MANUAL DE DESARROLLO DE SOFTWARE



SUPERINTENDENCIA DE SOCIEDADES

Bogotá, D.C., 05 de agosto de 2026

TABLA DE CONTENIDO

1. OBJETIVO 4
2. ALCANCE 4
3. RESPONSABLE 4
4. DEFINICIONES 4
5. CONTENIDO 5

5.1. ARQUITECTURA DE SISTEMAS DE INFORMACIÓN 8

5.2. REGLAS PARA EL DESARROLLO. 9

5.3. DIRECTRICES GENERALES. 11

5.3.1. Directrices de Seguridad 13

5.3.2. Directrices de Calidad 15

5.3.3. Directrices Administrativas. 17

5.4. NECESIDADES DE NEGOCIO – STAKEHOLDER MEETING. 21

5.4.1. Incidencias de sistemas de información 21

5.4.2. Mejoras a sistemas de información o Nuevas funcionalidades 23

5.5. LEVANTAMIENTO DE INFORMACIÓN SPRINT INCEPTION 25

5.5.1. Creación de los Requerimientos – Product Backlog. 25

5.5.2. Desarrollo – Construcción. 26

5.5.3. Reunión de Planificación – Sprint Planning Meeting. 27

5.5.4. Reuniones Diarias – Daily Scrum. 28

5.5.5. Entrega del trabajo realizado – Sprint Review 29

5.5.6. Ciclo de pruebas cuando el producto listo para producción 30

5.5.7. Excepciones a la aplicación de pruebas de seguridad para nuevas versiones o aplicaciones 32

5.5.8. Paso a producción 32

5.5.9. Permisos de acceso en Azure DevOps 33

5.6. REGISTRO VALORACIÓN, VIDA ÚTIL Y AMORTIZACIÓN DEL SOFTWARE 34

5.6.1. Valoración del Software 34

5.6.2. Asignación de la vida útil 35

5.6.3. Amortización del Software 37

5.6.4. Gestión de mejoras y nuevas funcionalidades 38

5.6.5. Revisión de la Vida Útil 39

6. ANEXOS Y REGISTROS 40

7. CONTROL DE CAMBIOS 40

1. OBJETIVO

Definir los pasos y lineamientos para el desarrollo seguro de sistemas de información en la Superintendencia de Sociedades.

2. ALCANCE

Va desde la definición de requerimientos hasta la puesta en operación de los productos de software desarrollados, nuevos desarrollos, ajustes a nivel técnico e integraciones entre sistemas de información.

3. RESPONSABLE

Grupo de Innovación, Desarrollo y Arquitectura de Aplicaciones (GIDAA)

4. DEFINICIONES

- **AGILE:** Es una filosofía que supone una forma distinta de trabajar y de organizarse. De tal forma que cada proyecto se 'trocea' en pequeñas partes que tienen que completarse y entregarse en pocas semanas. El objetivo es desarrollar productos y servicios de calidad que respondan a las necesidades de unos clientes cuyas prioridades cambian a una velocidad cada vez mayor.

- **ARQUITECTURA DE SISTEMAS DE INFORMACIÓN:** Es el dominio de la Arquitectura Empresarial que contiene los elementos para orientar a las entidades en la definición de la arquitectura de aplicaciones y que define; los componentes de los sistemas, las interacciones entre estos y la relación con las arquitecturas misional, de información y de infraestructura de TI.

- **METODOLOGÍAS AGILES DE DESARROLLO**: Las metodologías ágiles se centran en el factor humano y el producto software; es decir, ellas le dan mayor valor al individuo, a la colaboración del cliente y al desarrollo incremental del software con iteraciones muy cortas.
- **PRODUCT BACKLOG:** La Lista de Producto es una lista ordenada de todo lo que podría ser necesario en el producto, y es la única fuente de requisitos para cualquier cambio a realizarse en el producto. El Dueño de Producto es el responsable de la Lista de Producto, incluyendo su contenido, disponibilidad y ordenación.
- **HISTORIAS DE USUARIO:** técnica utilizada para especificar los requisitos del software. Se trata de formatos donde el cliente describe brevemente las características que el sistema debe poseer, sean requisitos funcionales o no funcionales. El tratamiento de las historias de usuario es muy dinámico y flexible. Cada historia de usuario es lo suficientemente comprensible y

delimitada para que los programadores puedan implementarla en unas semanas.

- **SCRUM:** Es un marco de trabajo por el cual las personas pueden abordar problemas complejos de forma adaptativa y a la vez entregar productos del máximo valor posible de forma productiva y creativamente. El marco de trabajo Scrum consiste en los Equipos Scrum, roles, eventos, artefactos y reglas asociadas. Cada componente dentro del marco de trabajo sirve a un propósito específico y es esencial para el éxito de Scrum y para su uso.
- **DEVOPS:** El término DevOps, que es una combinación de los términos ingleses development (desarrollo) y operations (operaciones), designa la unión de personas, procesos y tecnología para ofrecer valor a los clientes de forma constante. ¿Qué significa DevOps para los equipos? DevOps permite que los roles que antes estaban aislados (desarrollo, operaciones de TI, ingeniería de la calidad y seguridad) se coordinen y colaboren para producir productos mejores y más confiables. Al adoptar una cultura de DevOps junto con prácticas y herramientas de DevOps, los equipos adquieren la capacidad de responder mejor a las necesidades de los clientes, aumentar la confianza en las aplicaciones que crean y alcanzar los objetivos empresariales en menos tiempo.

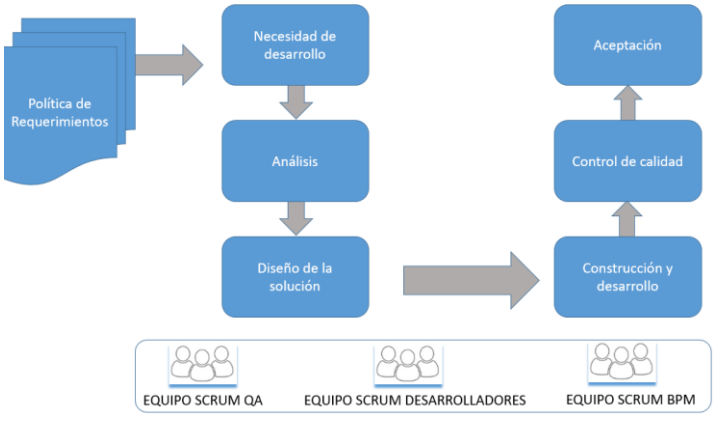
5. CONTENIDO

El foco de la metodología es tener en cuenta el cliente, como generarle valor a través de la operación tecnológica basada en estándares de calidad y de la implementación de proyectos de innovación alineando sus esfuerzos hacia el cumplimiento de la misión y objetivos institucionales. El principal reto es identificar cómo el cliente está percibiendo el servicio que estamos entregando, por lo que consideramos como principio guía trabajar con la fábrica interna enfocados hacia la mejora continua, teniendo como referencia el ciclo PHVA (Planear, Hacer, Verificar, Actuar).

Nos guiaremos por el marco de trabajo SCRUM y la herramienta Azure DevOps de Microsoft, con el propósito de obtener resultados incrementales en tiempos cortos para atender incidentes provenientes del soporte, mantenimiento evolutivo y nuevas necesidades, así como en la madurez del equipo de trabajo, siguiendo el modelo de Tuckam. A continuación, mostramos el esquema de alto nivel de trabajo con la fábrica.



Fuente: *https://softwareblog03.wordpress.com/2017/04/24/modelos-de-ciclo-de-vida-del-software/*



La Superintendencia de Sociedades cuenta con la herramienta de Azure Devops, que proporciona una gestión del ciclo de vida de la aplicación, a través de la administración del código fuente, informes, administración de requisitos, compilaciones automatizadas, pruebas y administración de versiones, desde el año 2019 se impulsó el uso de la herramienta indicando a los proveedores que

desarrollaban software o realizaban mejoras sobre las aplicaciones existentes incluir el código fuente en esta herramienta y dejarlo a disposición para futuros proyectos.

De los módulos que ofrece la herramienta Azure Devops, para la gestión del ciclo de desarrollo de software, en la Superintendencia de Sociedades se usan:

Azure Pipelines: CI que funciona con cualquier lenguaje, plataforma y nube. Conéctese a GitHub o a cualquier otro repositorio de Git y lleve a cabo implementaciones continuas.

Azure Board: Realice un seguimiento eficaz del trabajo con paneles kanban, trabajos pendientes, paneles de equipo e informes personalizados.

Azure Artifacts: Los paquetes de Maven, npm y NuGet se alimentan de orígenes públicos y privados.

Azure Repo: Repositorios privados de Git ilimitados, hospedados en la nube para su proyecto. Solicitudes de incorporación de cambios colaborativas, administración avanzada de archivos y mucho más.

Azure Test Plans: Permite generar los planes de pruebas para cada proyecto; ejecutar y documentar los casos de prueba.

El equipo de fábrica interno creado en abril de 2020, hace uso de esta herramienta, conformada por diferentes componentes que permiten implementar una estrategia de trabajo consolidado facilitando seguimientos y controles de cambio, basados en los lineamientos del marco SCRUM.

Esta herramienta nos permite trabajar con prácticas de DevOps como:

Integración (CI)

La integración continua es una práctica de desarrollo de software en la que los desarrolladores fusionan mediante combinación los cambios de código en la rama de código principal con frecuencia. En la integración continua se utilizan pruebas automáticas, que se ejecutan cada vez que se hace "commit" de código nuevo. De este modo, el código de la rama principal siempre es estable.

Control de versiones

Control de versiones es la práctica de administrar el código por versiones, haciendo un seguimiento de las revisiones y del historial de cambios para facilitar la revisión y la recuperación del código.

Desarrollo ágil de software

Los equipos que practican la metodología ágil proporcionan mejoras y cambios continuos a los clientes, recopilan sus comentarios y, después, aprenden y ajustan el software en función de lo que el cliente quiere y necesita.

Infraestructura como código

La infraestructura cómo código define las topologías y los recursos del sistema de un modo descriptivo que permite a los equipos administrar esos recursos igual que lo harían con el código. Las diferentes versiones de esas definiciones se pueden almacenar en sistemas de control de versiones, donde se pueden revisar y revertir, de nuevo, igual que el código.

Administración de configuración

Administración de la configuración hace referencia a la administración del estado de los recursos de un sistema, incluidos los servidores, las máquinas virtuales y las bases de datos. El uso de herramientas de administración de la configuración permite a los equipos distribuir cambios de un modo controlado y sistemático, lo que reduce el riesgo de modificar la configuración del sistema.

Supervisión continua

Supervisión continua significa tener visibilidad total y en tiempo real del rendimiento y el estado de toda la pila de aplicaciones, desde la infraestructura subyacente donde se ejecutan las aplicaciones hasta los componentes de software de niveles superiores.

El esquema scrum diseñado como metodología de trabajo con la fábrica de software interna se muestra a continuación:

5.1. ARQUITECTURA DE SISTEMAS DE INFORMACIÓN

Como referencia para las definiciones de la Dirección de Tecnologías de la Información y las Comunicaciones (DTIC), se toma como base todas las directrices del área y de la Entidad, como columna principal para el Grupo de Innovación, Desarrollo y Arquitectura de Aplicaciones (GIDAA) se debe contar con los diagramas de arquitectura de las aplicaciones y sistemas de Información, y la construcción y/o actualización del documento GTI-FM-027 Arquitectura de Software.

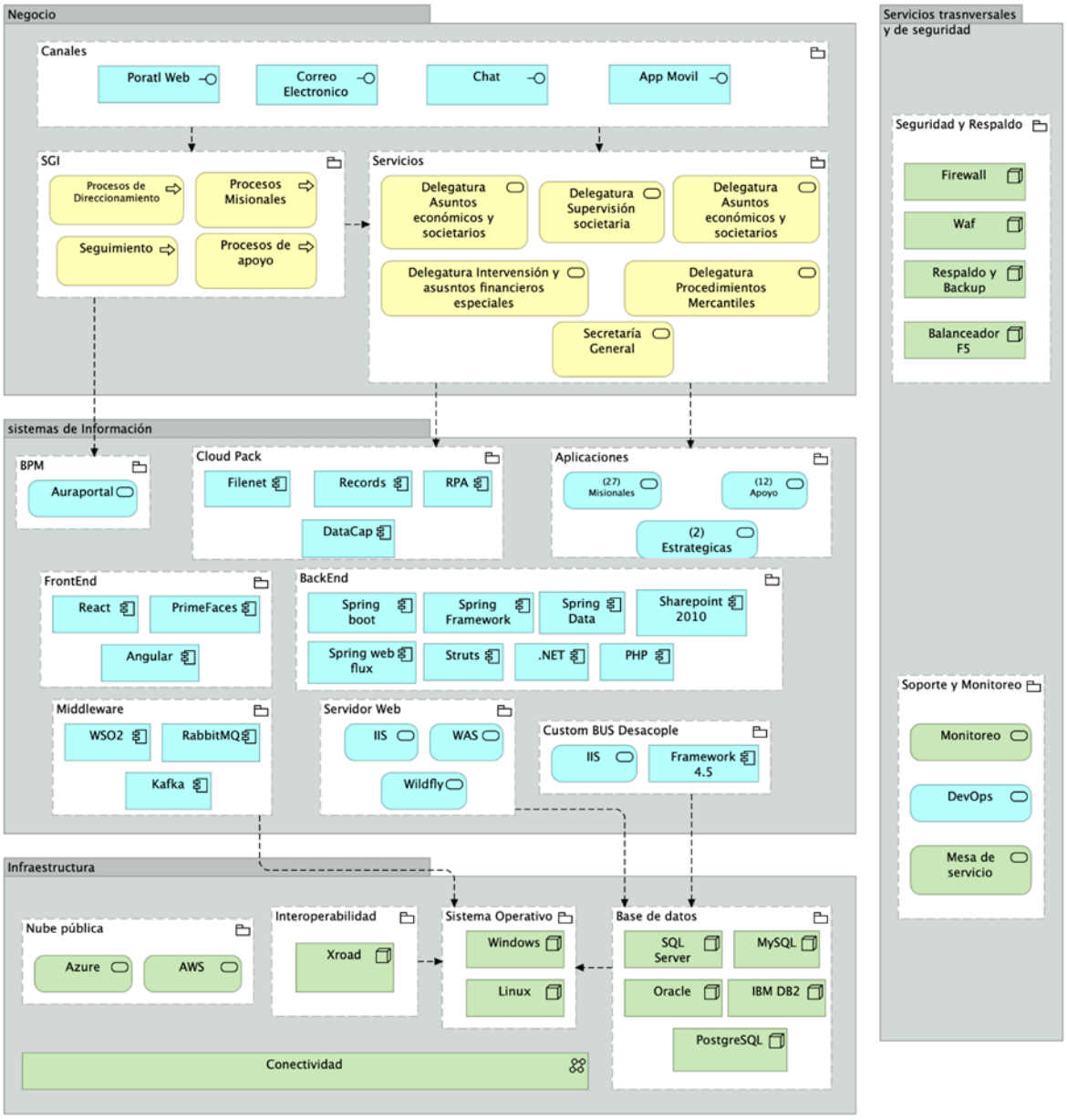


Imagen 1- Arquitectura de tecnologías para sistemas de información

5.2. REGLAS PARA EL DESARROLLO.

A continuación, se relacionan las principales reglas para el desarrollo de Software in house on mediante contratación con un tercero.

Sin embargo, debe complementarse con lo descrito en la guía GTI-GU-020 Lineamientos de Arquitectura y Desarrollo de Software

5.1.1. Capa de Presentación

La capa de presentación (FRONTEND) para los nuevos sistemas o mejoras a sistemas anteriores debe ser web con patrón MVC/Angular/React/NetCore/Java, C#, JQuery y Bootstrap.

5.1.2. Capa de Servicios

La capa de conexión con datos internos o servicios externos (BACKEND) solo puede acceder a los datos de las bases de datos o a la información de otras aplicaciones por medio de web services, estos web services se integran por los componentes del proyecto Desacople (WebApi e Interoperabilidad) y microservicios los cuales cuentan conconfiguración para acceso por SOAP y REST. Estos servicios son alojados en el repositorio de azure Devops por lo que todos los componentes deben ser compatibles con este tipo de lógicas.

5.1.3. Estándar en el nombramiento de objetos de bases de datos

Para todo lo relacionado a nombramiento de objetos y artefactos en base de datos, se debe tener en cuenta lo descrito en el documento GTI-MO-001-Modelo de Gobierno de Datos de la Superintendencia de Sociedades

5.1.4. Capa de la Lógica del Negocio (BLL) – Core:

La lógica de negocio no se deja en las bases de datos, por lo que esta capa debe contener todas las trasformaciones, procesos y algoritmos particulares del negocio que permitan interpretar la información para el usuario final.

De acuerdo con la necesidad de la lógica de la transacción en proceso se puede necesitar acceder a:

a. Los API de integración basados en protocolos SOAP y REST: este permite acceder mediante servicios web a:
    a. La base de datos del Sistema de Información General de Sociedades (SIGS) que es la Base de datos principal de la Superintendencia de Sociedades. Esta base de datos contiene en su mayoría todas las tablas del negocio.
    b. Integrar con el sistema de Gestión documental
    c. Conectar la autenticación y control de acceso del sistema de información al Directorio Activo
    d. Envío de mensajería por servicio web

b. Web: Los webservices que pertenecen a los sistemas CORE de la entidad.

5.1.5. API acceso a Datos:

Sobre esta capa se realiza la invocación de consultas y ejecución de procedimientos almacenados sobre las bases de datos de la Entidad.

5.1.6. Servicios de Acceso a Datos:

Esta capa representa los servicios que son expuestos por otras aplicaciones de terceros y/o internas.

5.1.7. Data Sources – CRUD:

Esta capa representa las bases de datos de la organización y/o las bases de datos de los sistemas de terceros.

5.1.8. Capa de Modelo:

Esta capa contiene las clases de los objetos de bases de datos o de lógica de negocio que es utilizado por los sistemas, el objetivo es disminuir la creación de clases de objetos y dejarlo consolidado en esta capa

5.1.9. Conceptos Transversales:

Esta capa contiene objetos de seguridad, administración, comunicación y notificación transversal a las demás capas con el fin de no reprocesar y/o duplicar funcionalidades.

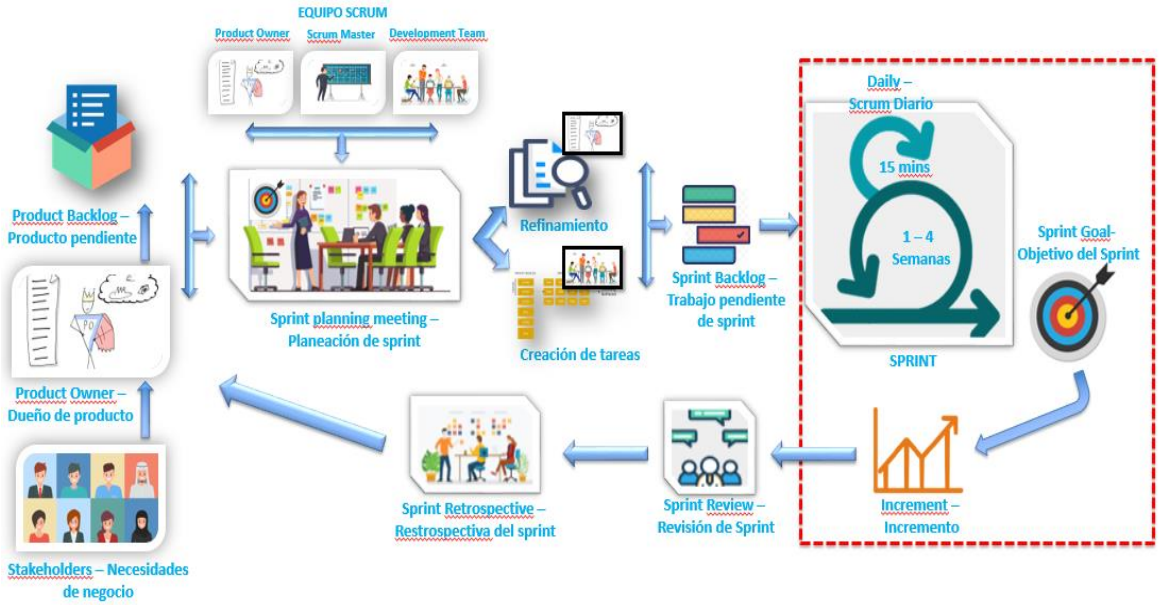
NOTA:

Cualquier cambio a la arquitectura debe ser consultado y aprobada por los líderes de los grupos de tecnología.

5.3. DIRECTRICES GENERALES.

- Las aplicaciones no deben conectarse directamente a bases de datos que no sean las propias, en caso de que una aplicación requiera integración con otra lo debe hacer obligatoriamente mediante la capa de interoperabilidad de la Superintendencia de Sociedades
- Los desarrollos de aplicaciones internas por parte de la Superintendencia de Sociedades o por un proveedor contratado para desarrollos, deben ser construidos con tecnología Microsoft y/o java y librerías compatibles que estén autorizadas por el equipo de tecnología (MVC/Angular/React/NetCore/Java, C#, JQuery y Bootstrap).
- En los desarrollos y adquisiciones de sistemas de información se debe contemplar que el estándar de bases de datos a utilizar es SQL.

- Para el caso del proyecto de Inteligencia Artificial, el proveedor es del segmento AWS, lo cual implica que sus desarrollos y herramientas están soportadas en herramientas de esta nube.
  - Los artefactos de bases de datos se deben dejar en DevOps.
  - Estructura de las carpetas
    - ScriptSQL
      - Funciones
      - Procedimientos
      - Tablas
- Los despliegues hacia el ambiente de pruebas se realizarán mediante solicitud del desarrollador responsable al funcionario encargado de la gestión de despliegues, para esto deberá confirmar la ejecución del merge de la rama a desplegar junto con las modificaciones necesarias sobre los archivos de configuración. al finalizar cada desarrollo y al confirmar un paso a producción, el desarrollador debe cargar en Devops el código fuente, compilado con los cambios de la sincronización de los demás desarrolladores de los equipos de trabajo.
- Todos los desarrollos deben pasar obligatoriamente por los ambientes de desarrollo y pruebas y debe dejarse resultado de las pruebas en DevOps.
- Se debe aplicar el conducto regular para comunicación con los usuarios, para lo cual todos los desarrollos deben iniciar con una historia de usuario (GTI-FM-021 Formato Historia de Usuario) que debe ser creada por el equipo de requerimiento y aprobada por el usuario funcional. Y
- Para los desarrollos de sistemas de información nuevos, se debe construir el documento formato GTI-FM-028 Especificación Requisitos.
- Para los desarrollos contratados se deben seguir estas directrices y lineamientos de desarrollo y deben quedar anexos al Estudio de conveniencia que se elabore, así como ser parte integral del contrato, como un anexo técnico.
- Se debe seguir y cumplir el flujo de fábrica de Software de la Superintendencia de Sociedades, el cual se presenta a continuación:



[2] Fuente: Microsoft Azure, (2020). Recuperado de: https://azure.microsoft.com/es-es/overview/what-is-devops/#practices

5.3.1. Directrices de Seguridad

- Todas las aplicaciones que se desarrollen o adquieran deben tener autenticación con LDAP o Registro de usuarios de la Superintendencia de Sociedades. (Validación contra el Directorio Activo).
- Todas las aplicaciones que se desarrollen o adquieran deben contar con un módulo de seguridad que contemple la gestión de usuarios, la gestión de contraseñas y los roles y perfiles de acceso a la información.
- El acceso en ambiente de producción (aplicaciones, servicios, bases de datos) solo lo pueden tener los funcionarios autorizados del Grupo GSAT o los que se determinen por parte de este Grupo.
- Los desarrolladores solo pueden ingresar a los ambientes de pruebas y desarrollo.
- Los usuarios y contraseñas para conexión de servicios, aplicaciones o bases de datos deben ser robustas y de ser posible usar un mecanismo de cifrado, con el fin de asegurar la confidencialidad de la información y deben ser diferentes a los empleados en ambientes de desarrollo y pruebas.
- Los desarrollos que impliquen cambios a datos deben cumplir el concepto de transacción, con el fin de asegurar que en caso de una falla en la ejecución del código los datos volverán a su valor inicial, con el fin de preservar la integridad de la información.

- Se deben ejecutar pruebas de vulnerabilidad a las aplicaciones antes de salir a producción, esto con el fin de mitigar y corregir defectos asociados con la seguridad de la información; los hallazgos de dichas pruebas deben estar en estado medio o bajo de riesgo; los de riesgo extremo y alto deben ser remediadas antes de su ingreso a producción.
- Se deben tener en cuenta las buenas prácticas de desarrollo para código seguro; se puede tomar como referencia la guía OWASP, en su última versión y las condiciones establecidas en el Anexo 3 de la Resolución 1519 de 2020 de Ministerio de tecnología de la información y las comunicaciones.
- Cada grupo de desarrollo debe contar con un set de datos en sus ambientes de desarrollo y pruebas, con el fin de no afectar la data de ambiente de producción.
- Se deben ejecutar pruebas de vulnerabilidad y de Ethical Hacking de forma periódica sobre las aplicaciones, con el fin de mitigar vulnerabilidades.
- En lo posible, se debe desarrollar módulos de consulta y monitoreo de actividades anómalas o diferentes a los estándares de operación.
- Se deben realizar pruebas de carga a las aplicaciones, acorde con la cantidad de usuarios y concurrencia esperada.
- Las aplicaciones deben contar con unos mínimos de registro de auditoría que permitan identificar actividades de acceso y salida al sistema de información, bases de datos y actualización (creación, modificación, borrado, consulta). Los registros de auditoria deberían llevar al menos los siguientes datos:
  - Identificación del usuario.
  - Actividades del sistema.
  - Fechas, horas y detalles de los eventos clave, por ejemplo, entrada, salida, modificaciones, borrados.
  - Identidad del dispositivo o ubicación, si es posible, e identificador del sistema.
  - Registros de intentos exitosos y rechazados, de acceso a los módulos del sistema.
  - Registros de otros intentos de acceso a recursos del sistema.
  - Cambios a la configuración y parámetros del sistema.

- Las aplicaciones que finalicen su ciclo de vida deben ser retiradas de los ambientes de desarrollo, pruebas y producción, conservando un backup o respaldo previo de todos sus componentes, datos y documentación de acuerdo con los establecido en el procedimiento GTI-PR-015 baja de aplicativos y de software en desuso.
- Las aplicaciones se deben implementar en modelo de alta disponibilidad, que permita hacer un manejo de concurrencia y pruebas ante fallas por indisponibilidad de la plataforma.
- En el desarrollo de aplicaciones o en las contratadas externamente, se debe tener en cuenta los estándares y Protocolos de comunicación y conexión seguros (HTTPS, FTPS, SSL, SSH, TLS, VPN´s, entre otros).

- Se deben priorizar en los desarrollos de nuevas aplicaciones o en las adquiridas externamente mediante acceso web a través de los navegadores más comunes del mercado y debe incluirse mecanismo de seguridad como certificado digital de sitio seguro y en los casos que aplique autenticación integrada con el Directorio Activo o el sistema que se establezca en la Entidad, así como una estructura de roles / perfiles para asegurar la confidencialidad de la información.
- El modelo de Software como Servicio, también debe ser considerado dentro de los proyectos de sistemas de información, como un mecanismo y opción de implementación y despliegue de los sistemas.
- Se deben establecer los mecanismos de respaldo de aplicaciones, código fuente, datos de prueba acorde con la dinámica en la construcción de software, esto con el fin de mitigar eventos de pérdida o daño de información y que la recuperación sea lo más rápida posible y con la menor cantidad de pérdida de información.
- Se debe contemplar la integración con los servicios ciudadanos digitales establecidos por la política de Gobierno Digital – MinTIC (interoperabilidad con otras Entidades a través de x-road, autenticación digital y carpeta ciudadana).

5.3.2. Directrices de Calidad

- Se debe asegurar la implementación de ambientes de desarrollo, pruebas y producción para todas las aplicaciones desarrolladas por la Entidad, bien sea a través de su fábrica de desarrollo o contratos específicos para este fin. Los desarrolladores solo pueden ingresar a los ambientes de pruebas y desarrollo.
- Todas las nuevas versiones de aplicaciones deben contar con una documentación mínima y un procedimiento de control de cambios: archivo léame, con la descripción de los cambios en la versión, número o consecutivo de la versión del reléase o versión de la aplicación, formato de control de cambios, scripts de bases de datos, versión de la aplicación o de los servicios.
- Toda la documentación de las aplicaciones: historias de usuario, manuales, diagramas de diseño, arquitectura, entre otros deben estar publicados en la herramienta DevOps.
- En el momento de levantamiento de información se debe solicitar a los usuarios la definición de los momentos de respaldo y restauración de información y los periodos de retención de datos.
- El código fuente debe ser documentado continuamente, esto con el fin de facilitar el mantenimiento y entendimiento de las aplicaciones.
- En el momento de entrega de un sistema de información desarrollado internamente o adquirido, se debe contemplar la entrega de los manuales: técnicos, funcionales, de operación, de usuario, de respaldo de información.
- Las nuevas versiones de aplicaciones deben cumplir como mínimo las siguientes pruebas antes de salir a producción: pruebas unitarias, pruebas funciones, pruebas

integrales con el usuario final; en el caso de los defectos, deben cumplir las pruebas unitarias por parte del desarrollador y las integrales por parte de los Analistas de Negocio – Requerimientos.

- En el desarrollo o adquisición de nuevas aplicaciones, sistemas y adecuaciones, se deben contemplar los siguientes aspectos de calidad:

  - **Alta disponibilidad**: Tolerancia a fallos y sus estrategias de recuperación de forma autónoma.
  - **Balance o balanceo de carga:** Capacidad que deben soportar los servidores Web a nivel transaccional.
  - **Procedimiento de despliegue y gestión de versiones:** Manejo de versiones y la configuración de componentes.
  - **Movilidad:** Interoperabilidad con terminales móviles y funcionamiento "off line" (Fuera de línea).
  - **Confiabilidad:** Capacidad del sistema de realizar las funciones para las que fue diseñado sin presentar fallos, la información registrada es almacenada en la BD para su posterior consulta y se mantiene la integridad en la base de datos y en el flujo de información.
  - **Rendimiento:** Los nuevos desarrollo o adquisiciones de sistemas y aplicaciones debe contemplar los siguientes tiempos:

    - El tiempo de carga de la aplicación y sus módulos es aceptable <4 seg.
    - El tiempo de procesamiento de la aplicación (funciones, cálculos, importación, exportación) es aceptable <5seg,
    - El tiempo de respuesta de una consulta es aceptable <6 seg,
    - Tiempos de respuesta establecidos con las áreas usuarias de la aplicación.

  - **Disponibilidad:** La disponibilidad del sistema debe responder al 100% de las necesidades de los usuarios 24 horas al día, 7 días a la semana y se debe realizar monitoreo de funcionamiento de la aplicación en línea, así como un plan de contingencia en caso de materialización de eventos inesperados que dejen indisponible el aplicativo y/o infraestructura tecnológica, las bases de datos o las comunicaciones.
  - **Portabilidad:** Migración a otra plataforma tecnológica sin afectar la operación del día a día de la Entidad, Soporta diferentes sistemas operativos, diferentes navegadores y funcionalidad de multilenguaje.
  - **Accesibilidad:** Las opciones (menús y campos) deben ser accesibles mediante el apuntador del mouse y/o teclado, el contenido es "responsive" frente a los dispositivos, el acceso interno y externo soporta el estándar W3C (World Wide Web Consortium).

- **Usabilidad y reusabilidad**: Que las opciones y componentes o funcionalidades del sistema puedan suministrar servicios a otros sistemas, deben aparecer los logos y créditos de la Superintendencia en la aplicación, se debe relacionar la información de soporte: teléfonos, correos y equipo de soporte, presenta la pertenencia legal (copyright) del producto, hacer de fácil entendimiento la aplicación para el usuario, diseñar pocos pasos para que el usuario pueda efectuar una acción específica o se deben presentar las opciones de la herramienta de una forma práctica, los mensajes de error presentados son claros y consistentes y siguen el lenguaje de los usuarios, se debe tener acceso a opciones de ayuda o FAQs en la herramienta, se deben se establecer controles sobre los campos a diligenciar por parte del usuario (fechas, texto, valores numéricos).
- **Interfaz con usuario.** Se debe desarrollar teniendo en cuenta:

  - Interoperabilidad con otros sistemas/recursos de información.
  - Utilización del lenguaje propio de los usuarios.
  - La información relacionada está organizada y agrupada.
  - Mantener el estándar de los colores institucionales.
  - Utilizar diferentes tamaños de letra y color según lo establecido institucionalmente.
  - Utilizar iconos claros y que permitan facilidad para acceder a las funciones.
  - Menús accesibles y con una secuencia lógica.
  - Funcionamiento correcto de enlaces y botones.
  - El contenido está correctamente escrito (redacción y ortografía)
  - Mensajes de error claros y sencillos con indicaciones de ayuda.

- **Integración:** Tener en cuenta la integración con herramientas o aplicaciones existentes, con diferentes orígenes de información, con los servicios del portal institucional, con productos propietario (proveedores), con herramientas de mensajería, con servidores de aplicaciones, con servicios multimedia (streaming), con sistemas de intercambio de información.

5.3.3. Directrices Administrativas.

- Para el caso de aplicaciones de terceros, por lo menos se debe contar con ambiente de pruebas y el de producción.
- Para los nuevos desarrollo o adquisiciones se debe contemplar el plan de capacitación, teniendo en cuenta:
  - Manuales y videos de funcionamiento de la aplicación.
  - Plan de sensibilización.
  - Evidencias de resultados de sensibilización.

  - Sensibilización usuarios final, usuario técnico, usuario Administrador, administrador bases de datos.
- Los nuevos desarrollos o adquisiciones de aplicaciones o sistemas de información deben contener internamente, cifras de uso y apropiación de sus módulos y manejo de información, de tal manera que se pueda monitorear si los usuarios obtienen provecho de los nuevos sistemas desarrollados o adquiridos.
- Para el desarrollo o adquisición de sistemas de información hay que tener en cuenta:
  - El entorno de hardware actual (tipos de servidores, bases de datos, redes).
  - El entorno de Software actual (sistemas operativos, herramientas informáticas y su licenciamiento).
- Para desarrollos o adquisiciones de sistemas de información en la nube se debe tener en cuenta la existencia de:
  - Uso de firewall.
  - Uso de WAF.
  - Cifrado en autenticación.
  - Cifrado en tránsito.
  - Conexiones y protocolos seguros.
  - Antivirus/Antimalware.
  - Monitoreo de actividades de los usuarios.
  - Monitoreo usuarios administradores.
  - Monitoreo de actividad de bases de datos (DAM Data Activity Monitoring).
  - Monitoreo de actividad en archivos (FAM - File Activity Monitoring).
  - Monitoreo de migración de datos con filtros URL y herramientas Data Loss Prevention.
  - Protección de datos en tránsito (Cifrado Cliente/Aplicación, Cifrado Enlace/Red, Cifrado basado en Proxy).
  - Alta disponibilidad.
  - Trazabilidad y Auditoria.
  - Respaldo de datos.
  - Gestión de usuarios y de identidad.
  - Contraseñas seguras.
  - Pruebas de vulnerabilidad
- En los desarrollos o adquisiciones de sistemas y aplicaciones se debe contemplar lo correspondiente a la ley de protección de datos (ley 1581 de 2012).
- Para pruebas se puede utilizar datos de producción despersonalizados.
- Todo desarrollo o adquisición de sistemas de información debe contar con análisis de riesgos, criterios de calidad, monitoreo de rendimiento y análisis de capacidad de infraestructura utilizada para funcionamiento.

Seguir los lineamientos establecidos por MinTIC en el Marco de Referencia de Arquitectura Empresarial (MRAE) del Estado colombiano frente al Dominio de Sistemas de Información, en su versión más reciente. - articles-237650_recurso_1.pdf

LINEAMIENTO	NOMBRE	DESCRIPCIÓN
MGGTI.LI.SI.01	Metodología para el desarrollo de sistemas de información	La dirección de Tecnologías y Sistemas de la Información o quien haga sus veces debe adoptar y personalizar una metodología alineada a las mejores prácticas para el desarrollo y mantenimiento de software, que oriente los proyectos de construcción o evolución de los sistemas de información que se desarrollen internamente o a través de terceros.
MGGTI.LI.SI.02	Catálogo de Sistemas de Información	La dirección de tecnologías y Sistemas de la información o quien haga sus veces debe garantizar la construcción y gestión del catálogo de sistemas de información, el cual integra la caracterización de cada uno de sus sistemas de información. Las entidades cabeza de sector adicionalmente deben consolidar y mantener actualizado el catálogo de sistemas de información sectorial.
MGGTI.LI.SI.03	Guía de estilo y usabilidad	La dirección de tecnologías y Sistemas de la información o quien haga sus veces debe definir, adoptar y/o adaptar una guía de estilo y usabilidad para la institución. Esta guía debe incorporar los lineamientos de gov.co y elementos de accesibilidad web.
MGGTI.LI.SI.04	Ambientes independientes en el ciclo de vida de los sistemas de información	La dirección de Tecnologías y Sistemas de la Información o quien haga sus veces debe implementar y mantener ambientes independientes (desarrollo, pruebas, producción) durante el ciclo de vida de los sistemas de información.
MGGTI.LI.SI.05	Análisis de requerimientos de los sistemas de información	La dirección de Tecnologías y Sistemas de la Información o quien haga sus veces debe incorporar dentro de sus procesos actividades formales de análisis y gestión de requerimientos de software en el ciclo de vida de los sistemas de información de manera que se garantice su trazabilidad y cumplimiento.
MGGTI.LI.SI.06	Integración, entrega y despliegue continuo durante el ciclo de vida de los sistemas de información	La dirección de Tecnologías y Sistemas de la Información o quien haga sus veces debe implementar dentro del proceso de desarrollo de sistemas de información, estrategias de integración, entrega y despliegue continuo en las actividades de desarrollos de sistemas de información.

LINEAMIENTO	NOMBRE	DESCRIPCIÓN
MGGTI.LI.SI.07	Plan de pruebas durante el ciclo de vida de los sistemas de información	La dirección de Tecnologías y Sistemas de la Información o quien haga sus veces debe estructurar un plan de pruebas que cubra aspectos funcionales y no funcionales sobre los nuevos desarrollos y mantenimientos evolutivos de los sistemas e información. La aceptación de cada una de las etapas de este plan debe estar vinculada a la transición del sistema de información a través de los diferentes ambientes.
MGGTI.LI.SI.08	Manual del usuario, técnico y de operación de los sistemas de información	La dirección de Tecnologías y Sistemas de la Información o quien haga sus veces debe asegurar que todos sus sistemas de información cuenten con la documentación técnica y funcional debidamente actualizada.
MGGTI.LI.SI.09	Plan de mantenimiento de los sistemas de información	La dirección de Tecnologías y Sistemas de la Información o quien haga sus veces debe elaborar un plan de mantenimiento anual de los sistemas de información de la Entidad.
MGGTI.LI.SI.10	Servicios de mantenimiento de sistemas de información con terceras partes	La dirección de Tecnologías y Sistemas de la Información o quien haga sus veces debe establecer criterios de aceptación y definir Acuerdos de Nivel de Servicio (ANS) cuando se tenga contratado con terceros el mantenimiento de los sistemas de información. Los ANS se deben aplicar en las etapas del ciclo de vida de los sistemas de Información que así lo requieran y se debe velar por la continuidad del servicio.
MGGTI.LI.SI.11	Plan de calidad de los sistemas de información	La dirección de Tecnologías y Sistemas de la Información o quien haga sus veces debe implementar un plan de aseguramiento de la calidad que contemple con claridad criterios de aceptación que generen valor durante el ciclo de vida de los sistemas de información.
MGGTI.LI.SI.12	Requerimientos no funcionales o atributos calidad de los sistemas de Información	La Dirección de tecnologías y Sistemas de la Información o quien haga sus veces, para la construcción o evolución de los Sistemas de Información, debe identificar los requerimientos no funcionales aplicables asociados a los atributos de calidad, garantizando su cumplimiento una vez entre en operación el sistema.
MGGTI.LI.SI.13	Accesibilidad	La Dirección de tecnologías y Sistemas de la Información o quien haga sus veces, debe garantizar que los sistemas de información y portales web que estén disponibles para el acceso a la ciudadanía o aquellos que de acuerdo con la caracterización de usuarios lo requieran, deben cumplir con las características de accesibilidad web.

LINEAMIENTO	NOMBRE	DESCRIPCIÓN
MGGTI.LI.SI.14	Arquitectura de Software	La dirección de tecnologías y Sistemas de la Información o quien haga sus veces debe definir, documentar y mantener actualizadas las arquitecturas de software de cada sistema de información de la Entidad.

Lineamientos y evidencias del Dominio de Sistemas de Información del MGGTI

5.4. NECESIDADES DE NEGOCIO – STAKEHOLDER MEETING.

Las necesidades de negocios se desprenden del trabajo de la organización apoyada en los servicios que ofrece TI los cuales son plasmados en requerimientos o requisitos.

Las necesidades del negocio se dan por incidentes, mejoras o nuevas funcionalidades; la solicitud o requerimiento realizado se debe registrar de manera completa, clara y posible con todos los antecedentes, mediante la herramienta Devops, a través del formulario 46002 o mediante correo electrónico.



Cada aplicación de la Supersociedades se ingresa como proyecto en la herramienta Azure DevOps, con el fin de administrar y documentar el ciclo de desarrollo.

5.4.1. Incidencias de sistemas de información

Se refieren a funcionamientos incorrectos de las aplicaciones, los incidentes o defectos serán documentados como Bugs en la herramienta Azure DevOps, en el proyecto asociado al aplicativo:



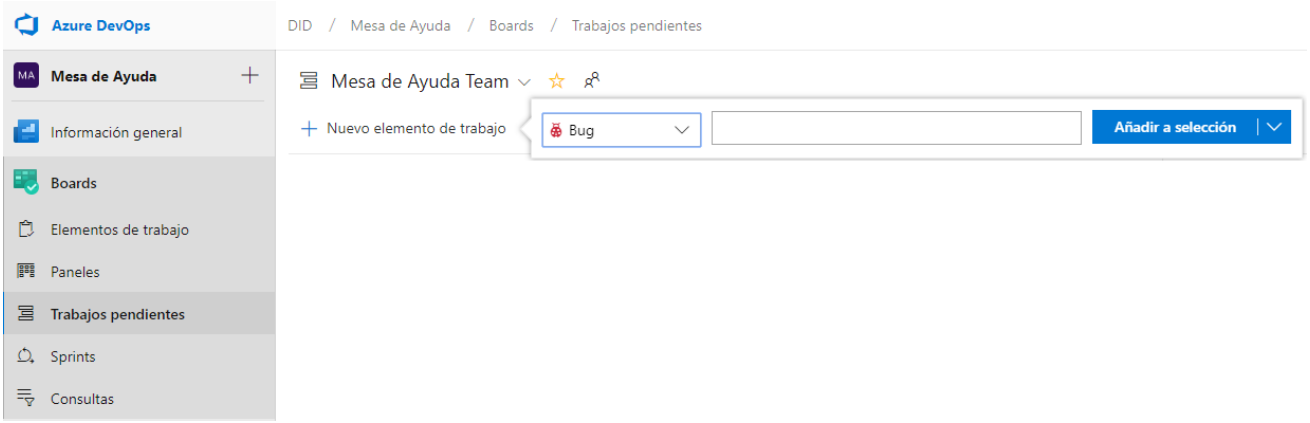


Un defecto o incidente puede ser identificado en dos momentos:

- **Ciclo de pruebas:** Aplica cuando se identifica un comportamiento que no se encuentra alineado a la definición entregada al equipo de desarrollo, y es identificada en las pruebas funcionales o integrales y se devuelve a desarrollo para que sea atendido.
- **En producción:** Aplica cuando un usuario de la SuperSociedades identifica un error o defecto de software en algún aplicativo de su uso diario. En este caso, debe escalar el hallazgo a través del portal de servicios tecnológicos manejado por la mesa de ayuda, la cual tiene los siguientes niveles de escalamiento:



Por cualquiera de las situaciones indicadas se debe registrar y gestionar a través de la herramienta Azure DevOps creando el bug en la sección Backlog o Trabajo pendiente.

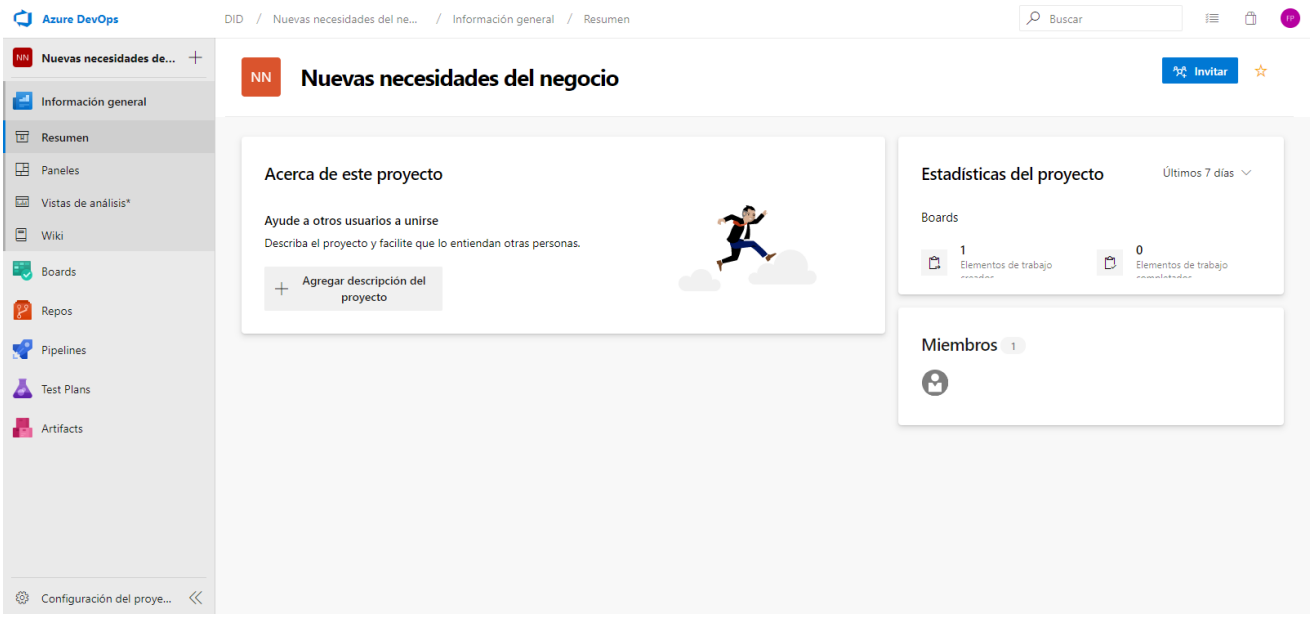


El soporte que no es atendido por mesa de ayuda, lo realiza el grupo de Innovación, Desarrollo y Arquitectura de Aplicaciones (GIDAA), el cual está compuesto por ingenieros de requerimientos y de desarrollo, que representan el grado de escalamiento más alto dentro de la organización de soporte.

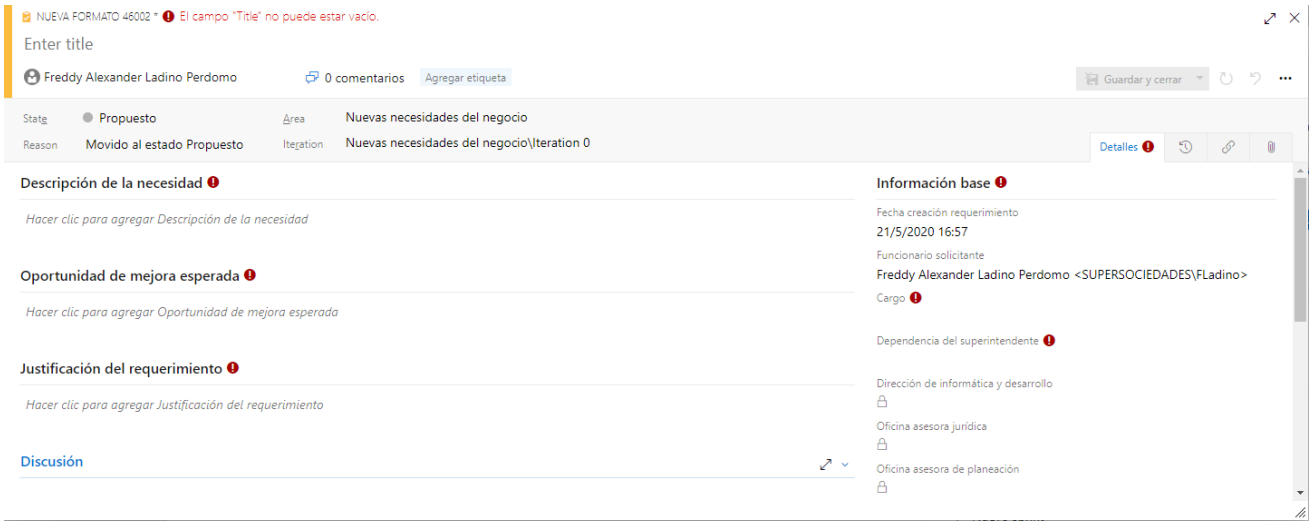
5.4.2. Mejoras a sistemas de información o Nuevas funcionalidades

Las mejoras o nuevas funcionalidades serán atendidas siguiendo las etapas del ciclo de desarrollo de software y dando cumplimiento al procedimiento GTI-PR-003_ImplementacionSistemasInformacion

Estos escalamientos deben realizarse a través de la herramienta Azure DevOps (http://devops.supersociedades.local/DID/Nuevas%20necesidades%20del%20negocio), específicamente en el proyecto con nombre "Nuevas necesidades del negocio" o mediante correo electrónico.



Para esto, se debe diligenciar el formulario electrónico 46002 (http://devops.supersociedades.local/DID/Nuevas%20necesidades%20del%20negocio/_backlogs/backlog/Nuevas%20necesidades%20del%20negocio%20Team/Requirements) contoda la información necesaria para que el escalamiento sea revisado y analizado A continuación la información requerida:



Título del requerimiento: Se requiere que el usuario cree un título corto pero diciente, relacionado a la solicitud que está escalando.

Descripción de la necesidad: Se debe explicar detalladamente la necesidad en términos de la problemática o brecha identificada.

Oportunidad de mejora esperada: Se debe explicar detalladamente el objetivo de la necesidad, qué va a mejorar si se resuelve la necesidad y qué se espera de la solución. Si existen temas normativos deben ser especificados.

Justificación del requerimiento: Se debe explicar de manera contundente y argumentativa la justificación del requerimiento mediante beneficios tangibles e intangibles

Información Base: Hace referencia a la información básica de la persona que realiza el escalamiento, tal como nombre, cargo y dependencia de la superintendencia de sociedades.

5.5. LEVANTAMIENTO DE INFORMACIÓN SPRINT INCEPTION

Con el propósito de dar solución a las necesidades de negocio escaladas, por cualquiera de las fuentes (defectos o mejoras), GIDAA se encargará del levantamiento de la información necesario para poder gestionar las solicitudes oportunamente.

Los requerimientos de acuerdo con lo definido y priorizado por los interesados se analizarán y formalizarán a través del formato GTI-FM-021 para documentar la(s) Historia(s) de Usuarios trabajadas en conjunto con el usuario funcional.

Adicionalmente, deberán quedar registradas en Azure DevOps (http://devops.supersociedades.local/DID) dentro del proyecto según corresponda. Para iniciar la planeación del desarrollo, es necesario que el usuario funcional APRUEBE las definiciones consignadas en la(s) historia(s) de usuario(s).

Para la construcción de sistemas de información nuevos, se deberá diligenciar el formato de especificación de requisitos ERS.

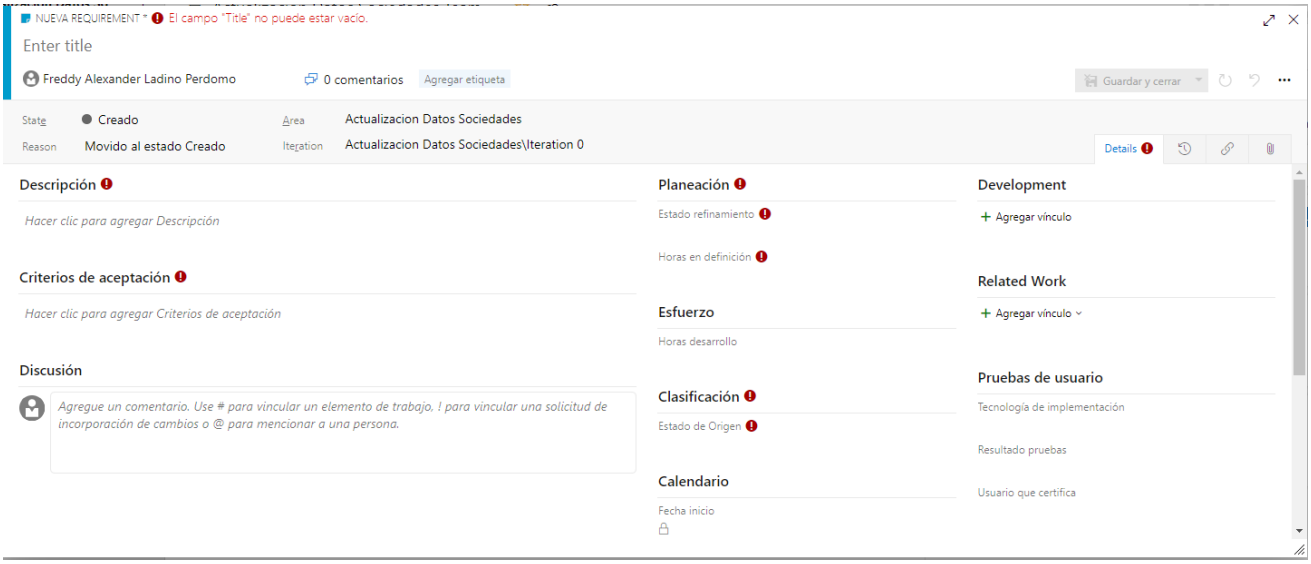
5.5.1. Creación de los Requerimientos – Product Backlog.

El Product Backlog está conformado originalmente por un conjunto de requerimientos levantados y documentados por el equipo de Analistas de requerimientos, resultado de las peticiones realizadas por las áreas funcionales; o por solicitudes realizadas a

la DTIC mediante memorandos, correos electrónicos o comités directivos, de años anteriores o de fecha actual; integrado por descripciones, generales - historias de usuario o épicas – conjunto de historias de usuario.

La Entidad ha identificado para todas las necesidades crear historias de usuario que reúnen la parte funcional + la parte técnica identificando éstas como los criterios de aceptación de los productos.

Las Historias de usuario se documentarán en la herramienta Azure DevOps (http://devops.supersociedades.local/DID), teniendo en cuenta la organización de los aplicativos definida en la herramienta.



5.5.2. Desarrollo – Construcción.

Para comenzar la ejecución de lo planeado como es atender los requerimientos del negocio se crearon con el equipo de fábrica tres sub-equipos de trabajo:



Los Roles de cada equipo Scrum son:

- Product Owner: será equipo funcional, dueño de las necesidades.
- Equipo desarrollo: equipo de profesionales que realizarán las tareas del ciclo de desarrollo

Cada uno de los equipos de la Supersociedades, conformados por los profesionales de fábrica debe trabajar con los elementos definidos y que aplican en el desarrollo de las actividades técnicas bajo el marco SCRUM, los cuales son:

Artefactos:

- Product Backlog.
- Sprint Backlog.

Eventos:

- Sprint Planning Meeting.
- Daily Scrum
- Sprint Review

5.5.3. Reunión de Planificación – Sprint Planning Meeting.

Cada equipo debe realizar la reunión de planeación para programar las tareas a realizar que consiste en tomar las historias de usuario que conforman el producto entregable las deben llevar a tareas técnicas, las cuales deberán estimarse en puntos de desarrollo para ser ejecutadas dentro del sprint planeado.

La Reunión de Planificación de Sprint tiene una duración máxima dependiendo el tiempo definido para el sprint:

Tiempo sprint	Duración planeación
1 semana	2 horas
2 semanas	4 horas
3 semanas	6 horas
4 semanas	8 horas

El líder de equipo se deberá asegurar que el evento se lleve a cabo y que los asistentes entiendan su propósito.

La Reunión de Planificación de Sprint responde a las siguientes preguntas:

- ¿Qué puede entregarse en el Incremento resultante del Sprint que comienza?

Objetivo del Sprint: el objetivo del Sprint es una meta establecida para el Sprint que puede ser alcanzada mediante la implementación de la Lista de requerimientos o Backlog. La lista de historias de usuario seleccionadas del Backlog, ofrecen una función coherente, que puede ser el objetivo del Sprint.

- ¿Cómo se conseguirá hacer el trabajo necesario para entregar el Incremento?
  Creación de tareas: la historia de usuario se desagregará en tareas de desarrollo, con el propósito de segmentar el trabajo en pequeños objetivos y hacer un mejor seguimiento al avance del proyecto.

  Estimación: el equipo de desarrollo deberá realizar una estimación del tiempo que va a llevar desarrollar las tareas anteriormente mencionadas, teniendo como base la capacidad del equipo (cantidad de personas asignadas y la dedicación diaria que tendrá). Esto implica que el equipo se comprometa a cumplir un objetivo, basados en tiempos que ellos mismos definieron.

El resultado de esta reunión es la definición del **Sprint Backlog** – conjunto de tareas a realizar.

5.5.4. Reuniones Diarias – Daily Scrum.

Una vez realizada la reunión de planificación de sprint, se inician los desarrollos en el sprint, el cual es un bloque de tiempo de quince días, durante el cual se crea un incremento de producto "Terminado", utilizable y potencialmente desplegable. Se deberá tener en cuenta lo siguiente en cada sprint:

- No se realizan cambios que puedan afectar al Objetivo del Sprint;
- Los objetivos de calidad no disminuyen; y,
- El alcance puede ser clarificado y renegociado entre el Dueño de Producto (Product Owner) y el Equipo de Desarrollo (Development Team), a medida que se va aprendiendo más.

Cada Sprint puede considerarse un proyecto con un horizonte no mayor de un mes. Al igual que los proyectos, los Sprints se usan para lograr algo.

Cada nuevo Sprint deberá comenzar inmediatamente después de la finalización del Sprint previo. Dentro del sprint se deberá realizar:

El **Daily Scrum** es una reunión diaria, cuyo ideal es que se desarrolle en un tiempo de 15 minutos, se haga a la misma hora y en el mismo lugar con el fin de que el Equipo de Desarrollo sincronice sus actividades y cree un plan para las siguientes 24 horas. Se

hace con el fin de inspeccionar el trabajo avanzado desde el último Daily Scrum y haciendo una proyección acerca del trabajo que podría completarse antes del siguiente.

Durante la reunión, cada miembro del Equipo de Desarrollo explica:

¿Qué hice ayer? ¿qué ayudó al Equipo de Desarrollo a lograr el Objetivo del Sprint?

¿Qué haré hoy para ayudar al Equipo de Desarrollo a lograr el Objetivo del Sprint?

¿Veo algún impedimento que evite que el Equipo de Desarrollo o yo logremos el Objetivo del Sprint?

El Equipo de Desarrollo usa el **Daily Scrum** para evaluar el progreso hacia el Objetivo del Sprint y para evaluar qué tendencia sigue este progreso hacia la finalización del trabajo contenido en la Lista de requerimientos del Sprint.

Dentro del desarrollo del sprint se debe contemplar las pruebas unitarias a cargo del desarrollador, el equipo de desarrolladores debe entregar un producto funcionado, de tal forma que el equipo de Scrum QA, pueda realizar las pruebas técnicas y funcionales con los dueños del producto y dejar listo el producto para entrega a operación.

Para los procesos de prueba se debe cumplir con lo indicado en el formato del SGI GTI-FM-004 Plan, Diseño, Ejecución y evaluación de pruebas y/o documentarlas mediante los test case y test plan en la herramienta Devops. El cual estará a cargo del equipo SCRUM QA.

5.5.5. Entrega del trabajo realizado – Sprint Review

Cada equipo debe realizar una reunión de entrega del producto, en el caso del equipo SCRUM de desarrolladores entregará el producto para que el equipo SCRUM QA lo deje listo para entrega al área de operaciones para su despliegue.

Esta reunión se lleva a cabo luego que se terminó el sprint, para inspeccionar el Incremento y adaptar la Lista de requerimientos si fuese necesario. En este evento se realiza la entrega del objetivo alcanzado en el Sprint, el cual puede ser:

- Entrega del equipo de desarrollo de la funcionalidad a pruebas funcionales o
- Entrega de software al usuario funcional (entrega en la reunión de aprobación de de cambios).

El resultado de la Revisión de Sprint es una Lista de Producto revisada, que define los elementos de la Lista de Producto posibles para el siguiente Sprint. Es posible además que la Lista de Producto reciba un ajuste general para enfocarse en nuevas oportunidades de mejora o priorizaciones.

Luego de que el producto esté listo para entrega al área solicitante, en esta reunión se informa que se seguirán los pasos para puesta en operación del producto, el cual consiste en la creación del formato RFC, presentación en la Reunión de aprobación de cambios y despliegue del producto en operación y uso de este o pruebas de marcha blanca en caso de que sea un nuevo producto o funcionalidad que impacte el sistema de información o proceso bpm en general.

Para cambios al ambiente productivo se debe desarrollar lo indicado en GTI-PR-006_CambiosAmbienteProductivo a través del formato GTI-FM-003 Solicitud de Cambio al ambiente productivo.

5.5.6. Ciclo de pruebas cuando el producto listo para producción.

Durante la ejecución del sprint, los equipos realizarán reuniones diarias, de entrega y de lecciones aprendidas para ir formando al equipo en el manejo de la metodología scrum en una cultura de DevOps, llegando a futuro a usar todas las prácticas de DevOps que aún no se tienen implementadas en la Superintendencia de Sociedades.

Con la evolución del área de TI de la SuperSociedades, lo cual se logra con la migración de aplicaciones legacy, , integración de aplicaciones, madurez en el uso de la nube, formación continua de los equipos, consolidación de equipos se pretende llegar a usar prácticas que aún no se contemplaron en esta metodología, como son:

- Integración y entrega continuas (CI/CD) – Pruebas automáticas.
- Infraestructura como código
- Administración de configuración
- Supervisión continua

Por ahora dentro de esta metodología se contempla que, con el propósito de entregar un software de calidad al usuario final, se realizará una sesión de pruebas con los usuarios funcionales para garantizar que la funcionalidad desarrollada cumple con los estándares definidos en las Historias de usuario entregadas al equipo de desarrollo:

5.5.6.1. Pruebas funcionales e Integrales.

Las pruebas funcionales e integrales deberán ser documentadas en el formato **GTI-FM--004 Plan, diseño, ejecución y evaluación de pruebas**, y estarán a cargo del equipo SCRUM QA, consisten en evaluar la consistencia del software entregado por el equipo de desarrollo vs las definiciones entregadas por el Dueño de producto en los requerimientos. Para esto, cada funcionalidad definida deberá ser probada exhaustivamente, de tal forma

que se garantice la calidad del software, en caso de que no cumpla algún requisito, se le deberá informar al equipo de desarrollo para que corrijan el hallazgo y se deberá volver a hacer la prueba.

Las pruebas integrales incluyen al usuario final, con el propósito de integrarlo en el proceso, para que sea transparente y validar que la funcionalidad cumple con sus expectativas. Luego de estas pruebas, si salen exitosas, el usuario funcional deberá certificarlas, para avanzar en el flujo hacia el paso a producción.

Finalmente, el formato deberá ser cargado en la herramienta Azure DevOps y mostrado en la reunión de aprobación de cambios para la aprobación del paso a producción.

5.5.6.2. Pruebas de seguridad

Las pruebas de seguridad permiten evaluar el nivel de seguridad y confianza que posee el sistema de información construido, previniendo así debilidades y vulnerabilidades durante el ciclo del desarrollo del software.

Para los desarrollos adquiridos mediante un tercero como compra de software o para los desarrollados por el equipo de fábrica interna, como mínimo se deberán ejecutar los siguientes tipos de pruebas de seguridad:

- Escaneo de seguridad a nivel de librerías: Esta actividad consiste en verificar que las librerías que se utilizan en una aplicación provienen de fuentes confiables, permitiendo así reducir el riesgo de que la aplicación incorpore código malicioso.

- Análisis de código estático: Este análisis se realiza en las primeras etapas del desarrollo, sobre el código sin necesidad de ejecutarlo. Esto con el fin de mejorar la calidad y mantenibilidad del código fuente, detectar errores de programación, violaciones de buenas prácticas y vulnerabilidades de seguridad.

- Análisis de vulnerabilidades web: Este tipo de análisis permite identificar vulnerabilidades en aplicaciones web, como inyección SQL, XSS y autenticación insegura. Este análisis se basa en la identificación de los 10 riesgos más críticos en seguridad web, según el OWASP top 10:
  - ✓ Controles de acceso rotos
  - ✓ Fallos criptográficos
  - ✓ Ataques de inyección
  - ✓ Diseño inseguro
  - ✓ Configuraciones incorrectas de seguridad

- ✓ Componentes vulnerables y obsoletos
- ✓ Fallos de identificación y autenticación
- ✓ Fallos de integridad de software y datos
- ✓ Fallos de registro y monitoreo de seguridadg
- ✓ Falsificación de solicitudes del lado del servidor

- Análisis de vulnerabilidades de host: Con este análisis se busca detectar puertos abiertos que puedan mostrar información aun atacante para ser exploit en los servidores.

5.5.7. Excepciones a la aplicación de pruebas de seguridad para nuevas versiones o aplicaciones

De manera excepcional, podrán omitirse las pruebas de seguridad descritas en este lineamiento durante el despliegue de nuevas versiones o aplicaciones únicamente cuando se trate de cambios de nivel BAJO, siempre que se cumplan todas las siguientes condiciones:

- El cambio no introduce ni modifica funcionalidades asociadas a autenticación, autorización, cifrado, tratamiento de datos personales o consumo de servicios externos.
- No se han incorporado nuevas librerías, componentes o dependencias externas.
- El código afectado no corresponde a módulos críticos ni a componentes expuestos a usuarios o redes públicas.
- El entorno tecnológico (versiones de frameworks, servidores de aplicación o bases de datos) se mantiene sin alteraciones.
- El área técnica responsable ha documentado el cambio y la documentación ha sido entregada previo a la Reunión de Aprobación de Cambios.

5.5.8. Paso a producción

Una vez terminadas las pruebas integrales y realizados los ajustes relacionados a las pruebas de seguridad y teniendo la certificación del usuario funcional, se da por terminado el ciclo de pruebas y es notificado al procedimiento de GTI-PR-014_EntregaDespliegueSI, ya que los requerimientos del sprint ya cumplen con los estándares de calidad definidos para poder pasar a producción la funcionalidad en

mención. Para esto, el equipo de desarrollo deberá diligenciar el formato GTI-FM-003 Solicitud de cambio al ambiente productivo, el cual tiene la información relacionada al despliegue y cómo va a acoplarse con el software que ya se encuentra productivo. Con base en esto, en la reunión de aprobación de cambios se aprueba o rechaza el despliegue, basándose en la completitud del diligenciamiento del formulario y el impacto de desplegar el aplicativo hacia producción.

5.5.9. Permisos de acceso en Azure DevOps

Las distintas etapas del ciclo de desarrollo — incluyendo el levantamiento de requerimientos, el desarrollo e implementación del código fuente, así como las pruebas—deberán ser gestionadas y documentadas a través de la plataforma Azure DevOps. La asignación de permisos, por proyecto, se realizará conforme al rol que desempeñe cada integrante del equipo, según lo especificado en la siguiente tabla:

Perfil	Grupo	Visualización / Accesos
Analista / Tester	Project Administrators	1. Información general (excepto Wiki) 2. Boards 4. Pipelines 5. Test Plans
Desarrollador	Build Administrators	1. Información general 2. Boards 3. Repos 4. Pipelines 5. Test Plans 6. Artefactos
Líder de Analistas	Administradores Project Collection Build Administrators	Acceso total a todas las funcionalidades del proyecto
Arquitecto	Administradores Project Collection Build Administrators	Acceso total a todas las funcionalidades del proyecto
Líder de Desarrollo	Administradores Project Collection Build Administrators	Acceso total a todas las funcionalidades del proyecto
Coordinador GIDAA	Administradores Project Collection Build Administrators	Acceso total a todas las funcionalidades del proyecto
DevOps_Admin	Administradores Project Collection Build Administrators	Acceso total a todas las funcionalidades del proyecto

5.6. REGISTRO VALORACIÓN, VIDA ÚTIL Y AMORTIZACIÓN DEL SOFTWARE

Este registro deberá hacerse a todos los desarrollos de software construidos por equipos internos o mediante esquemas mixtos bajo responsabilidad de la Superintendencia de Sociedades, que hayan culminado satisfactoriamente las etapas definidas en la metodología de desarrollo y hayan sido puestos en producción.
La valoración deberá realizarse una vez el software haya sido liberado en ambiente productivo y se encuentre operando de manera estable.
El registro del valor del software deberá realizarse de acuerdo con lo establecido a continuación:

5.6.1. Valoración del Software

La valoración del software se realizará una vez el desarrollo haya sido puesto en producción y finalizado su estabilización.

Recursos que deben ser considerados

El valor del software corresponderá a la suma de los recursos invertidos durante su desarrollo, si aplica, entre ellos:

a. **Recursos Humanos**
- Analistas de requerimientos
- Líderes funcionales
- Arquitectos de software.
- Diseñadores UX/UI.
- Desarrolladores.
- Ingenieros DevOps.
- Tester
- Líderes técnicos.
- Gerentes o coordinadores del proyecto.

b. **Recursos Tecnológicos**
- Licencias utilizadas exclusivamente para el proyecto.
- Servicios en la nube consumidos durante el desarrollo.
- Herramientas especializadas adquiridas para la construcción.

c. Servicios Externos
- Consultorías.
- Desarrollos tercerizados.
- Pruebas especializadas.

Fórmula de Valoración

El valor del software será calculado mediante la siguiente fórmula:

Valor del software = Costo recurso humano + Costo Recurso Tecnológico + Costo Servicios Externos

Para poder registrar el valor del software, se deberá contar con la información registrada en el formato GTI-FM-XXX Valoración del Software

5.6.2. Asignación de la vida útil

La vida útil corresponde al período durante el cual se espera que el software genere valor para la Entidad mediante el soporte efectivo de los procesos de negocio para los cuales fue desarrollado.

La vida útil no necesariamente coincide con la existencia física o técnica del software, sino con el tiempo durante el cual se espera que mantenga su utilidad funcional, operacional y estratégica dentro de la organización.

Su definición deberá realizarse una vez el software haya sido puesto en producción y forme parte del inventario de activos intangibles de la entidad.

La siguiente tabla establece los rangos de referencia que se deben utilizar para asignar la vida útil inicial de los desarrollos de software.

Condición	Vida Útil Sugerida
Aplicaciones tácticas o temporales	2 años
Aplicaciones operacionales	5 años
Aplicaciones CORE de negocio	8 años

Aplicaciones tácticas o temporales (2 años)

Soluciones desarrolladas para atender necesidades específicas de corto plazo o iniciativas con un horizonte temporal limitado.

Generalmente presentan una o varias de las siguientes características:

- Soportan proyectos transitorios.
- Atienden requerimientos regulatorios temporales.
- Son utilizadas por un grupo reducido de usuarios.
- Tienen baja complejidad funcional.

- Se espera su reemplazo o retiro en un período corto.

Aplicaciones operacionales (5 años)

Son soluciones orientadas a soportar procesos administrativos u operativos de la organización, con expectativas de uso de mediano plazo.
Sus características principales son:

- Soportan actividades recurrentes de la operación.
- Presentan una demanda estable de usuarios.
- Poseen requerimientos de mantenimiento moderados.
- Pueden ser sustituidas por nuevas versiones o plataformas en el mediano plazo.

Aplicaciones CORE de negocio (8 años)

Son aplicaciones que soportan los procesos misionales o críticos de la organización y cuya indisponibilidad tendría un impacto significativo sobre la operación institucional.

Normalmente presentan las siguientes características:

- Son esenciales para la prestación de servicios.
- Constituyen la fuente principal de información de negocio.
- Soportan procesos críticos o misionales.
- Requieren altos niveles de disponibilidad y continuidad.
- Son objeto de mejoras continuas sin necesidad de ser reemplazadas completamente.

Como mecanismo de apoyo para determinar la vida útil del software, se podrá hacer uso de la siguiente matriz:

Criterio	Bajo	Medio	Alto
Criticidad del negocio	1	2	3
Horizonte de uso esperado	1	2	3
Dependencia tecnológica	3	2	1
Obsolescencia tecnológica esperada	3	2	1
Necesidad de evolución funcional	1	2	3

El resultado será interpretado según la siguiente tabla:

Puntaje Total	Vida Útil Recomendada
5 a 7 puntos	3 años
8 a 11 puntos	5 años
12 a 15 puntos	8 años

Para registrar la vida útil del software se deberá diligenciar el apartado *Registro Vida Util"* del formato GTI-FM-XXX Valoración de software.

5.6.3. Amortización del Software

La amortización es el mecanismo mediante el cual se reconoce gradualmente el valor del software a lo largo de su vida útil estimada.
Para la Superintendencia de Sociedades, la amortización iniciará a partir de la fecha en que el software sea puesto en producción y se encuentre disponible para su utilización por parte de los usuarios finales.

La amortización no representa una salida de efectivo, sino la distribución contable y administrativa del valor del activo durante el período en que genera beneficios para la entidad.

La Amortización será calculada mediante el uso del método lineal de amortización. Este mecanismo se considera el más adecuado para distribuir uniformemente el valor del software durante su vida útil.

Bajo este método, el mismo valor será amortizado en cada período mientras no existan modificaciones en la vida útil o en el valor del activo.

Fórmula de cálculo

Amortización Anual = Valor del software / Vida Útil

Donde:

- **Valor del Software:** Valor determinado mediante el procedimiento de valoración establecido en esta metodología.
- **Vida Útil:** Número de años asignados al activo de software.

5.6.4. Gestión de mejoras y nuevas funcionalidades

Una vez un software ha sido valorado y puesto en producción, cualquier intervención posterior deberá ser evaluada para determinar si corresponde a:

- Mantenimiento correctivo.
- Mantenimiento evolutivo.
- Proyecto de mejora o ampliación funcional.

La clasificación determinará si los costos serán reconocidos como gasto operativo o si deberán incorporarse al valor del activo de software.

Actividades que no modifican el valor del software

Los costos asociados a las siguientes actividades no incrementarán el valor del software y serán tratados como gastos de mantenimiento:

- Corrección de errores.
- Solución de incidentes.
- Ajustes de configuración.
- Actualizaciones de seguridad.
- Ajustes regulatorios menores.
- Optimización de rendimiento.
- Refactorización de código sin impacto funcional.
- Actualización de versiones tecnológicas sin incorporación de nuevas capacidades.

Estas actividades no generan un aumento en la capacidad funcional del activo y, por lo tanto, no modifican su valor ni su vida útil.

Actividades que incrementan el valor del software

Los costos podrán capitalizarse e incrementar el valor del software cuando las mejoras:

- Incorporen nuevas funcionalidades relevantes.
- Permitan soportar nuevos procesos de negocio.
- Generen beneficios adicionales medibles.
- Amplíen significativamente la cobertura de usuarios.
- Incrementen la capacidad transaccional o de procesamiento.
- Extiendan la vigencia tecnológica del activo.
- Modifiquen sustancialmente la arquitectura funcional de la solución.

En estos casos, los costos de desarrollo de la mejora deberán ser valorados utilizando el mismo procedimiento definido para el desarrollo inicial.

Actualización del Valor del Software

Cuando una mejora sea considerada capitalizable, el valor del software será actualizado mediante la siguiente fórmula:

Nuevo Valor del software = Valor neto Actual + Valor de la mejora

5.6.5. Revisión de la Vida Útil

Toda mejora considerada capitalizable deberá incluir una evaluación de la vida útil del software.

La revisión debe determinar si la mejora:

- Mantiene la vida útil original

  Se aplica cuando las nuevas funcionalidades no modifican significativamente la expectativa de uso del sistema.

- Amplía la vida útil

Se aplica cuando la mejora:

- Moderniza tecnológicamente la solución.
- Sustituye componentes obsoletos.
- Extiende la utilización esperada del activo.
- Incrementa significativamente los beneficios futuros.

Recalculo de la Amortización

Cuando una mejora incremente el valor del software y/o modifique la vida útil, la amortización deberá recalcularse sobre el valor neto actualizado.

La nueva amortización se calculará mediante la fórmula:

Nueva Amortización Anual = Nuevo valor del software / Nueva Vida Útil

Para poder registrar la modificación del valor del software, se deberá adicionar la información en el Formato GTI-FM-XXX Valoración de software, construido en la creación del registro nuevo del software.

6. ANEXOS Y REGISTROS

- GIN-PO-001 Documento de Políticas del SGI
- GIN-PO-003 Política de Seguridad y Privacidad de la Información del SGSI
- GTI-PR-003_ImplementacionSistemasInformacion
- GTI-PR-006_CambiosAmbienteProductivo
- GTI-PR-014_EntregaDespliegueSI
- GTI-GU-020 Lineamientos de Arquitectura y Desarrollo de Software
- GTI-FM-003 Solicitud de Cambio al ambiente productivo
- GTI-FM-004 Plan, Diseño, Ejecución y evaluación de pruebas.
- GTI-FM-021 Historia de Usuarios.
- GTI-FM-027 Arquitectura de Software
- GTI-FM-028 Especificacion de Requisitos
- GTI-FM-XXX Valoración de Software

7. CONTROL DE CAMBIOS

Versión	Fecha	Descripción del Cambio
001	29-05-2020	Creación del documento
002	15-08-2020	Se adicionan directrices de desarrollo y se elimina lo correspondiente a uso del formato GINT-F-006.
003	28/04/2025	Actualización de entregables, actualización de parámetros de la metodología, Articulación con la nueva versión del MRAE- v3. Inclusión de nuevos formatos y guías. De acuerdo con las nuevas disposiciones, se modifica la codificación del tipo documental. Nombre anterior: GINT-MT-001
004	14/09/2026	Se realiza ajuste al documento por códigos desactualizados mencionados dentro del documento como son GTI-GU-020, GIN-PO-001, GIN-PO-003 y se incluye el apartado No. 5.6 Registro, valoración, vida útil y amortización del software, y el formato GTI-FM-00X Valoración del Software

Elaboró	Revisó	Aprobó
Nombre: Marisol Cargo: Profesional Grupo de Sistemas y Arquitectura – Profesional Grupo de Innovación, Desarrollo y Arquitectura de Aplicaciones Fecha: 31 de agosto de 2026	Nombre: Marisol Cargo: Coordinador Grupo de Sistemas y Arquitectura – Coordinador Grupo de Innovación, Desarrollo y Arquitectura de Aplicaciones Fecha: 31 de agosto de 2026	Nombre: Ricardo Fernelix Ríos Cargo: director de Tecnologías de la Información y las Comunicaciones Fecha: 14 de septiembre de 2026