de la seguridad en ambientes productivos.
- Supervisión de la seguridad en todas las etapas de DevOps.
- Presentación de paneles gráficos con los riesgos encontrados en la nube en toda la infraestructura administrada por Azure.

Además, ofrece servicios propios de la nube para evitar implementaciones personalizadas que pueden poner en riesgo las siguientes funciones ante un posible impacto en la seguridad:

- Identidad.
- Protección de datos.
- Administración de claves.
- Configuraciones de aplicación.

5.3.2. Escalabilidad

Maneja procesos de escalado automático para la asignación dinámica de recursos para satisfacer los requisitos de rendimiento. Este proceso ayuda a mantener los niveles de rendimiento deseados y cumplir con los acuerdos de nivel de servicio.

Azure reduce la necesidad de personal para supervisar continuamente el rendimiento de los sistemas alojados y la toma de decisiones al momento de incorporar o eliminar recursos.

Esta nube permite escalar verticalmente y horizontalmente. Debido al nivel de servicio que debe ofrecer Banco Ecuador se detallan la forma en la que se va a escalar (Microsoft, 2020).

- **Escalado vertical:** Se utilizará para mover aplicaciones a máquinas virtuales más grandes. Este tipo de escalado implica indisponibilidad

temporal del sistema, lo cual está permitido en horarios de poca transaccionalidad.

- **Escalado horizontal:** Se utilizará en el momento en que se requiera agregar o quitar instancias de un recurso, en este caso, las aplicaciones no dejarán de ejecutarse.

5.3.3. Descentralización

Dentro de la documentación de Azure en torno a la descentralización, se trata el caso específico de un banco, en donde se propone utilizar la siguiente arquitectura explicada en la figura 18.

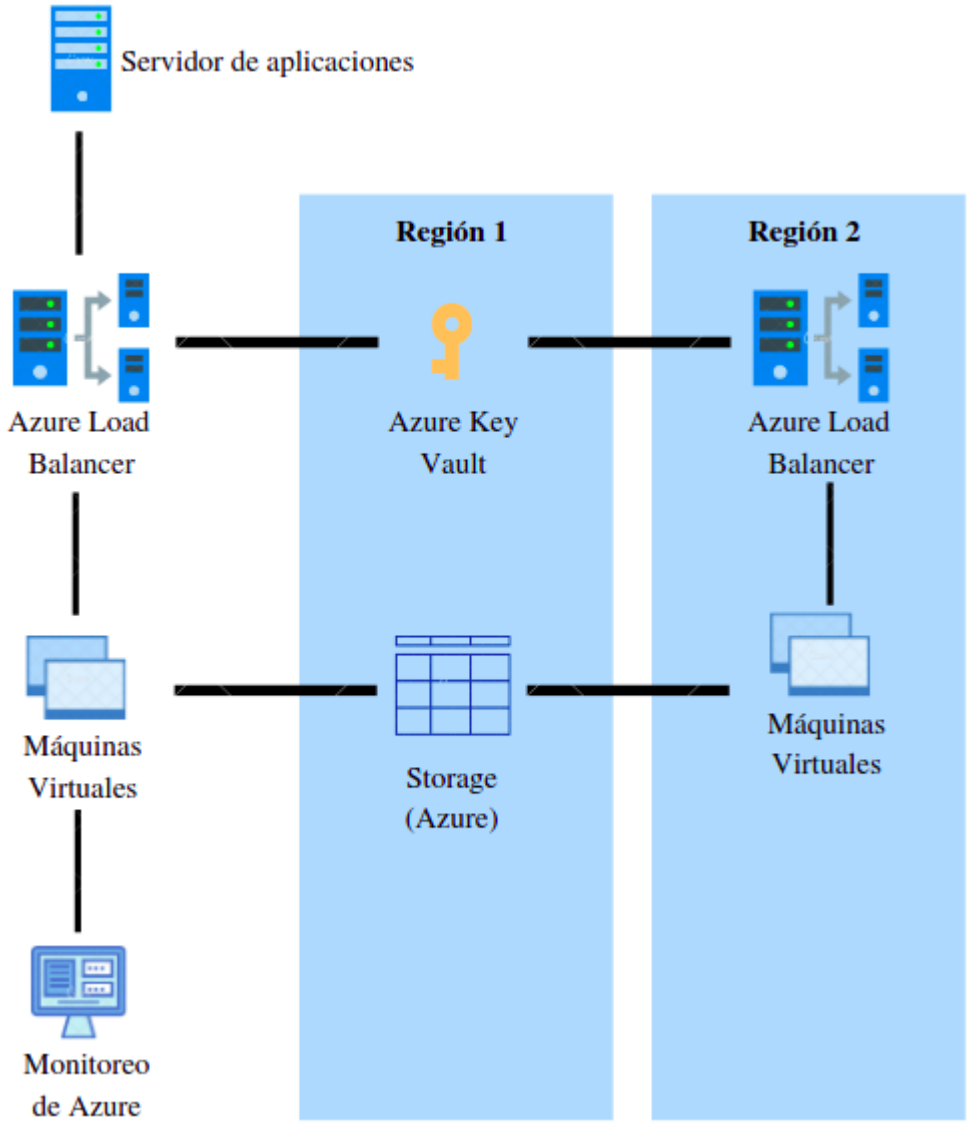


Figura 18. Arquitectura descentralizada. Adaptado de (Microsoft, 2021)

Esta arquitectura muestra los componentes necesarios de backend según Azure, para crear una red de cadena de bloques empresarial privada, escalable, segura y supervisada. Los componentes que intervienen y su funcionalidad se detalla en la tabla 36.

Tabla 36
Componentes

Componente	Características
Máquinas virtuales	Proporcionan capacidad de proceso bajo demanda.
Azure *Key Vault*	Almacenamiento seguro para claves privadas.
Azure *Load Balancer*	Distribuye las solicitudes de RPC, emparejamiento y aplicaciones descentralizadas de gobernanza.
Azure *Storage*	Almacenamiento de información de red persistente y coordina la concesión.
Operations Management Suite	Proporciona una visión de los nodos disponibles y las transacciones por minuto.

5.3.4. Administración de infraestructura

Se utilizará la infraestructura como servicio (IaaS) de Azure para habilitar el escalamiento vertical en función de la demanda. De esta manera se invierte el dinero en infraestructura más eficientemente, pagando por lo que se usa, es decir, cada componente que conformará la infraestructura es tratado como un componente de servicio individual que será alquilado durante el tiempo que se lo necesite.

Los escenarios con los que contará Banco Ecuador son los siguientes:

- **Desarrollo y pruebas:** Configurar y dar de baja entornos de desarrollo y pruebas para ejecuciones y pruebas de calidad de desarrollos internos.
- **Hospedaje de sitios web:** Debido al bajo costo en la nube comparado con un hospedaje web tradicional.
- **Almacenamiento, copias de seguridad y recuperación:** Manejo adecuado de almacenamiento de información y crecimiento bajo demanda. Simplifica la planificación y administración de los sistemas de copia de seguridad y recuperación.
- **Aplicaciones web:** Proporciona las herramientas necesarias para despliegues de aplicaciones web, como almacenamiento, servidores web y de aplicación y recursos de red.
- **Monitoreo en tiempo real:** Tiene herramientas que recopilan los logs de las aplicaciones desplegadas en la nube y lo transforman a métricas configurables, mostradas en paneles gráficos.
- **Análisis de macrodatos:** Enormes conjuntos de datos que contienen patrones, tendencias y asociaciones con fuerte potencial.

5.3.5. Monitoreo de aplicaciones

Azure ofrece un gran número de herramientas y conectores de datos para el monitoreo de aplicaciones. Para Banco Ecuador se vio conveniente utilizar la herramienta Azure Monitor que se ajusta perfectamente a sus necesidades:

Azure Monitor es un servicio configurable por desarrolladores o DevOps profesionales usado para monitorear aplicaciones en producción. Detecta automáticamente problemas de rendimiento y, además, incluye herramientas de análisis para ayudar a diagnosticar problemas y ayudar a entender la manera en la que los usuarios interactúan con las aplicaciones.

6. Oportunidades y soluciones

El presente capítulo consolida y conceptualiza todas las iniciativas realizadas en la fase previa, con el fin de dar solución a los problemas encontrados en Banco Ecuador.

6.1. Arquitectura de negocio

Actualmente el *framework* utilizado por Banco Ecuador para el desarrollo de software de manera ágil es *Scrum*, pero como se puede ver en la arquitectura de negocio, existen algunos problemas en torno a su correcta aplicación. Los dos aspectos en los que se debe poner atención son:

- Establecer una técnica para priorizar los ítems que conforman el *Product Backlog*.
- Fortalecer la ceremonia de refinamiento de historias de usuario.

Con base en esta problemática se propone una iniciativa que ayudará con la priorización y el refinamiento de las historias de usuario, la cual se detalla a continuación.

6.1.1. Priorizar los ítems del *Product Backlog* basados en la técnica MoSCoW

Antes de abordar el desarrollo de la técnica a adoptar para la priorización de los ítems del *Product Backlog*, es necesario entender el significado de este elemento dentro de *Scrum*.

El *product backlog* es una lista emergente y ordenada de aquello necesario para mejorar el producto. Los ítems que lo conforman son los candidatos a ser tomados por el equipo *Scrum* para su futura implementación.

MoSCoW es la técnica que se utilizará para priorizar los ítems del *product backlog*, esta técnica es nombrada así debido a la combinación de las primeras letras de: *Must Have* (debe tener), *Should Have* (debe tener), *Could Have* (podría tener) y *Would not Have* (no tendrá en esta ocasión). Este conjunto de categorías está definido de la siguiente manera (IIBA, 2005):

M (*Must have*): Requerimientos que deben ser implementados en la versión final del producto. Con esto, el producto puede ser considerado un éxito.

S (*Should have*): Requerimientos de prioridad alta. Deben formar parte del producto final, pero si por cualquier motivo justificado no debe estar desarrollado podría ser prescindible.

C (*Could have*): Son requerimientos deseables, pero no imprescindibles. Puede formar parte del producto final, siempre que existan recursos económicos y recursos humanos suficientes.

W (*Would not have*): Requerimientos que están descartados por el momento, pero en el futuro podrán ser considerados y pasarán a formar parte de una de las categorías anteriores.

Esta técnica ayuda al equipo *Scrum* a tener una visión más clara del por qué se está trabajando en el desarrollo de ciertas características. Además, ayuda a desarrollar funcionalidades que ofrecen mayor valor al negocio.

Los procesos que se llevarán a cabo son:

- Actividades por realizar.
- Formato.
- Responsables.

6.1.1.1. Actividades por realizar

- Identificar los *stakeholders*, con la finalidad de que solo estos actores involucrados tengan poder de decisión sobre las características del producto. Se recomienda incluir también al equipo responsable del desarrollo.
- Definir un actor principal que resuelva desacuerdos entre los participantes y ayude a llegar a un consenso. Se puede utilizar votaciones o definir el valor de negocio de las características analizadas con el objetivo de seleccionar las que aporten más valor al negocio.
- Distribuir los recursos económicos y humanos entre las iniciativas más críticas y de mayor aporte al negocio. En esta actividad se dejará de lado los ítems del *product backlog* que hayan sido clasificados como *"would not have"*.
- Preparar una lista de todas las características y requisitos del proyecto para tener una visión global de lo esperado del producto. Esta lista debe ser de conocimiento de toda la organización para transparentar el camino que se llevará a cabo hasta tener un producto final.
- Empezar a priorizar basado en las categorías que conforman esta técnica. En la tabla 37 se muestra una recomendación de preguntas que se pueden hacer para categorizar cada ítem del *product backlog* (Miranda, 2011).

Tabla 37
Preguntas recomendadas

Categoría	Preguntas
Must have	1. ¿Qué pasaría si lanzamos el producto sin esa característica? 2. ¿Hay alguna forma más simple de completar esta tarea? 3. ¿Funcionará la aplicación sin esta característica?
Should have	1. ¿Cuánto valor añade esta característica al proyecto? 2. ¿Se puede desarrollar esta característica en futuros *sprints*?
Could have	1. ¿Esta característica es necesaria para la función básica del proyecto? 2. ¿Cuánto impacto tendrá esto en el proyecto si se deja de lado?
Would not have	1. ¿Hay alguna posibilidad de que esta característica tenga prioridad en el futuro?

6.1.1.2. Formato

Una vez priorizados los ítems del *product backlog*, se debe transformar a historias de usuario basados en la siguiente plantilla recomendada por Atlassian (Attlasian Agile Coach, 2021).

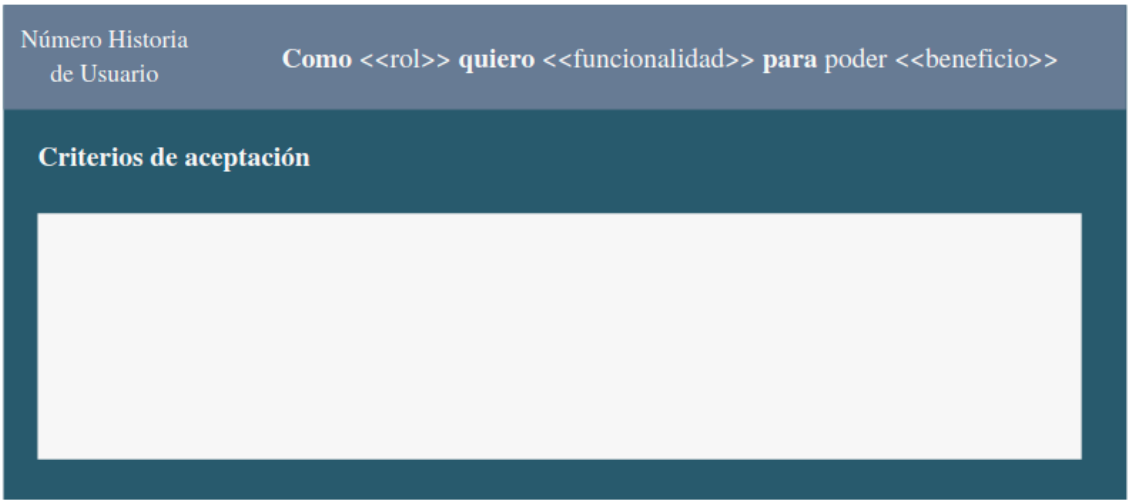


Figura 19. Plantilla historia de usuario. Tomado de (Atlassian Agile Coach, 2021)

Esta estructura funciona de la siguiente manera:

- Número de historia de usuario, es un identificador único.
- Como <<perfil>>, se refiere al perfil para el cual se desarrolla una funcionalidad. No solo un rol, sino entender y conocer a la persona que va a usar la funcionalidad descrita en la historia de usuario.
- Quiero <<funcionalidad>>, aquí se detalla en lenguaje natural la funcionalidad de la historia de usuario. No debe tener detalle de implementación.
- Para poder <<beneficio>>, aquí se detalla el beneficio de la implementación de esta historia de usuario. Qué problema es el que se está resolviendo.
- Criterios de aceptación, son los escenarios mínimos requeridos para que la implementación de la historia de usuario sea exitosa. Si no se cumplen con estos criterios, la historia de usuario no puede ser etiquetada como finalizada.

6.1.1.3. Responsables

Los responsables y sus tareas se detallan en la tabla 38.

Tabla 38
Responsables y tareas

Responsable	Tareas
Stakeholders	Resolver dudas de negocio. Exponer necesidades. Ayudar en la descripción de los criterios de aceptación.
Equipo de desarrollo	Resolver dudas tecnológicas. Ayudar en la descripción de los criterios de aceptación.
Product Owner	Es el responsable de direccionar el producto y ordenar los ítems del *product backlog* con base en las aportaciones de los *stakeholders* y el equipo de desarrollo.

Dentro de la arquitectura de negocio también se evidenciaron problemas en cuanto a la calidad del código, para lo cual se debe poner especial atención en la automatización de pruebas y en el establecimiento del proceso para la revisión por pares como lo recomienda CMMI Dev 1.3.

Las iniciativas que se propone para resolver estos problemas son las siguientes:

- Estrategia para la implementación de pruebas automáticas basados en la pirámide de Cohn, con el fin de hacer regresiones de manera automática evitando posibles errores debido a nuevas funcionalidades.
- Establecer un proceso de revisión por pares basados en CMMI Dev 1.3

6.1.2. Estrategia para la implementación de pruebas automáticas basados en la pirámide de Cohn

Se utilizará la pirámide de pruebas de Mike Cohn como referente para desarrollar la estrategia de implementación de pruebas automáticas. (Cohn, 2009) establece varios niveles de pruebas y el grado de automatización que debe tener cada uno. La figura 20 muestra la pirámide.

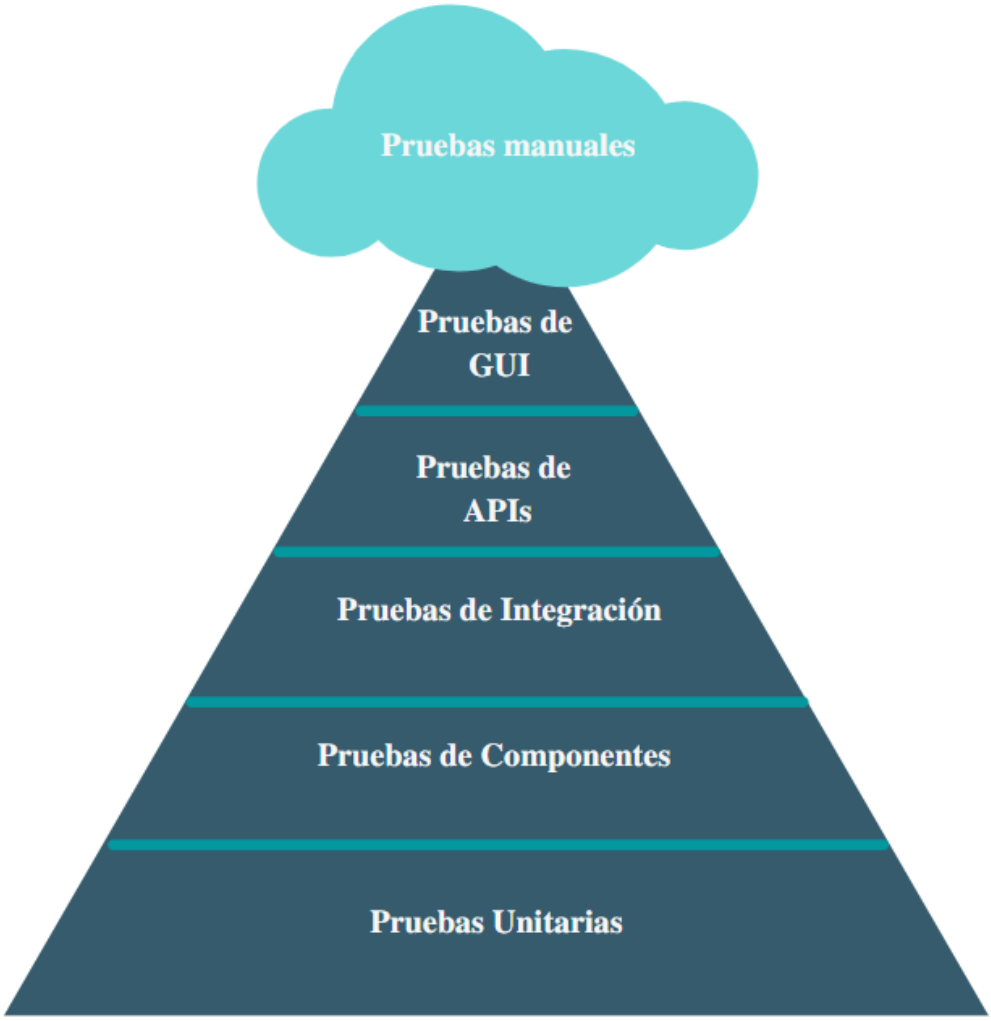


Figura 20. Pirámide de Cohn

En donde plantea lo siguiente:

Pruebas unitarias: Al ser la base de la pirámide, se recomienda automatizar la mayor cantidad de pruebas unitarias de manera que si una funcionalidad falla en este punto, pueda ser tempranamente detectada y solucionada, evitando así problemas en los siguientes niveles.

Pruebas de componentes, integración y de APIs: Son los siguientes candidatos a ser automatizados, debido a que son las capas más estables de una aplicación. En estos casos se recomienda que la cantidad de pruebas automáticas en estos niveles sea menor a las pruebas automáticas de la base de la pirámide.

Pruebas de GUI: En este nivel (Cohn, 2009), dice que las pruebas automáticas deben ser las menores posibles debido a los siguientes factores:

- Son pruebas variables debido a la cantidad de cambios que puede tener una interfaz gráfica.
- Las pruebas de este tipo toman más tiempo que el resto de capas de una aplicación.
- Tiene muchas dependencias con otros componentes.

Pruebas manuales: Son pruebas exploratorias manuales, tienen poca documentación y casi ningún plan de pruebas en concreto.

Una vez entendida la pirámide se puede plantear una estrategia de automatización de pruebas. Las actividades a realizar son las siguientes:

- Implementar TDD (*test-driven development*) en pruebas unitarias para validar los componentes y su correcto funcionamiento de manera individual.
- Realizar flujos *end to end* para las pruebas de APIs y de integración, de manera que se puedan abarcar escenarios de pruebas suficientes para validar las reglas de negocio de las aplicaciones.
- Establecer escenarios de pruebas de interfaz de usuario, en este caso se recomienda usar Cucumber BDD como *framework* de automatización.

Al ejecutar las tareas mencionadas, se puede armar un flujo de pruebas y despliegue automático representado en la figura 21.

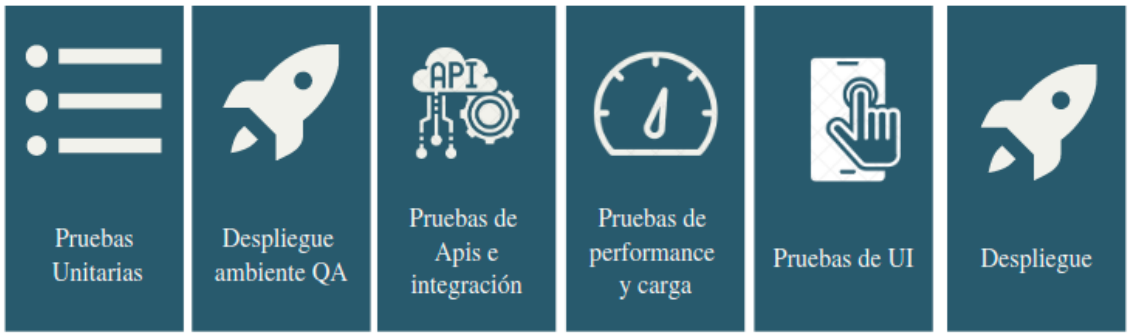


Figura 21. Flujo de pruebas y despliegue automático

Se ha agregado pruebas de performance y carga como recomendación personal, con el objetivo de medir la velocidad y el consumo de recursos de la aplicación, la cantidad usuarios simultáneos que soporta la arquitectura y la capacidad de reaccionar ante altas cargas.

6.1.3. Proceso de revisión por pares basados en CMMI Dev 1.3

Basados en el problema de Banco Ecuador en cuanto a la falta de verificación del código fuente por compañeros de trabajo del mismo nivel que los autores, se propone que se establezca el proceso de revisión por pares como lo recomienda CMMI Dev 1.3. En la verificación, un área de proceso de ingeniería, se tiene como propósito asegurar que los productos de trabajo seleccionados cumplan con los requisitos especificados.

Este es un proceso incremental y continuo debido a que se realiza durante el desarrollo del producto de software. El flujo de trabajo en este proceso consta de los siguientes pasos:

- Verificación de los requisitos funcionales y no funcionales.
- Verificación de los productos de trabajo que se van desarrollando.

- Verificación del producto terminado.

Al tener una visión de alto nivel de lo que implica la revisión entre pares, se procede a detallar su ejecución.

6.1.3.1. Realizar la revisión entre pares

Se debe aplicar esta técnica con la finalidad de identificar defectos que deberán ser eliminados posteriormente y también para recomendar los cambios que sean necesarios. CMMI Dev 1.3 sugiere aplicar esta técnica a productos de trabajo desarrollados en los proyectos y también a documentación y productos de trabajo de formación que normalmente se desarrollan por grupos de soporte. Para este caso se utilizará la revisión por pares para la verificación de código fuente con base en productos de trabajo que serán detallados más adelante.

La verificación entre pares consta de tres subprocesos:

- Preparar las revisiones entre pares.
- Realizar las revisiones entre pares.
- Analizar los datos de las revisiones entre pares.

6.1.3.1.1. Preparar las revisiones entre pares

Las actividades que se deben ejecutar en este subproceso son:

- Identificar a los participantes.
- Preparar y actualizar cualquier material a utilizar.
- Crear un calendario de la revisión entre pares.

Dentro de estas actividades, CMMI Dev 1.3 sugiere un conjunto grande de sub prácticas, de las cuales se seleccionaron las siguientes basados en las necesidades de Banco Ecuador:

- Establecer inspecciones de código fuente, como tipo de revisión entre pares.
- Establecer y mantener los criterios de entrada y salida para la revisión entre pares.
- Establecer y mantener los criterios para solicitar otra revisión entre pares.
- Establecer y mantener listas de comprobación para asegurar que los productos de trabajo se revisan consistentemente. Estas listas pueden contener:
    - Reglas de construcción.
    - Guías de diseño.
    - Completitud.
    - Grado de corrección.
    - Mantenibilidad.
    - Tipos de defectos comunes.
- Desarrollar un calendario detallado de la revisión entre pares.

6.1.3.1.2. Realizar las revisiones entre pares

La revisión se debe realizar sobre el producto de trabajo, no sobre la persona que lo realizó. Uno de los propósitos de la revisión entre pares es encontrar y eliminar defectos en fases tempranas, lo cual se alinea perfectamente con la técnica utilizada para crear la estrategia de automatización de pruebas.

Es importante seguir las siguientes pautas recomendadas por CMMI Dev 1.3 para la correcta realización de la revisión entre pares (Software Engineering Institute. 2010):

Las revisiones deben estar suficientemente preparadas.

- Gestionar y controlar la realización de las revisiones.
- Registrar datos consistentes y suficientes.
- Registrar los elementos de acción.

Las sub prácticas recomendadas son:

- Identificar y documentar defectos y otras cuestiones sobre el producto de trabajo.
- Registrar los resultados de la revisión entre pares, incluyendo los elementos de acción.
- Identificar elementos de acción y comunicar las cuestiones a las partes interesadas relevantes.

6.1.3.1.3. Analizar los datos de las revisiones entre pares

Una vez finalizada la revisión entre pares, se debe analizar los datos recolectados. Para el caso de Banco Ecuador se deberían ejecutar las siguientes sub prácticas:

- Almacenar los datos para futuras consultas y análisis.
- Analizar los datos de la revisión entre pares, entre los cuales se incluyen:
    - Fase en la que se detectó el defecto.
    - Número de defectos encontrados contra el número de defectos esperados.
    - Tipos de defectos detectados.
    - Causas de los defectos.

6.2. Arquitectura de Sistemas / Información

Las iniciativas que se proponen dentro de la arquitectura de sistemas / información están enfocadas en ayudar a plantear una tecnología inicial a utilizarse para la migración sistemas en arquitecturas *on premise* hacia la nube y a la implementación de una herramienta que ayude con la centralización de monitoreo de aplicaciones de Banco Ecuador.

6.2.1. Docker para contenerizar aplicaciones

Docker es una plataforma de software que permite crear, probar e implementar aplicaciones de manera rápida.

Se considerada necesaria utilizar esta herramienta debido a la facilidad que ofrece al momento de crear contenedores que alojarán las aplicaciones de *back end* y front end.

6.2.1.1. Actividades para la implementación

Desacoplamiento de componentes

Las aplicaciones de Banco Ecuador que actualmente están en arquitectura on premise tienen muchos componentes acoplados, como por ejemplo bases de datos, utilitarios para formatos de datos, utilitarios para generación de logs, componentes para manejo de datos en caché, etc. Por lo tanto, es necesario descomponer estos servicios en contenedores independientes.

Seleccionar imagen base

Docker cuenta con un repositorio con imágenes preestablecidas que pueden ser utilizadas como base de las aplicaciones que se van a contenerizar. A partir de la imagen seleccionada se debe iniciar con el proceso de configuración para crear los contenedores correspondientes.

Creación de Dockerfile

Se debe crear un archivo llamado Dockerfile en donde se definen los pasos necesarios para construir la imagen. Dentro de estos pasos se definen las configuraciones mínimas para que las aplicaciones puedan ser ejecutadas.

Configuración, pruebas e implementación

Las configuraciones finales son a nivel de las aplicaciones, en donde se establecen las conexiones a recursos externos o a otros grupos de contenedores. Estas configuraciones pueden estar dentro del contenedor o a nivel de variables de entorno.

Las pruebas que se deben hacer a las aplicaciones basadas en contenedores es igual a aplicaciones en arquitectura on premise. Además, es necesario realizar pruebas de carga una vez que las configuraciones hayan sido completadas.

Finalmente, se debe desplegar los contenedores siguiendo el flujo definido por el equipo *Scrum*, de manera que estén alojados en todos los ambientes de pruebas y de producción.

6.2.2. Implementar Azure Monitor

Azure Monitor es una herramienta disponible en la nube de Microsoft Azure. Esta herramienta ayuda a maximizar el rendimiento y la disponibilidad de las aplicaciones y a identificar proactivamente los problemas en un corto tiempo.

6.2.2.1. Actividades para la implementación

Registros de Log

Se propone que los logs que deben formar parte de esta implementación estén asociados a las siguientes categorías:

- Logs de Auditoría, en donde se encuentre el registro de las operaciones transaccionales de Banco Ecuador. Esta información debe estar encriptada por seguridad de la información alojada en estos logs.

- Logs de Excepciones, se recomienda esta información para tener una trazabilidad de los inconvenientes presentados en las aplicaciones que serán monitoreadas.
- Logs de Rendimiento, a pesar de que Azure Monitor cuenta con esta característica, en una primera fase se reutilizarán estos logs que ya son parte de las aplicaciones de Banco Ecuador. Se decide hacerlo de esta manera debido a que la activación de esta característica implica un desarrollo extra tanto en aplicaciones de *front end* como *back end*.

Validación de licencias

Banco Ecuador cuenta con un contrato vigente con Microsoft Azure, por lo tanto, todo proyecto que sea alojado en esta nube tiene habilitada la herramienta de monitoreo.

Recopilación de datos por Azure Monitor

Los principales logs que se analizarán con esta herramienta vienen dados de las aplicaciones de Banco Ecuador, pero Azure Monitor recopila información adicional en los siguientes niveles:

- **Datos de supervisión de aplicaciones:** Son datos sobre el rendimiento y la funcionalidad del código fuente.
- **Datos de supervisión del sistema operativo invitado:** Datos sobre el sistema operativo sobre el cual se ejecutan las aplicaciones monitoreadas.
- **Datos de supervisión de recursos de Azure:** Son los datos sobre el funcionamiento de los recursos de Azure.

Se tienen más niveles, pero para este caso puntual tenemos suficiente con los descritos.

Creación de paneles

Los paneles de esta herramienta son totalmente configurables permitiendo combinar distintos tipos de datos en un único panel. Los paneles pueden ser compartidos con otros usuarios de Azure.

En la Figura 22 se muestra un ejemplo de la presentación de estos paneles.

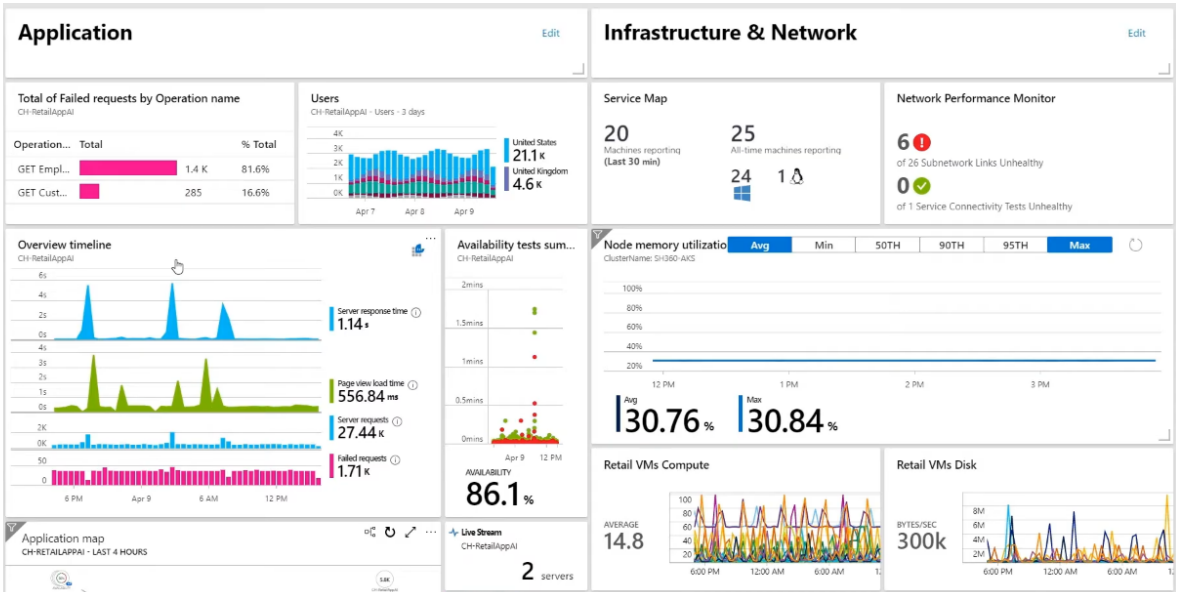


Figura 22. Paneles de Azure Monitor. Tomado de (Microsoft Azure, 2021)

Configuración de alertas

Azure Monitor permite crear alertas que ayudarán a informar sobre estados críticos de los componentes monitoreados e incluso es capaz de aplicar acciones correctivas. Las reglas configuradas en estas alertas están basadas en métricas y registros, en donde:

- Las reglas de alertas basadas en métricas proporcionan alertas casi en tiempo real con valores numéricos.
- Las reglas basadas en los registros permiten una lógica compleja con datos de varios orígenes.

Implementación

El roadmap para la implementación está representado en la Figura 23.

Figura 23. Roadmap implementación monitoreo

6.2.3. Elección de proyecto piloto

En relación con las iniciativas planteadas, es necesario contar con un proyecto piloto sobre el cual se puedan aplicar todas las mejoras propuestas. Las características con las que debe contar este proyecto son:

- Tener pocas funcionalidades en producción.
- Contar con un número de usuarios bajo.
- Contar con un equipo a cargo del producto con conocimientos tecnológicos y de negocio.
- El proyecto debe estar desplegado en un ambiente on premise.

6.3. Arquitectura de Datos

Con el fin de organizar y centralizar la documentación se sugiere la implementación de Confluence. Esta herramienta permite crear, colaborar y organizar todo el trabajo en un solo lugar.

Confluence ofrece diferentes opciones de alojamiento como *cloud*, *data center* y *server*, se recomienda la opción de *cloud* para aprovechar las ventajas que ofrece

en cuanto a mantenimiento, actualizaciones automáticas, rendimiento y escalado elástico.

6.3.1. Implementación de Confluence

La implementación consta de cinco pasos:

Paso 1: Conocer los espacios

Confluence se organiza en espacios. Los espacios son colecciones de páginas relacionadas en las que se trabaja en conjunto con los usuarios que tienen acceso. Se pueden crear cuantos espacios sean necesarios.

Para Banco Ecuador es necesario crear los siguientes espacios:

- Documentación técnica.
- Documentación correspondiente a flujos de negocio.
- Documentación de minutas de reuniones.
- Planes de despliegue.
- Diccionario de datos.

Paso 2: Crear un espacio

Una vez definidos los espacios, se los puede crear de la siguiente manera:

- Ir al sitio de Confluence de la organización.
- En la pantalla de inicio seleccionar “Crear espacio”.
- Seleccionar el tipo de espacio a crear.
- Llenar los campos que se solicitan.
- Establecer los permisos para el espacio creado.
- Clic en “Crear”.

Al finalizar la creación se direccionará al espacio de “Presentación”. Aquí se puede detallar el objetivo del espacio creado y quienes tendrán acceso.

Paso 3: Gestionar usuarios y permisos

Con la cuenta de administrador, Confluence permite gestionar usuarios, grupos y permisos de forma manual. Los permisos globales que permite controlar son (Atlassian, 2021):

- Permisos para crear espacios.
- Gestionar accesos a usuarios sin licencia.
- Accesos a los perfiles de usuario.

Además, se pueden gestionar los siguientes permisos a los espacios creados:

- Acceso al contenido de un espacio.
- Realizar comentarios a contenidos con acceso.
- Creación, edición o carga de contenido.

Confluence es público de manera predeterminada, así que, si no se hace una correcta gestión de permisos, cualquier persona con acceso al sitio de Confluence de Banco Ecuador podrá ver y modificar todo el contenido disponible.

6.4. Arquitectura Tecnológica

Actualmente Banco Ecuador cuenta con una licencia activa de la nube de Microsoft Azure por lo tanto se propone migrar las aplicaciones en arquitectura on premise hacia esta nube. Se recomienda establecer un plan de migración estructurado de la siguiente manera:

- Plan de monitoreo: Se necesita reestructurar el plan de monitoreo de las aplicaciones que se migrarán a la nube, basados en los componentes involucrados en esta arquitectura.

- Diagramas de arquitectura: Se deben definir los nuevos diagramas de arquitectura.
- Documentación de capacity: Es necesario ejecutar pruebas de capacidad de la arquitectura actual para tener claro el dimensionamiento que se debe considerar al implementar la nueva arquitectura.
- Definir un grupo resolutor: Se necesita tener un grupo resolutor en caso de que los componentes de la arquitectura de la nube fallen, de esta manera, se asegura su correcto funcionamiento y ejecución de planes contingentes en caso de que sea necesario. Actualmente Banco Ecuador cuenta con un grupo resolutor, el cual debe ser analizado y modificado si es el caso.
- Adjuntar el plan de ponderaciones: Toda aplicación de Banco Ecuador cuenta con un plan de ponderaciones el cual muestra de mayor a menor el grado de criticidad de sus funcionalidades. Se debe adjuntar este plan para mantener el nivel de disponibilidad de cada funcionalidad.
- Revisar la matriz de escalamiento: Esta matriz contiene el orden de involucrados y su nivel de responsabilidad ante la presencia de un incidente. Es necesario validar el estado actual de la matriz frente a la migración hacia la nube.
- Revisar la matriz de riesgo: Validar si la documentación actual cumple con todos los riesgos involucrados después de la migración.

6.5. Consolidación de iniciativas

A continuación, se muestra la consolidación de las iniciativas encontradas. Todas las implementaciones serán hechas por el personal de Banco Ecuador, por lo tanto, el valor aproximado se calcula con base en el personal necesario y el tiempo empleado.

Tabla 39
Consolidación de iniciativas

Arquitectura	Iniciativa	Costo aproximado ($)
Negocio	Priorizar los ítems del *product backlog* basados en la técnica MoSCoW	32000.00
Negocio	Establecer la estrategia para la implementación de pruebas automáticas basados en la pirámide de Cohn	8582.90
Negocio	Establecer el proceso de revisión por pares basados en CMMI Dev 1.3	35533.00
Sistemas/Información	Docker para contenerizar aplicaciones	8800.00
Sistemas/Información	Implementación de Azure Monitor	15500.00
Datos	Implementación de Confluence	2250.00
Tecnológica	Migración hacia nube de Microsoft Azure	33800.00

7. Plan de migración

El presente capítulo propone una priorización de las iniciativas con base en el análisis de impacto y el análisis de esfuerzo y mantiene un orden lógico para su posterior implementación.

7.1. Análisis de impacto

Para el análisis de impacto se realizó una matriz de correlación entre los objetivos del área de la fábrica de software in house y las iniciativas detalladas en el capítulo anterior. La rúbrica que se utilizó para encontrar el impacto en cada iniciativa es: Bajo (0 - 0.7), Medio (0.8 – 1.4), Alto (1.5 - 2).

Tabla 40
Análisis de impacto

				Objetivos								
				20%	18%	8%	12%	16%	12%	14%	1	
No	Arquitectura	Id	Iniciativa	Potenciar los canales digitales disponibles a través de planes de mantenimiento y liberación de nuevas funcionalidades.	Fortalecer los controles de calidad al código fuente en versiones nuevas y existentes.	Mejorar la clasificación y el acceso a la documentación disponible.	Centralizar el monitoreo de los componentes de las aplicaciones.	Implementar una estrategia para la automatización de pruebas.	Fortalecer el proceso de priorización de nuevas funcionalidades en los productos de Banco Ecuador.	Estandarizar el manejo de logs en todas las aplicaciones.	Valoracion cualitativa	Impacto
1	Negocio	PN01	Priorizar los ítems del Product Backlog basados en la técnica MoSCow								0,74	Bajo
2	Negocio	PN02	Establecer la estrategia para la implementación de pruebas automáticas basados en la pirámide de Cohn								0,90	Medio
3	Negocio	PN03	Establecer el proceso de revisión por pares basados en CMMI Dev 1.3								1,00	Medio
4	Sistemas/Información	PS04	Docker para contenerizar aplicaciones								1,34	Medio
5	Sistemas/Información	PS05	Implementación de Azure Monitor								1,02	Medio
6	Datos	PD06	Implementación de Confluence								0,70	Bajo
7	Tecnológica	PD07	Migración hacia nube de Microsoft Azure								1,58	Alto

Nota. Se mide el impacto que tiene cada iniciativa sobre los objetivos empresariales de Banco Ecuador.

7.2. Análisis de esfuerzo

Para el análisis de esfuerzo se consideraron tres criterios de esfuerzo: recursos económicos, complejidad y capacidad de TI. Con el resultado obtenido se lo categorizó en tres niveles de esfuerzo basados en la siguiente rúbrica: Bajo (1 – 1.7), Medio (1.7 – 2.4), Alto (2.4 - 3).

Tabla 41
Análisis de esfuerzo

				Criterios Esfuerzo				
				40%	30%	30%		
No	Área	Id	Iniciativa	Recursos Económicos	Complejidad	Capacidad TI	Suma ponderada	Esfuerzo
1	Negocio	PN01	Priorizar los ítems del Product Backlog basados en la técnica MoSCow	●	●	●	1,00	Bajo
2	Negocio	PN02	Establecer la estrategia para la implementación de pruebas automáticas basados en la pirámide de Cohn	●	▲	●	1,90	Medio
3	Negocio	PN03	Establecer el proceso de revisión por pares basados en CMMI Dev 1.3	●	◆	●	1,90	Medio
4	Sistemas/Información	PS04	Docker para contenerizar aplicaciones	●	▲	●	1,60	Bajo
5	Sistemas/Información	PS05	Implementación de Azure Monitor	●	▲	●	1,90	Medio
6	Datos	PD06	Implementación de Confluence	▲	●	●	1,70	Bajo
7	Tecnológica	PD07	Migración hacia nube de Microsoft Azure	◆	▲	●	2,70	Alto

Escala de Esfuerzo
Bajo: entre 1 - 1,7
Medio: entre 1,7 y 2,4
Alto: entre 2,4 y 3

Nota. Se mide el esfuerzo necesario para completar las iniciativas.

7.3. Fases

Se combinaron los resultados obtenidos en el análisis de impacto y el análisis de esfuerzo y se dividieron las iniciativas en fases. Estas fases ayudarán más adelante a crear el cronograma de actividades para la implementación.

Tabla 42
Fases

No	Dominio	Id	Iniciativa	Impacto	Esfuerzo	Prioridad	Fase
1	Negocio	PN01	Priorizar los ítems del Product Backlog basados en la técnica MoSCow	Bajo	Bajo	Habilitante	1
2	Negocio	PN02	Establecer la estrategia para la implementación de pruebas automáticas basados en la pirámide de Cohn	Medio	Medio	Habilitante	1
3	Negocio	PN03	Establecer el proceso de revisión por pares basados en CMMI Dev 1.3	Medio	Medio	Habilitante	2
4	Sistemas/Información	PS04	Docker para contenerizar aplicaciones	Medio	Bajo	Habilitante	2
5	Sistemas/Información	PS05	Implementación de Azure Monitor	Medio	Medio	Habilitante	3
6	Datos	PD06	Implementación de Confluence	Bajo	Bajo	Habilitante	4
7	Tecnológica	PD07	Migración hacia nube de Microsoft Azure	Alto	Alto	Alta	4

Nota. Se distribuyen las iniciativas en fases con base en el impacto y el esfuerzo de cada una.

7.4. *Roadmap*

Se establece un *roadmap* basado en el orden de implementación de cada iniciativa por cada arquitectura. El detalle del *roadmap* se encuentra en el Anexo 3.

7.5. Cronograma de actividades

Tabla 43
Cronograma para elección del proyecto piloto

Id	Nombre	Duración	Comienzo	Fin
1	**Elección del proyecto piloto**	**3 días**	**mar 27/4/21**	**jue 29/4/21**
2	Preparar documentación del proyecto piloto	**30 días**	**vie 30/4/21**	**jue 10/6/21**
3	Aprobar documentación	4 días	vie 30/4/21	mié 5/5/21
4	Aprobar proyecto	1 día	jue 6/5/21	jue 6/5/21

Tabla 44
Cronograma para priorización de *product backlog*

Id	Nombre	Duración	Comienzo	Fin
5	**Priorización de *product backlog***	4 días	vie 7/5/21	mié 12/5/21
6	Identificar *stakeholders*	**21 días**	**jue 13/5/21**	**jue 10/6/21**
7	Definir actor principal	5 días	jue 13/5/21	mié 19/5/21
8	Listar características de *items* de *product backlog*	5 días	jue 20/5/21	mié 26/5/21

Tabla 45
Cronograma para priorización de ítems de *product backlog* aplicando Moscow

Id	Nombre	Duración	Comienzo	Fin
9	**Priorizar aplicando Moscow**	5 días	jue 27/5/21	mié 2/6/21
10	Taller para priorizar fase 1	5 días	jue 3/6/21	mié 9/6/21
11	Taller para priorizar fase 2	1 día	jue 10/6/21	jue 10/6/21
12	Taller para priorizar fase 3	**59 días**	**vie 11/6/21**	**mié 1/9/21**
13	Transformar *items* de *product backlog* en historias de usuario	3 días	vie 11/6/21	mar 15/6/21
14	Presentar historias de usuario priorizadas	1 día	mié 16/6/21	mié 16/6/21

Tabla 46
Cronograma para automatización de pruebas

Id	Nombre	Duración	Comienzo	Fin
15	**Automatización de pruebas**	1 día	jue 17/6/21	jue 17/6/21
16	Listar funcionalidades de aplicaciones	2 días	vie 18/6/21	lun 21/6/21
17	Distribuir funcionalidades al *back end*	15 días	jue 22/7/21	mié 11/8/21
18	Distribuir funcionalidades al *front end*	15 días	jue 12/8/21	mié 1/9/21
19	Distribuir funcionalidades del *back end* que puedan ser probadas con *tests* unitarios	3 días	mar 20/7/21	jue 22/7/21
20	Aplicar TDD en pruebas unitarias	15 días	vie 23/7/21	jue 12/8/21
21	Aplicar pruebas *end to end* a funcionalidades *back end*	**35 días**	**vie 13/8/21**	**jue 30/9/21**
22	Establecer guion de pruebas para UX	1 día	vie 13/8/21	vie 13/8/21
23	Aplicar Cucumber para pruebas de *front end*	**9 días**	**lun 16/8/21**	**jue 26/8/21**

Tabla 47
Cronograma para revisión por pares primera parte

Id	Nombre	Duración	Comienzo	Fin
24	**Revisión por pares**	2 días	lun 16/8/21	mar 17/8/21
25	Identificar a los participantes	2 días	mié 18/8/21	jue 19/8/21
26	**Verificar los requisitos funcionales**	5 días	vie 20/8/21	jue 26/8/21
27	Listar historias de usuario marcadas como terminadas	**9 días**	**vie 27/8/21**	**mié 8/9/21**
28	Tomar las historias de usuario que sean parte de los requisitos funcionales	2 días	vie 27/8/21	lun 30/8/21
29	Revisar lista de historias de usuario funcionales	2 días	mar 31/8/21	mié 1/9/21
30	**Verificar requisitos no funcionales**	5 días	jue 2/9/21	mié 8/9/21
31	Listar historias de usuario marcadas como terminadas	2 días	jue 9/9/21	vie 10/9/21
32	Tomar las historias de usuario que sean parte de los requisitos no funcionales	**4 días**	**lun 13/9/21**	**jue 16/9/21**
33	Revisar lista de historias de usuario no funcionales	1 día	lun 13/9/21	lun 13/9/21
34	Preparar material a utilizar	1 día	mar 14/9/21	mar 14/9/21

Nota. Contiene las tareas de verificar los requisitos funcionales y verificar requisitos no funcionales.

Tabla 48
Cronograma para revisión por pares segunda parte

Id	Nombre	Duración	Comienzo	Fin
35	**Establecer listas de comprobación para revisión de pares**	1 día	mié 15/9/21	mié 15/9/21
36	Listas de reglas de construcción	1 día	jue 16/9/21	jue 16/9/21
37	Listas para guías de diseño	**6 días**	**vie 17/9/21**	**vie 24/9/21**
38	Listas para determinar completitud de un producto	2 días	vie 17/9/21	lun 20/9/21
39	Listas de tipos de defectos comunes	2 días	mar 21/9/21	mié 22/9/21
40	**Realizar la revisión entre pares**	2 días	jue 23/9/21	vie 24/9/21
41	Identificar defectos sobre los productos revisados	**3 días**	**lun 27/9/21**	**mié 29/9/21**
42	Documentar los defectos encontrados	1 día	lun 27/9/21	lun 27/9/21
43	Identificar los elementos de acción	2 días	mar 28/9/21	mié 29/9/21
44	**Analizar los datos de las revisiones entre pares**	**35 días**	**jue 30/9/21**	**mié 17/11/21**
45	Almacenar la documentación obtenida en la revisión entre pares	**39 días**	**jue 18/11/21**	**mar 11/1/22**
46	Analizar los datos	10 días	jue 18/11/21	mié 1/12/21

Nota. Contiene las tareas de establecer lista de comprobación, realizar la revisión entre pares y analizar los datos de las revisiones entre pares.

Tabla 49
Cronograma para contenerización de aplicaciones

Id	Nombre	Duración	Comienzo	Fin
47	**Contenerización de aplicaciones**	2 días	jue 2/12/21	vie 3/12/21
48	**Desacoplar componentes**	2 días	lun 6/12/21	mar 7/12/21
49	Revisar arquitectura de aplicaciones en on premise	15 días	mié 8/12/21	mar 28/12/21
50	Identificar componentes involucrados	2 días	mié 29/12/21	jue 30/12/21
51	Categorizar componentes	2 días	vie 31/12/21	lun 3/1/22
52	Establecer arquitectura de componentes desacoplados	1 día	mar 4/1/22	mar 4/1/22
53	Seleccionar imagen base	3 días	mié 5/1/22	vie 7/1/22
54	Creación de Dockerfile para componentes desacoplados	2 días	lun 10/1/22	mar 11/1/22
55	Configurar aplicaciones contenerizadas	**5 días**	**mié 12/1/22**	**mar 18/1/22**
56	Pruebas de aplicaciones contenerizadas	2 días	mié 12/1/22	jue 13/1/22
57	**Implementación**	**2 días**	**vie 14/1/22**	**lun 17/1/22**
58	Despliegue en ambiente de desarrollo	**1 día**	**mar 18/1/22**	**mar 18/1/22**
59	Despliegue en ambiente de test	**38 días**	**mié 12/1/22**	**vie 4/3/22**
60	Despliegue en ambiente de preproducción	5 días	mié 12/1/22	mar 18/1/22
61	Despliegue en ambiente de producción	10 días	mié 19/1/22	mar 1/2/22

Tabla 50
Cronograma para implementación de Azure Monitor

Id	Nombre	Duración	Comienzo	Fin
62	**Implementación de Azure Monitor**	**11 días**	**mié 2/2/22**	**mié 16/2/22**
63	Categorizar los logs de las aplicaciones	1 día	mié 2/2/22	mié 2/2/22
64	Almacenar logs en tablas de Azure	3 días	jue 10/2/22	lun 14/2/22
65	Definir paneles para monitoreo	2 días	mar 15/2/22	mié 16/2/22
66	Crear *queries* hacia tablas con registros de logs	2 días	jue 17/2/22	vie 18/2/22
67	Generar paneles de monitoreo	3 días	lun 21/2/22	mié 23/2/22
68	Identificar personal a cargo de paneles de monitoreo	2 días	jue 24/2/22	vie 25/2/22
69	Exportar paneles	5 días	lun 28/2/22	vie 4/3/22
70	Crear alertas	3 días	5-ene-22	7-ene-22
71	Gestionar accesos a paneles	2 días	10-ene-22	11-ene-22

Tabla 51
Cronograma para implementación de Confluence

Id	Nombre	Duración	Comienzo	Fin
72	**Implementación de Confluence**	**47 días**	**15-nov-21**	**18-ene-22**
73	Identificar los espacios a crearse	2 días	12-ene-22	13-ene-22
74	**Crear espacios**	**46 días**	**15-nov-21**	**17-ene-22**
75	Crear espacio para documentación técnica	2 horas	14-ene-22	14-ene-22
76	Crear espacio para documentación de flujo de negocio	2 horas	14-ene-22	14-ene-22
77	Crear espacio para documentación de minutos de reuniones	2 horas	14-ene-22	14-ene-22
78	Crear espacio para documentación de despliegues de versiones	2 horas	14-ene-22	14-ene-22
79	Crear espacio para diccionario de datos	2 horas	17-ene-22	17-ene-22
80	Identificar los usuarios que tendrán acceso a los espacios	6 horas	17-ene-22	17-ene-22
81	**Gestionar accesos**	**45 días**	**17-nov-21**	**18-ene-22**
82	Permisos para crear espacios	2 horas	18-ene-22	18-ene-22
83	Permisos para acceso a contenido de espacios	2 horas	18-ene-22	18-ene-22
84	Permisos para crear comentarios en espacios disponibles	2 horas	18-ene-22	18-ene-22
85	Permisos para gestionar contenido en espacios	2 horas	18-ene-22	18-ene-22

Tabla 52
Cronograma para la migración de hacia Microsoft Azure

Id	Nombre	Duración	Comienzo	Fin
86	**Migración hacia Microsoft Azure**	**64 días**	**19-ene-22**	**18-abr-22**
87	Reestructurar plan de monitoreo	5 días	19-ene-22	25-ene-22
88	Definir documentación de diagrama de arquitectura	10 días	26-ene-22	8-feb-22
89	***Capacity***	**49 días**	**9-feb-22**	**18-abr-22**
90	Establecer pruebas de capacidad	1 día	9-feb-22	9-feb-22
91	Ejecutar pruebas de capacidad a arquitectura actual	3 días	10-feb-22	14-feb-22
92	Documentar resultado de pruebas	2 días	15-feb-22	16-feb-22
93	Definir grupos resolutores	2 días	17-feb-22	18-feb-22
94	Revisar plan de ponderaciones	3 días	21-feb-22	23-feb-22
95	Revisar matriz de escalamiento	2 días	24-feb-22	25-feb-22
96	Revisar matriz de riesgo	5 días	28-feb-22	4-mar-22

Conclusiones

- Es importante involucrar a los desarrolladores en la automatización de pruebas, de manera que con el desarrollo de software sea sencillo cubrir todas las funcionalidades y validaciones existentes en las pruebas. De esta manera se crea un código de calidad y con mejor rendimiento en cuanto a entrega de funcionalidades.
- El uso de la tecnología y el fortalecimiento de productos ofrecidos por la industria bancaria es una tendencia tal como se lo describe en el contexto de la fase preliminar, por lo tanto, el desarrollo de este trabajo potencia de gran manera el camino hacia la transformación digital de Banco Ecuador.
- Contar con un monitoreo adecuado de las aplicaciones de la organización, garantiza una oportuna ejecución de acciones correctivas necesarias para evitar problemas de indisponibilidad y por ende afectación en el negocio.
- Aplicar correctamente una metodología ágil ayuda a gestionar de mejor manera cualquier proyecto de *software* que se lleve a cabo dentro de la organización, debido al nivel de participación que se da a todos los involucrados en el proyecto. De esta manera se logran productos o servicios que verdaderamente aportan valor y son acogidos por los usuarios.
- La migración de proyectos que cuentan con arquitecturas *on premise* hacia arquitecturas en la nube, deben estar acompañadas de un análisis detallado de su situación actual, con la finalidad de contar con una línea base, a partir de la cual se pueda dimensionar los nuevos componentes que conformarán la arquitectura del proyecto.
- El compromiso de Banco Ecuador de ofrecer canales digitales óptimos y de excelente calidad que reemplacen a los físicos se ve cubierto gracias a la implementación de pruebas automáticas y a la revisión de código por pares. De esta manera se garantiza un producto final de calidad y que cubre las necesidades de sus clientes.

- Debido al gran número de usuarios afiliados a canales digitales, es necesario contar con metodologías ágiles que ayuden a liberar más rápidamente nuevas funcionalidades que cubran las necesidades de los clientes finales. *Scrum* se adapta perfectamente a este requerimiento, brindando las herramientas necesarias para crear proyectos exitosos en tiempos más cortos que con metodologías tradicionales.

Recomendaciones

- Es indispensable contar con personal capacitado y comprometido en la implementación de las iniciativas planteadas. De esta manera se asegura una correcta finalización en cada proyecto.
- A pesar de que se sugiere la implementación de pruebas automáticas, se debe poner especial atención en la gestión de errores pendientes de revisión, con la finalidad de no dejar por fuera casos en donde el código fuente de las aplicaciones de Banco Ecuador genera problemas en su calidad y experiencia de usuario.
- Para poder tener un correcto proceso de priorización de ítems del *product backlog*, es indispensable que se cuente con el equipo humano completo, de manera que a través de este proceso se pueda encaminar el proyecto hacia un objetivo común concebido por todos los involucrados.
- Las iniciativas propuestas en el capítulo de oportunidades y soluciones deben contar con el patrocinador adecuado para que se puedan ejecutar con éxito. Por lo cual es importante dar a conocer la problemática actual de la organización en todas las arquitecturas analizadas en el presente capstone. Sin el apoyo adecuado, ningún proyecto planteado puede llegar a ser implementado con éxito.

Referencias

Atlassian Agile Coach. (4 de junio de 2021). *Historias de usuario con ejemplos y plantilla.* https://www.atlassian.com/es/agile/project-management/user-stories

Atlassian. (6 de abril de 2021). *Conceptos básicos de Confluence.* https://www.atlassian.com/es/software/confluence/guides/get-started/confluence-overview

Atlassian. (6 de abril de 2021). *Configurar tu sitio y tus espacios.* https://www.atlassian.com/es/software/confluence/guides/get-started/set-up

BBVA en 2014. (2014). https://accionistaseinversores.bbva.com/wp-content/uploads/2017/02/BBVAen2014_tcm926-508564.pdf

Cohn, M. (2009). *Succeeding with Agile: Software Development Using Scrum*. (1a ed.). Canada: Addison-Wesley Professional.

IIBA, (2005). *A Guide to the Business Analysis Body of Knowledge* Versión 2.0. Estados Unidos: Kevin Brennan. p. 102

Izrailevsky, Y., Vlaovic, S., Meshenberg, R. (10 de octubre de 2020). *Netflix se muda a la nube*. https://about.netflix.com/es/news/completing-the-netflix-cloud-migration

Kang et al. (2005). *The Open Web Application Security*. Edición 2.0 Black Hat. Estados Unidos.

Marous, J. (2019). *Innovation in Retail Banking*. Temenos. página 15

Marous, J. (2020). *Retail Banking Trends & Prorities*. Temenos. página 21

Menzinsky, A., López, G., Palacio, J. (2019). *Scrum Manager – Troncal I* Versión 2.6.1. Iubaris Info 4 Media SL.

Microsoft Azure. (3 de abril de 2021). *Azure Monitor.* https://azure.microsoft.com/es-mx/services/monitor/

Microsoft. (26 de febrero de 2021). *Confianza descentralizada entre bancos*. https://docs.microsoft.com/es-es/azure/architecture/example-scenario/apps/decentralized-trust

Microsoft Azure. (5 de abril de 2021). *Introducción a Azure Monitor.* https://docs.microsoft.com/es-mx/azure/azure-monitor/overview

Microsoft. (13 de mayo de 2020). *Resistencia de la plataforma Azure.* https://docs.microsoft.com/es-es/dotnet/architecture/cloud-native/infrastructure-resiliency-azure

Microsoft. (19 de enero de 2021). *Seguridad de Azure para aplicaciones nativas en la nube.* https://docs.microsoft.com/es-es/dotnet/architecture/cloud-native/azure-security

Miranda, E. (2011). *The boxing planning: Buffered Moscow rules*. ACM SIGSOFT Software Engineering.

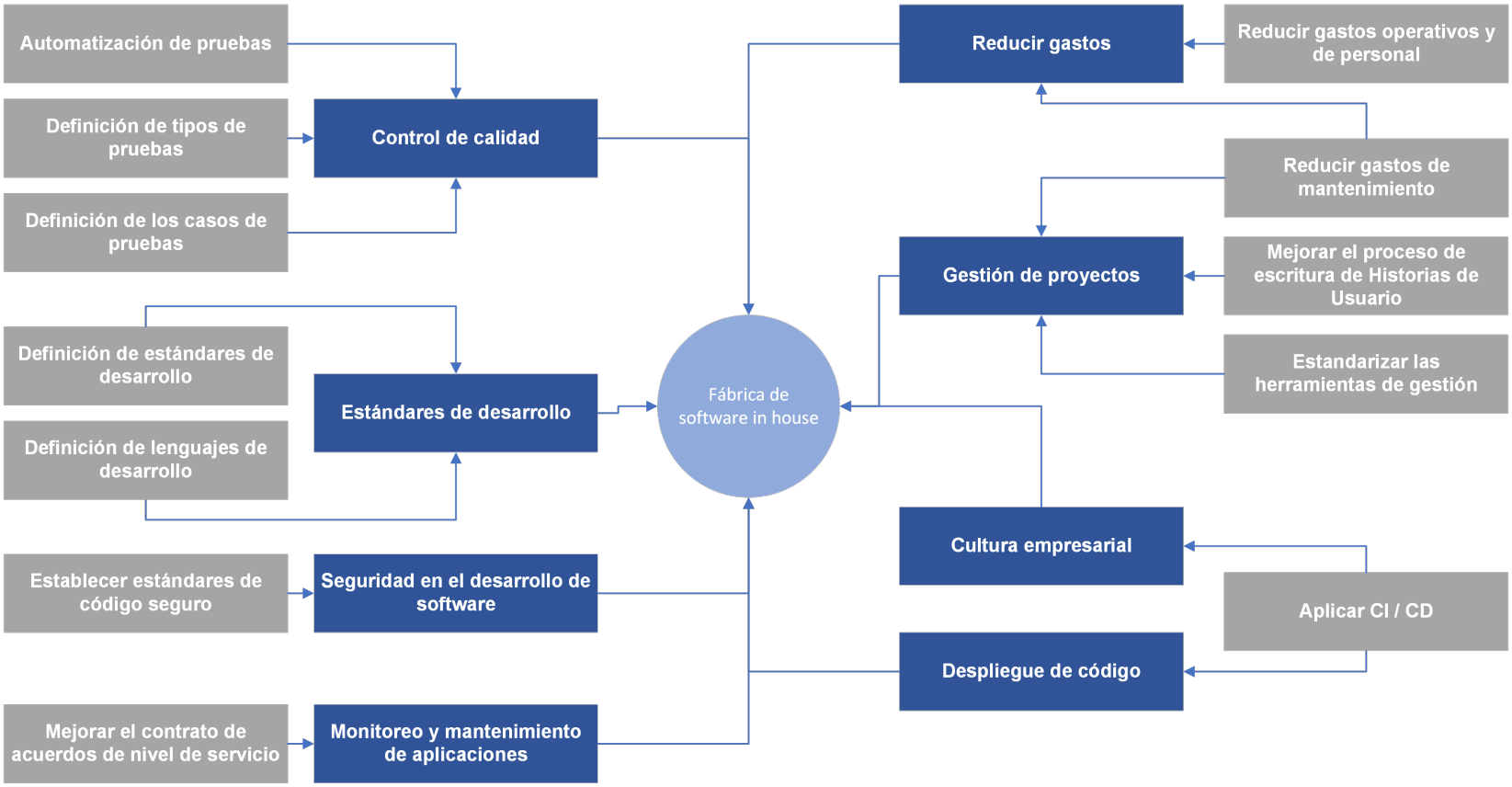
Schwaber ,k., Sutherland, J. (2020). *La Guía de Scrum*.

Software Engineering Institute. (2010). *CMMI para Desarrollo*, Versión 1.3. Estados Unidos: Editorial Universitaria Ramón Areces.

The Open Group. (2018). *The TOGAF Standard*, Versión 9.2. Estados Unidos: The Open Group. p 201 – 212

ANEXOS

Anexo 1.


Automatización de pruebas
Definición de tipos de pruebas
Definición de los casos de pruebas
Control de calidad
Definición de estándares de desarrollo
Definición de lenguajes de desarrollo
Estándares de desarrollo
Establecer estándares de código seguro
Seguridad en el desarrollo de software
Mejorar el contrato de acuerdos de nivel de servicio
Monitoreo y mantenimiento de aplicaciones
Fábrica de software in house
Reducir gastos
Reducir gastos operativos y de personal
Reducir gastos de mantenimiento
Gestión de proyectos
Mejorar el proceso de escritura de Historias de Usuario
Estandarizar las herramientas de gestión
Cultura empresarial
Aplicar CI / CD
Despliegue de código

Anexo 2.

Integración

	Madurez y Capacidad		Puntaje	3,0	4,1
1			Puntaje Ponderado	3,0	4,1
ID	Dimensión	Descripción	Nivel de Importancia	Actual	Objetivo
1	Vigencia tecnológica	Nivel de incorporación de componentes tecnológicos modernos y aceptados por el mercado de soluciones de software y cumplimiento de estándares internacionales para usabilidad, robustez, portabilidad	Muy Importante	Nivel 3	Nivel 4
2	Interoperabilidad	Capacidad de la aplicación para intercambiar información de manera automática con otras aplicaciones	Muy Importante	Nivel 3	Nivel 5
3	Confiabilidad	Esquema de garantía que ofrece la aplicación para dar resultados precisos, un buen desempeño y consistencia de los datos.	Muy Importante	Nivel 3	Nivel 4
4	Arquitectura	Esquema de administración del conocimiento asociado a la aplicación, modelos arquitectonicos, diseños, guias de uso, administración actualizados.	Importante	Nivel 4	Nivel 4
5	Soporte	Esquema de apoyo/soporte con el que cuenta la aplicación para atender inquietudes de los usuarios, incidentes y requerimientos de operación.	Muy Importante	Nivel 2	Nivel 4
6	Seguridad	Nivel de protección con que cuenta la aplicación para contrarrestar los riesgos de vulnerabilidad que afecten la disponibilidad, integridad y confidencialidad de la información y los servicios tecnologicos que asiste.	Importante	Nivel 3	Nivel 4
7	Flexibilidad/Mantenimiento	Capacidad de la aplicación para adaptarla a nuevos requerimientos en tiempos y costos razonables a los alcances del cambio, con la asistencia a versionamiento, documentación y marco de trabajo para los cambios.	Importante	Nivel 3	Nivel 5
8	Gobernabilidad	Esquema de supervisión ejecutiva que se le da a la aplicación y aseguramiento de su alineamiento estrategico con las prioridades del negocio.	Muy Importante	Nivel 3	Nivel 3

Seguridad

	Madurez y Capacidad - Aplicaciones de Seguridad		Puntaje	3,6	4,6
1			Puntaje Ponderado	3,7	4,7
ID	Dimensión	Descripción	Nivel de Importancia	Actual	Objetivo
1	Vigencia tecnológica	Nivel de incorporación de componentes tecnológicos modernos y aceptados por el mercado de soluciones de software y cumplimiento de estándares internacionales para usabilidad, robustez, portabilidad.	Muy Importante	Nivel 3	Nivel 4
2	Interoperabilidad	Capacidad de la aplicación para intercambiar información de manera automática con otras aplicaciones.	Muy Importante	Nivel 3	Nivel 5
3	Confiabilidad	Esquema de garantía que ofrece la aplicación para dar resultados precisos, un buen desempeño y consistencia de los datos.	Muy Importante	Nivel 5	Nivel 5
4	Arquitectura	Esquema de administración del conocimiento asociado a la aplicación, modelos arquitectonicos, diseños, guias de uso, administración actualizados.	Importante	Nivel 3	Nivel 4
5	Soporte	Esquema de apoyo/soporte con el que cuenta la aplicación para atender inquietudes de los usuarios, incidentes y requerimientos de operación.	Muy Importante	Nivel 3	Nivel 5
6	Seguridad	Nivel de protección con que cuenta la aplicación para contrarrestar los riesgos de vulnerabilidad que afecten la disponibilidad, integridad y confidencialidad de la información y los servicios tecnologicos que asiste.	Muy Importante	Nivel 5	Nivel 5
7	Flexibilidad/Mantenimiento	Capacidad de la aplicación para adaptarla a nuevos requerimientos en tiempos y costos razonables a los alcances del cambio, con la asistencia a versionamiento, documentación y marco de trabajo para los cambios.	Muy Importante	Nivel 2	Nivel 4
8	Gobernabilidad	Esquema de supervisión ejecutiva que se le da a la aplicación y aseguramiento de su alineamiento estrategico con las prioridades del negocio.	Muy Importante	Nivel 5	Nivel 5

Automatización de pruebas

1	**Madurez y Capacidad**		**Puntaje**	1,4	3,3
			Puntaje Ponderado	1,4	3,3
ID	**Dimensión**	**Descripción**	**Nivel de Importancia**	**Actual**	**Objetivo**
1	Vigencia tecnológica	Nivel de incorporación de componentes tecnológicos modernos y aceptados por el mercado de soluciones de software y cumplimiento de estándares internacionales para usabilidad, robustez, portabilidad.	Importante	Nivel 1	Nivel 3
2	Interoperabilidad	Capacidad de la aplicación para intercambiar información de manera automática con otras aplicaciones.	Muy Importante	Nivel 1	Nivel 4
3	Confiabilidad	Esquema de garantía que ofrece la aplicación para dar resultados precisos, un buen desempeño y consistencia de los datos.	Importante	Nivel 2	Nivel 4
4	Arquitectura	Esquema de administración del conocimiento asociado a la aplicación, modelos arquitectonicos, diseños, guias de uso, administración actualizados.	Importante	Nivel 2	Nivel 3
5	Soporte	Esquema de apoyo/soporte con el que cuenta la aplicación para atender inquietudes de los usuarios, incidentes y requerimientos de operación.	Poco Importante	Nivel 1	Nivel 3
6	Seguridad	Nivel de protección con que cuenta la aplicación para contrarrestar los riesgos de vulnerabilidad que afecten la disponibilidad, integridad y confidencialidad de la información y los servicios tecnologicos que asiste.	Importante	Nivel 1	Nivel 3
7	Flexibilidad/Mantenimiento	Capacidad de la aplicación para adaptarla a nuevos requerimientos en tiempos y costos razonables a los alcances del cambio, con la asistencia a versionamiento, documentación y marco de trabajo para los cambios.	Importante	Nivel 2	Nivel 3
8	Gobernabilidad	Esquema de supervisión ejecutiva que se le da a la aplicación y aseguramiento de su alineamiento estrategico con las prioridades del negocio.	Poco Importante	Nivel 1	Nivel 3

Analítica de datos

1	Madurez y Capacidad - Sistema de activos v1 (Desarrollo inhouse)		Puntaje	2,3	4,0
			Puntaje Ponderado	2,5	4,2
ID	Dimensión	Descripción	Nivel de Importancia	Actual	Objetivo
1	Vigencia tecnológica	Nivel de incorporación de componentes tecnológicos modernos y aceptados por el mercado de soluciones de software y cumplimiento de estándares internacionales para usabilidad, robustez, portabilidad.	Muy Importante	Nivel 4	Nivel 4
2	Interoperabilidad	Capacidad de la aplicación para intercambiar información de manera automática con otras aplicaciones.	Poco Importante	Nivel 1	Nivel 3
3	Confiabilidad	Esquema de garantía que ofrece la aplicación para dar resultados precisos, un buen desempeño y consistencia de los datos.	Muy Importante	Nivel 3	Nivel 5
4	Arquitectura	Esquema de administración del conocimiento asociado a la aplicación, modelos arquitectonicos, diseños, guias de uso, administración actualizados.	Importante	Nivel 1	Nivel 3
5	Soporte	Esquema de apoyo/soporte con el que cuenta la aplicación para atender inquietudes de los usuarios, incidentes y requerimientos de operación.	Importante	Nivel 1	Nivel 3
6	Seguridad	Nivel de protección con que cuenta la aplicación para contrarrestar los riesgos de vulnerabilidad que afecten la disponibilidad, integridad y confidencialidad de la información y los servicios tecnologicos que asiste.	Muy Importante	Nivel 3	Nivel 5
7	Flexibilidad/Mantenimiento	Capacidad de la aplicación para adaptarla a nuevos requerimientos en tiempos y costos razonables a los alcances del cambio, con la asistencia a versionamiento, documentación y marco de trabajo para los cambios.	Importante	Nivel 2	Nivel 4
8	Gobernabilidad	Esquema de supervisión ejecutiva que se le da a la aplicación y aseguramiento de su alineamiento estrategico con las prioridades del negocio.	Muy Importante	Nivel 3	Nivel 5

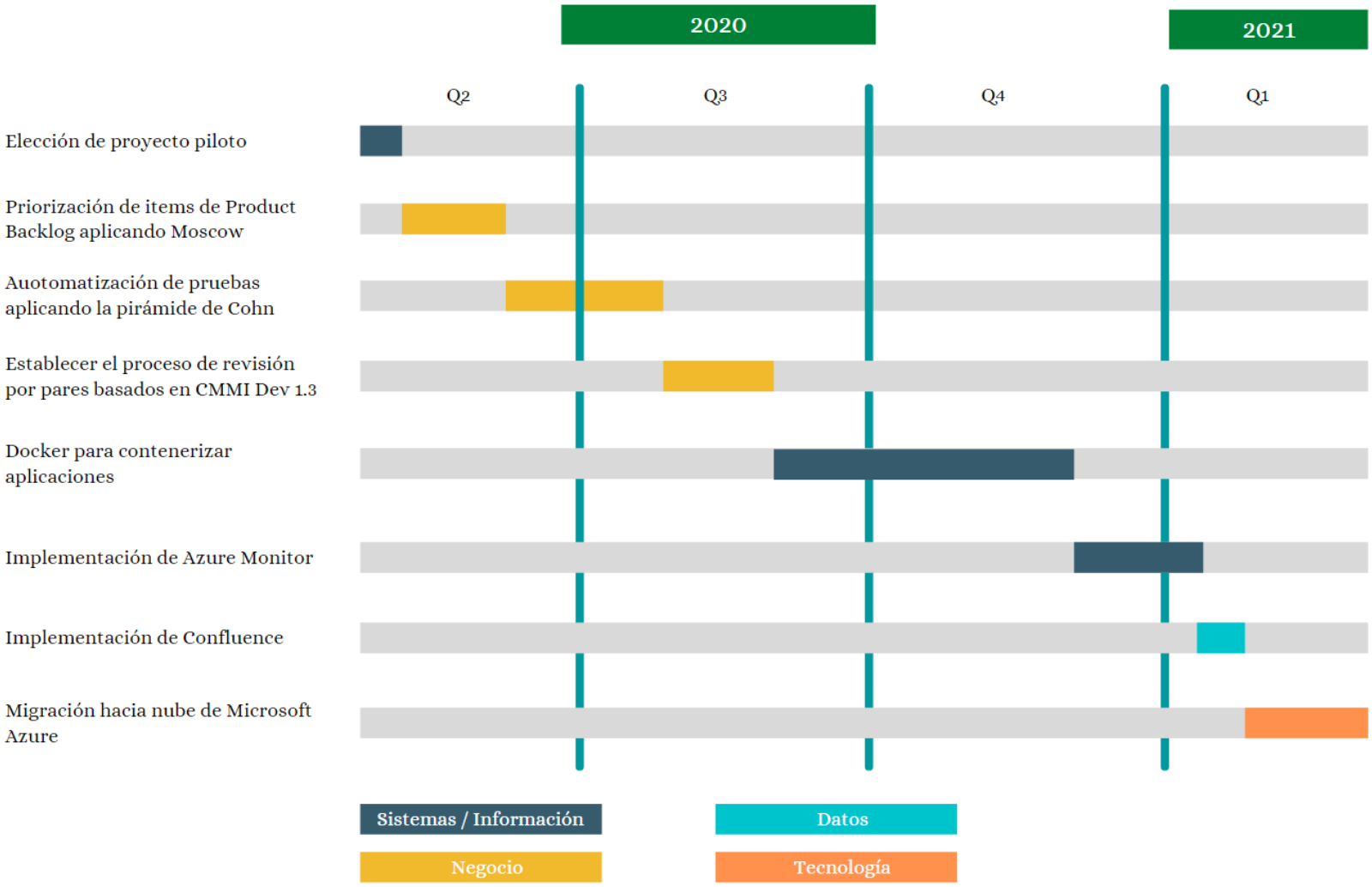
Monitoreo de aplicaciones

1	Madurez y Capacidad - Chat empresarial v1		Puntaje	3,1	4,5
			Puntaje Ponderado	3,1	4,6
ID	Dimensión	Descripción	Nivel de Importancia	Actual	Objetivo
1	Vigencia tecnológica	Nivel de incorporación de componentes tecnológicos modernos y aceptados por el mercado de soluciones de software y cumplimiento de estándares internacionales para usabilidad, robustez, portabilidad.	Muy Importante	Nivel 2	Nivel 5
2	Interoperabilidad	Capacidad de la aplicación para intercambiar información de manera automática con otras aplicaciones.	Muy Importante	Nivel 3	Nivel 5
3	Confiabilidad	Esquema de garantía que ofrece la aplicación para dar resultados precisos, un buen desempeño y consistencia de los datos.	Muy Importante	Nivel 3	Nivel 5
4	Arquitectura	Esquema de administración del conocimiento asociado a la aplicación, modelos arquitectonicos, diseños, guias de uso, administración actualizados.	Importante	Nivel 3	Nivel 3
5	Soporte	Esquema de apoyo/soporte con el que cuenta la aplicación para atender inquietudes de los usuarios, incidentes y requerimientos de operación.	Muy Importante	Nivel 3	Nivel 4
6	Seguridad	Nivel de protección con que cuenta la aplicación para contrarrestar los riesgos de vulnerabilidad que afecten la disponibilidad, integridad y confidencialidad de la información y los servicios tecnologicos que asiste.	Muy Importante	Nivel 3	Nivel 4
7	Flexibilidad/Mantenimiento	Capacidad de la aplicación para adaptarla a nuevos requerimientos en tiempos y costos razonables a los alcances del cambio, con la asistencia a versionamiento, documentación y marco de trabajo para los cambios.	Muy Importante	Nivel 4	Nivel 5
8	Gobernabilidad	Esquema de supervisión ejecutiva que se le da a la aplicación y aseguramiento de su alineamiento estrategico con las prioridades del negocio.	Muy Importante	Nivel 4	Nivel 5

Aplicaciones colaborativas

1	Madurez y Capacidad - Chat empresarial v1		Puntaje	2,1	3,3
			Puntaje Ponderado	2,4	3,5
ID	Dimensión	Descripción	Nivel de Importancia	Actual	Objetivo
1	Vigencia tecnológica	Nivel de incorporación de componentes tecnológicos modernos y aceptados por el mercado de soluciones de software y cumplimiento de estándares internacionales para usabilidad, robustez, portabilidad.	Muy Importante	Nivel 3	Nivel 4
2	Interoperabilidad	Capacidad de la aplicación para intercambiar información de manera automática con otras aplicaciones.	Importante	Nivel 2	Nivel 3
3	Confiabilidad	Esquema de garantía que ofrece la aplicación para dar resultados precisos, un buen desempeño y consistencia de los datos.	Importante	Nivel 3	Nivel 3
4	Arquitectura	Esquema de administración del conocimiento asociado a la aplicación, modelos arquitectonicos, diseños, guias de uso, administración actualizados.	Poco Importante	Nivel 1	Nivel 3
5	Soporte	Esquema de apoyo/soporte con el que cuenta la aplicación para atender inquietudes de los usuarios, incidentes y requerimientos de operación.	Importante	Nivel 2	Nivel 3
6	Seguridad	Nivel de protección con que cuenta la aplicación para contrarrestar los riesgos de vulnerabilidad que afecten la disponibilidad, integridad y confidencialidad de la información y los servicios tecnologicos que asiste.	Muy Importante	Nivel 3	Nivel 5
7	Flexibilidad/Mantenimiento	Capacidad de la aplicación para adaptarla a nuevos requerimientos en tiempos y costos razonables a los alcances del cambio, con la asistencia a versionamiento, documentación y marco de trabajo para los cambios.	Poco Importante	Nivel 1	Nivel 2
8	Gobernabilidad	Esquema de supervisión ejecutiva que se le da a la aplicación y aseguramiento de su alineamiento estrategico con las prioridades del negocio.	Importante	Nivel 2	Nivel 3

Anexo 3.


2020
2021
Q2
Q3
Q4
Q1
Elección de proyecto piloto
Priorización de items de Product Backlog aplicando Moscow
Auotomatización de pruebas aplicando la pirámide de Cohn
Establecer el proceso de revisión por pares basados en CMMI Dev 1.3
Docker para contenerizar aplicaciones
Implementación de Azure Monitor
Implementación de Confluence
Migración hacia nube de Microsoft Azure
Sistemas / Información
Datos
Negocio
Tecnología