




**ALCALDÍA MAYOR  
DE BOGOTÁ D.C.**  
CULTURA RECREACIÓN Y DEPORTE  
Instituto Distrital de las Artes


# **POLÍTICA DE DESARROLLO DE SOFTWARE**



	<b>GESTIÓN DE TECNOLOGÍAS DE LA INFORMACIÓN Y LAS COMUNICACIONES - TIC</b>	Código: GTIC-POL-03
		Fecha: 10/05/2021
	<b>POLÍTICA DE DESARROLLO DE SOFTWARE</b>	Versión: 01
		Página 2 de 29


HISTORICO DE CAMBIOS		
Versión	Fecha de Emisión	Cambios realizados
01	10/05/2021	Emisión Inicial

Elaboró:	Revisó:	Aprobó:	Avaló:
<b>Hernán Mauricio Rincón Bedoya</b> Contratista Tecnologías de la Información			
<b>Cristian Camilo Calderón Tapia</b> Contratista Tecnologías de la Información	<b>Camila Crespo Murillo</b> Contratista Oficina Asesora de Planeación y Tecnologías de la Información	<b>Carlos Alfonso Gaitán Sánchez</b> Jefe de la Oficina Asesora de Planeación y Tecnologías de la Información	<b>Carlos Alfonso Gaitán Sánchez</b> Jefe de la Oficina Asesora de Planeación y Tecnologías de la Información
<b>Steven Hernández Ríos</b> Contratista Tecnologías de la Información			
<b>Laura Viviana Cruz Martínez</b> Contratista Tecnologías de la Información			

	<b>GESTIÓN DE TECNOLOGÍAS DE LA INFORMACIÓN Y LAS COMUNICACIONES - TIC</b>	<b>Código: GTIC-POL-03</b>
		<b>Fecha: 10/05/2021</b>
	<b>POLÍTICA DE DESARROLLO DE SOFTWARE</b>	<b>Versión: 01</b>
		<b>Página 3 de 29</b>

## CONTENIDO

1.	OBJETIVO	4
2.	ALCANCE	4
3.	GLOSARIO	4
4.	CONSIDERACIONES	11
4.1	Arquitecturas de referencia para el desarrollo de software	11
4.2	Atributos de calidad	11
4.3	Principios y Patrones de diseño	12
4.4	Metodologías de desarrollo	13
4.5	Gestión y documentación de los sistemas de información	14
4.6	Estándares, herramientas y tecnologías para el desarrollo de software	14
5.	LINEAMIENTOS	15
5.3	Declaración de la política	15
5.2	Responsabilidades	16
5.3	Política de desarrollo de software	16
5.3.1	Arquitecturas de referencia para el desarrollo de software	16
5.3.2	Atributos de calidad	17
5.3.3	Principios y patrones de diseño	22
5.3.4	Metodologías de desarrollo	23
5.3.5	Gestión y documentación de los sistemas de información	23
5.3.6	Estándares, herramientas y tecnologías para el desarrollo de software	24
5.3.7	Lineamientos para terceros externos a la entidad	26
5.3.8	Uso de Software del IDARTES por parte de terceros	28
5.4	Verificación de la Política	28
5.5	Aplicabilidad	29
6.	REFERENCIAS	29

	<b>GESTIÓN DE TECNOLOGÍAS DE LA INFORMACIÓN Y LAS COMUNICACIONES - TIC</b>	Código: GTIC-POL-03
		Fecha: 10/05/2021
	<b>POLÍTICA DE DESARROLLO DE SOFTWARE</b>	Versión: 01
		Página 4 de 29

## 1. OBJETIVO

Establecer las políticas aplicables a todo el ciclo de vida de desarrollo de software para el Instituto Distrital de las Artes - IDARTES, con el fin de regular y establecer el marco normativo interno frente a la gestión y desarrollo de software alineado al cumplimiento de la política de Gobierno Digital, expresada en el Decreto 1008 del 14 de junio de 2018.


Al contar con estas políticas debidamente documentadas se establecerán los lineamientos que guiarán el comportamiento personal y profesional sobre la información y la construcción de software que el Instituto Distrital de las Artes en su misión de garantizar el ejercicio de los derechos culturales de los habitantes del Distrito Capital gestiona día a día.

## 2. ALCANCE


Las políticas aquí definidas aplican a todos los servidores públicos, contratistas y practicantes que tienen un vínculo laboral y/o contractual con el Instituto o personal de otras entidades que intervengan en los procesos de desarrollo de software de la entidad, y por tanto tendrán acceso a los sistemas de información para el cumplimiento de sus labores, obligaciones y funciones, por lo cual, tienen la responsabilidad de velar por los pilares de la seguridad de la información, especialmente si la información se encuentra tipificada como información pública clasificada o pública reservada.

## 3. GLOSARIO


- **Ambiente de Desarrollo:** Es la infraestructura tecnológica (hardware y software), controlado por el o los desarrolladores de la funcionalidad con el fin de poder documentar el código de programación sin afectar otros ambientes.
- **Ambiente de Producción:** Es la infraestructura tecnológica (hardware y software), en donde se aloja la última versión estable y en uso de la funcionalidad desarrollada.

	<b>GESTIÓN DE TECNOLOGÍAS DE LA INFORMACIÓN Y LAS COMUNICACIONES - TIC</b>	Código: GTIC-POL-03
		Fecha: 10/05/2021
	<b>POLÍTICA DE DESARROLLO DE SOFTWARE</b>	Versión: 01
		Página 5 de 29


- **Ambiente de Pruebas:** Es la infraestructura tecnológica (hardware y software), controlado tanto por el o los desarrolladores de software, como por el o los usuarios de pruebas del área de negocio que solicita el desarrollo del software, con el fin de poder realizar las pruebas necesarias a la funcionalidad en desarrollo con el fin de garantizar su correcto funcionamiento.
- **Atributos del sistema:** Son las características o parámetros específicos de la funcionalidad a desarrollar, es decir todo aquello que va a realizar el software.
- **Casos de Pruebas:** Es la documentación que se realiza con el fin de determinar si el sistema de información construido cumple con los requerimientos funcionales y no funcionales identificados en la etapa de análisis.
- **Ciclo de vida de Desarrollo de Software:** Conjunto de actividades de un proyecto de construcción y resultados asociados que generan un producto de software.
- **Códigos o Archivos fuentes:** Son los documentos que contienen las líneas de código que se ejecutaron en el lenguaje de programación, utilizado para el desarrollo del producto o software.
- **Desarrollo de software:** Actividades involucradas desde la concepción de la idea, análisis, diseño, construcción e implementación de aplicaciones de software.
- **Ejecutables o Instaladores:** Son aquellos archivos consolidados del software desarrollado que sirven para poder ejecutar o instalar la aplicación en un equipo de cómputo.
- **Mantenimiento de Software:** Actividades involucradas para generar pequeños cambios realizados a una aplicación de software que está en ambiente de producción y en uso.
- **Modelo de datos:** Representación gráfica de la estructura de los datos que serán capturados y almacenados en una base de datos.
- **POO:** Programación orientada a objetos

	<b>GESTIÓN DE TECNOLOGÍAS DE LA INFORMACIÓN Y LAS COMUNICACIONES - TIC</b>	Código: GTIC-POL-03
		Fecha: 10/05/2021
	<b>POLÍTICA DE DESARROLLO DE SOFTWARE</b>	Versión: 01
		Página 6 de 29

- **Clase:** Es una abstracción que representa un elemento que puede ser tangible o intangible pero que posee características o atributos y comportamientos o métodos. Los atributos pueden tener modificadores de acceso generalmente público, privado y protegido, estos determinan el encapsulamiento de la clase.
- **Modificadores de acceso:** son instrucciones o conjunto de palabras clave propias de cada lenguaje de programación orientado a objetos que permite controlar la visibilidad de clases, estado (propiedades) y funcionalidades (métodos) de una aplicación desde otras partes de la misma. Los modificadores más comunes son public, private y protected.
- **Objeto:** Es una instancia de una clase UML dentro de una aplicación de software, que consta de un estado (el conjunto de atributos), y comportamientos, se pueden entender como pequeñas funcionalidades que hacen parte de la solución total desarrollada.
- **APIE:** Acrónimo que hace referencia a los principios de diseño orientado a objetos en el orden: Abstracción, Polimorfismo, Inheritance (Herencia) y Encapsulamiento.
- **SOLID:** Principios de diseño cuya finalidad es hacer un código más mantenible y limpio.
- **Abstracción:** Capacidad de representar la información que es importante para el contexto del problema.
- **Herencia:** capacidad de construir nuevas clases a partir de clases existentes, la clase hija hereda el estado (atributos) y el comportamiento (métodos) definidos. Generalmente se usa la palabra clave extends para denotar la clase padre. En algunos lenguajes de programación no existe la herencia múltiple, es decir que una clase sólo puede heredar de una clase padre.
- **Polimorfismo:** Capacidad de un método de devolver valores diferentes dadas ciertas condiciones (parámetros y herencia).


	<b>GESTIÓN DE TECNOLOGÍAS DE LA INFORMACIÓN Y LAS COMUNICACIONES - TIC</b>	Código: GTIC-POL-03
		Fecha: 10/05/2021
	<b>POLÍTICA DE DESARROLLO DE SOFTWARE</b>	Versión: 01
		Página 7 de 29

- **Polimorfismo por sobrecarga:** En el evento en que una clase tiene dos métodos con el mismo nombre y cada método tiene diferente número o tipo de parámetros. El entorno de ejecución que está instanciando la clase puede distinguir cuál método usar.
- **Polimorfismo por sobrescritura:** Cuando se hereda, desde la clase hija es posible redefinir un método propio de la clase padre. El entorno de ejecución que está instanciando la clase puede distinguir cuál método usar.
- **Encapsulamiento:** Habilidad de un objeto de decidir qué partes puede exponer a otros objetos, se usa generalmente los modificadores de acceso para tal fin.
- **Clase Abstracta:** Al igual que las clases, representan una abstracción. No obstante, se enfocan en generalidades de manera que son usadas para iniciar jerarquía de clases, pueden tener estados (atributos) y comportamientos (métodos) y puede implementar dichos métodos (tener contenido en los métodos). A diferencia de una clase convencional con la clase abstracta no es posible instanciar objetos directamente, por lo que esta acción se debe hacer a través de una clase hija o clase concreta que herede de la clase abstracta.
- **Interfaz:** Al igual que las clases abstractas, representan una abstracción de un elemento que representa el inicio de una jerarquía de clases. A diferencia de la clase abstracta la interfaz no tiene estado (atributos) y el comportamiento (métodos) solo se definen “salvo algunos lenguajes como Java 8 o C# 8 en donde sí se pueden implementar los métodos”, creando un contrato con las clases hijas que lo implementan, pues estas deben de forma obligatoria implementar los métodos. A diferencia de las clases abstractas, una clase puede implementar más de una interfaz, pero dependiendo del lenguaje de programación sólo puede heredar de una sola clase padre.
- **Prototipo:** Es una versión inicial de la funcionalidad desarrollada mediante la cual se busca establecer el camino de lo que se piensa desarrollar tanto a nivel de funcionalidad como a nivel de diseño de la aplicación.


	<b>GESTIÓN DE TECNOLOGÍAS DE LA INFORMACIÓN Y LAS COMUNICACIONES - TIC</b>	<b>Código: GTIC-POL-03</b>
		<b>Fecha: 10/05/2021</b>
	<b>POLÍTICA DE DESARROLLO DE SOFTWARE</b>	<b>Versión: 01</b>
		<b>Página 8 de 29</b>

- **Pruebas de Aceptación:** Son aquellas que se realizan con el usuario final donde se tienen todas las funcionalidades dispuestas, para la prueba total de la solución desarrollada, se realizan con el fin de identificar si el producto cumple con las necesidades pactadas o si aún es necesario realizar cambios o ajustes sobre el mismo.
- **Pruebas de Integración:** Posterior a las pruebas unitarias se consolidan los componentes que se requieren y se realizan pruebas de varios de ellos con el fin de identificar su correcto comportamiento.
- **Pruebas Unitarias:** Son las pruebas que se realizan a cada una de las funcionalidades por separado con el fin de identificar su comportamiento de manera unitaria.
- **Requerimiento:** Es la concepción de la idea la cual busca generar soluciones a necesidades específicas, el cual se puede manejar mediante una herramienta automatizada de software.
- **Software:** Son las aplicaciones desarrolladas bajo un lenguaje de programación y un entorno de ejecución que dan solución a un problema o necesidad específica.
- **Software de sistema:** son los encargados de hacer el enlace con el hardware para ofrecer una interfaz al usuario.
- **Software de programación:** conocidos como IDE entornos integrados de desarrollo, se caracterizan por ser editores de código, depuradores, compiladores y son herramientas para la construcción de software de aplicación
- **Software de aplicación:** son herramientas que se derivan de la programación de software y permiten al usuario llevar a cabo tareas específicas y optimizar procesos cotidianos.
- **Pilares de seguridad de información:** son características de la información en un entorno seguro (integridad, disponibilidad, confidencialidad, autenticidad y no repudio).




	<b>GESTIÓN DE TECNOLOGÍAS DE LA INFORMACIÓN Y LAS COMUNICACIONES - TIC</b>	<b>Código: GTIC-POL-03</b>
		<b>Fecha: 10/05/2021</b>
	<b>POLÍTICA DE DESARROLLO DE SOFTWARE</b>	<b>Versión: 01</b>
		<b>Página 9 de 29</b>

- **Repositorio:** espacio exclusivo para guardar información digital y la mantiene de forma ordenada, por ejemplo, el conjunto de datos de un sistema de información.
- **Arquitectura de negocio / Arquitectura misional:** Describe los elementos de una institución, que le permiten implementar su misión. Esta arquitectura incluye el catálogo de servicios misionales; el modelo estratégico; el catálogo de procesos misionales, estratégicos y de soporte; la estructura organizacional, y el mapa de capacidades institucional. Se utiliza como punto de partida para el diseño de la arquitectura de TI.
- **Arquitectura de Referencia:** Es un diseño de alto nivel, sin detalles tecnológicos o de productos, que se utiliza como una plantilla para guiar el bosquejo de otras arquitecturas más específicas. Esta plantilla incluye los principios de diseño que la guían, las decisiones de alto nivel que se deben respetar, los componentes que hacen parte de la solución, sus relaciones tanto estáticas como dinámicas, las recomendaciones tecnológicas y de desarrollo, las herramientas específicas de apoyo a la construcción y los componentes existentes reutilizables. El concepto de Arquitectura de Referencia se puede utilizar como base del diseño detallado de arquitecturas de solución, de software, de información o de plataforma tecnológica.
- **Arquitectura de Información / Datos:** Define la estructura con la cual está representada y almacenada la información de una organización, lo mismo que los servicios y los flujos de información existentes y que soporta. Incluye el modelo conceptual, el modelo de indicadores, los componentes de información y sus relaciones, y la representación lógica y física de los datos, entre otros. Esta arquitectura expresa también la relación que tiene con la arquitectura misional y con las demás arquitecturas de TI.
- **Arquitectura de Sistemas de Información / de aplicaciones:** Describe cada uno de los sistemas de información y sus relaciones entre ellos. Esta descripción se hace por medio de una ficha técnica que incluye las tecnologías y productos sobre los cuales está construido el sistema, su arquitectura de software, su modelo de datos, la información de desarrollo y de soporte, y los requerimientos de servicios tecnológicos, entre otros. Las relaciones entre los sistemas de información se detallan en una Arquitectura de Integración, que muestra la manera en que los sistemas comparten información y se sincronizan entre ellos. Esta arquitectura debe

	<b>GESTIÓN DE TECNOLOGÍAS DE LA INFORMACIÓN Y LAS COMUNICACIONES - TIC</b>	Código: GTIC-POL-03
		Fecha: 10/05/2021
	<b>POLÍTICA DE DESARROLLO DE SOFTWARE</b>	Versión: 01
		Página 10 de 29

mostrar también la manera como los sistemas de información se relacionan con el software de integración (buses de servicios), de sincronización (motores de procesos), de datos (manejadores de bases de datos) y de interacción (portales), entre otros.

- **Arquitectura de software:** Describe el conjunto de componentes de software que hacen parte de un sistema de información y las relaciones que existen entre ellos. Cada componente de software está descrito en términos de sus características funcionales y no funcionales. Las relaciones se expresan a través de conectores que reflejan el flujo de datos, de control y de sincronización. La arquitectura de software debe describir la manera en que el sistema de información maneja aspectos como seguridad, comunicación entre componentes, formato de los datos, acceso a fuentes de datos, entre otros.
- **Arquitectura de Servicios Tecnológicos / Arquitectura de Tecnología / Arquitectura de Infraestructura TI:** También es conocida como Arquitectura de Infraestructura. Incluye todos los elementos de TI que soportan la operación de la institución, entre los que se encuentran la plataforma hardware, la plataforma de comunicaciones y el software especializado (sistema operacional, software de comunicaciones, software de integración y manejadores de bases de datos, entre otros).
- **Arquitectura de solución:** Cuando aparece un nuevo requerimiento que afecta varios sistemas de información o varias arquitecturas, se elabora una arquitectura de solución, que define la manera en que se deben ajustar las arquitecturas actuales (información, servicios tecnológicos y sistemas de información) para resolverlo. Esta arquitectura de solución debe respetar las arquitecturas de referencia existentes. Garantiza que los problemas se resuelven con una visión amplia y de alto nivel, y que se tiene en cuenta el impacto de las decisiones que se toman.

	<b>GESTIÓN DE TECNOLOGÍAS DE LA INFORMACIÓN Y LAS COMUNICACIONES - TIC</b>	Código: GTIC-POL-03
		Fecha: 10/05/2021
	<b>POLÍTICA DE DESARROLLO DE SOFTWARE</b>	Versión: 01
		Página 11 de 29

## 4. CONSIDERACIONES

### 4.1 Arquitecturas de referencia para el desarrollo de software


La entidad cuenta con un recurso humano con gran talento que le ha permitido avanzar sustancialmente en el desarrollo de herramientas de software, es por eso que hoy se cuenta con varios sistemas de información que generan valor a las diferentes procesos administrativos y misionales de la entidad.

Conforme se avanza en la obtención de las diferentes metas trazadas dentro de Oficina Asesora de planeación y Tecnologías de la información referente a software, se hace necesario buscar la forma de estandarizar la organización de los proyectos, teniendo como objetivos que sean lo más flexibles, reutilizables, fácil de extender en el tiempo y que en caso de que los proyectos de software se hereden a otros responsables cuenten con la base organizacional para poder administrarlos y modificarlos.

### 4.2 Atributos de calidad

La construcción de una buena arquitectura de software impacta positivamente sobre su capacidad de satisfacer las necesidades de las partes interesadas. Cada característica de esta arquitectura (atributos de calidad) es una propiedad medible del sistema que permite evaluar aspectos de calidad tales como la disponibilidad, capacidad de mantenimiento, eficiencia, funcionalidad, seguridad, usabilidad, escalabilidad, facilidad de pruebas, despliegue y desarrollo. No existe una lista definitiva de atributos, pero los recién mencionados corresponden a los propuestos por el IDARTES para la construcción de todos sus productos de software.

De esta manera la entidad busca lograr de manera ordenada, estructurada, eficiente y segura, que sus aplicaciones y sus diferentes integraciones, minimicen los riesgos

	<b>GESTIÓN DE TECNOLOGÍAS DE LA INFORMACIÓN Y LAS COMUNICACIONES - TIC</b>	Código: GTIC-POL-03
		Fecha: 10/05/2021
	<b>POLÍTICA DE DESARROLLO DE SOFTWARE</b>	Versión: 01
		Página 12 de 29

relacionados a la calidad, costos, tiempo y alcance; aumentando así la capacidad de aceptación del sistema por parte del usuario.

### 4.3 Principios y Patrones de diseño

#### 4.3.1 Principios de Programación Orientada a Objetos APIE

El paradigma de programación orientada a objetos es el más utilizado en la actualidad y aunque hay otros paradigmas emergentes como la programación reactiva, se iniciará mencionando los principios básicos de la POO. A continuación, se mencionan:

- Abstracción
- Polimorfismo
- Herencia
- Encapsulamiento

En la guía de buenas prácticas para el desarrollo de software encuentra más detalle acerca de cómo implementar estos principios.

#### 4.3.2 Principios SOLID


Los 5 principios SOLID de diseño de software son:

- S – Single Responsibility Principle o principio de responsabilidad simple.
- O – Open/Closed Principle o principio de abierto / cerrado.
- L – Liskov Substitution Principle o principio de sustitución de Liskov.
- I – Interface Segregation Principle o principio de segregación de interfaces
- D – Dependency Inversion Principle o principio de inversión de dependencias.

los objetivos que se persiguen al codificar siguiendo los principios SOLID son:

**Objetivo general:** desarrollar un software de calidad.

**Objetivos específicos:**

	<b>GESTIÓN DE TECNOLOGÍAS DE LA INFORMACIÓN Y LAS COMUNICACIONES - TIC</b>	Código: GTIC-POL-03
		Fecha: 10/05/2021
	<b>POLÍTICA DE DESARROLLO DE SOFTWARE</b>	Versión: 01
		Página 13 de 29

1. Crear un **software eficaz**: que cumpla con su cometido y que sea robusto y estable.
2. Escribir un **código limpio y flexible** ante los cambios: que se pueda modificar fácilmente según necesidad, que sea reutilizable y mantenible.
3. Permitir **escalabilidad**: que acepte ser ampliado con nuevas funcionalidades de manera ágil.

En la guía de buenas prácticas para el desarrollo de software encuentra más detalle acerca de cómo implementar estos principios.

#### 4.3.3 Patrones de diseño

Los patrones de diseño describen una solución a un problema recurrente en diseño el cual ocurre en un contexto dado. Estas soluciones han sido extractadas de la solución de problemas reales y especificadas formalmente en documentos disponibles en la industria como un mecanismo de distribución de conocimiento.


Los patrones de diseño se clasifican en:

- Patrones Creacionales
- Patrones Estructurales
- Patrones de Comportamiento

En la guía de buenas prácticas para el desarrollo de software encuentra más detalle acerca de cómo implementar estos patrones.

#### 4.4 Metodologías de desarrollo

Durante el proceso de desarrollo de software se deben tener unas buenas prácticas y un proceso claro para trabajar colaborativamente en equipo. A la hora de poner en marcha un proyecto, toda empresa debe asegurar que el equipo implicado conoce sus tareas y plazos de tiempo de entrega. **Scrum** es una metodología de trabajo que nos

	<b>GESTIÓN DE TECNOLOGÍAS DE LA INFORMACIÓN Y LAS COMUNICACIONES - TIC</b>	Código: GTIC-POL-03
		Fecha: 10/05/2021
	<b>POLÍTICA DE DESARROLLO DE SOFTWARE</b>	Versión: 01
		Página 14 de 29

ayuda a conseguirlo y que, además, permite agilizar la entrega de valor al cliente en iteraciones cortas de tiempo.

Esta etapa requiere generar los siguientes entregables, dando instrucción al procedimiento realizado para la puesta en producción de la aplicación:

**Product Backlog (Artefacto):** El Product Backlog es una lista emergente y ordenada de lo que se necesita para mejorar el producto. Es la única fuente del trabajo realizado por el Scrum Team.


**Sprint Backlog (Artefacto):** El Sprint Backlog se compone del Objetivo del Sprint (por qué), el conjunto de elementos del Product Backlog seleccionados para el Sprint (qué), así como un plan de acción para entregar el Increment (cómo).

#### 4.5 Gestión y documentación de los sistemas de información

El hecho de tener la información de los sistemas de información bien documentados es un indicio que se ha realizado un ejercicio de planificación y diseño de los sistemas de información. Es importante para la entidad pues el conocimiento alrededor de un sistema no se centraliza en personas si no en documentos, garantizando así la transferencia del conocimiento.

#### 4.6 Estándares, herramientas y tecnologías para el desarrollo de software


Los estándares de desarrollo son necesarios para la generación de pautas o lineamientos que se adoptan para conseguir una uniformidad en el desarrollo, facilitando el mantenimiento y la actualización de los aplicativos. También se debe tener claro las herramientas y tecnologías que se pueden o no implementar en la entidad, por tal razón existen unas guías que ayudarán a los desarrolladores a llevar a cabo su trabajo.

	<b>GESTIÓN DE TECNOLOGÍAS DE LA INFORMACIÓN Y LAS COMUNICACIONES - TIC</b>	Código: GTIC-POL-03
		Fecha: 10/05/2021
	<b>POLÍTICA DE DESARROLLO DE SOFTWARE</b>	Versión: 01
		Página 15 de 29

## 5. LINEAMIENTOS

### 5.3 Declaración de la política

- a. La oficina asesora de planeación y tecnologías de la información se compromete con atributos de calidad de software como la modificabilidad, la escalabilidad, la seguridad de la información entre otros al implantar la presente política de desarrollo de software y apoyar los lineamientos aquí definidos.
- b. La verificación de la aplicación de la política de desarrollo de software está a cargo del equipo de Fábrica de Software perteneciente a la Oficina Asesora de Planeación y Tecnologías de la Información.
- c. Las personas involucradas y todos aquellos que tengan responsabilidad sobre el desarrollo de sistemas de información, deben optar los lineamientos contenidos en la presente política con la finalidad de cumplir con los pilares de la seguridad de la información del desarrollo de aplicaciones.
- d. La oficina asesora de planeación y tecnologías de la información es la responsable de hacer conocer a las personas involucradas y vinculadas con el instituto para que cumplan con la política de desarrollo de software, con apoyo con apoyo de la oficina asesora de comunicaciones en su divulgación.
- e. El incumplimiento del funcionario total o parcial de la presente política podrá constituirse como falta disciplinaria y por lo tanto podrá dar lugar a la acción e imposición de la sanción correspondiente establecida en la ley penal, en el código disciplinario único y en las demás normas que se encuentren vigentes.
- f. El incumplimiento de los contratistas, personal que labore en las instalaciones vinculado con proveedores del instituto IDARTES o personal externo (empresa o entidad externa), total o parcial de la presente política estará sujeto a las sanciones contractuales, civiles y/o penales a que haya lugar, por los daños y perjuicios generados al instituto o a terceros.

	<b>GESTIÓN DE TECNOLOGÍAS DE LA INFORMACIÓN Y LAS COMUNICACIONES - TIC</b>	Código: GTIC-POL-03
		Fecha: 10/05/2021
	<b>POLÍTICA DE DESARROLLO DE SOFTWARE</b>	Versión: 01
		Página 16 de 29

## 5.2 Responsabilidades

- a) La política de desarrollo de software es de aplicación obligatoria para todo el personal de desarrollo de software que labore en el instituto.
- b) Las directivas del instituto Distrital de las artes aprueban esta política y son responsables de la autorización de sus reformas.
- c) El oficial de seguridad de la oficina asesora de planeación y tecnologías de la información es la responsable de impulsar la ejecución y cumplimiento del atributo de calidad: Seguridad de los sistemas de información, incluido en esta política de desarrollo de software.
- d) El equipo de fábrica de software es el responsable de impulsar la ejecución y cumplimiento de la política de desarrollo de software.
- e) Los responsables del desarrollo, adquisición y mantenimiento de los sistemas de información tienen la obligación de velar que dicho sistema considerado como un activo de información se mantenga disponible, íntegro y confidencial, mientras esté en el proceso de desarrollo, mantenimiento y puesta en marcha.


## 5.3 Política de desarrollo de software

Teniendo en cuenta la amplitud de perspectivas del proceso de desarrollo de software, se decidió abarcar las políticas por temáticas, las cuales se detallan a continuación:

### 5.3.1 Arquitecturas de referencia para el desarrollo de software

Todo proyecto de desarrollo de software que se desee adelantar en la entidad debe implementar la arquitectura Hexagonal.



	<b>GESTIÓN DE TECNOLOGÍAS DE LA INFORMACIÓN Y LAS COMUNICACIONES - TIC</b>	Código: GTIC-POL-03
		Fecha: 10/05/2021
	<b>POLÍTICA DE DESARROLLO DE SOFTWARE</b>	Versión: 01
		Página 17 de 29

Esta arquitectura permitirá desacoplar la infraestructura de la lógica del negocio, por lo que para su correcta implementación se debe:

- Tener una carpeta SRC donde se guardará toda la lógica del negocio.
- Dentro de la carpeta SRC subcarpetas con cada uno de los objetos de dominio que tendrá la solución ejemplo (usuario, contrato, paz y salvo, etc.).
- Para una correcta implementación de la arquitectura en cada una de las carpetas del dominio se debe crear 3 carpetas:
  - Dominio: contendrá la definición del objeto de dominio, los value object, las excepciones de dominio y la interfaz que va a interactuar con la infraestructura.
  - Aplicación: contendrá los casos de usos del dominio, en otras palabras, contendrá las acciones que podrá realizar ese objeto de dominio.
  - Infraestructura: contendrá la implementación de la interfaz definida en el dominio.
- Se debe recordar como regla de esta infraestructura que la capa superior podrá conocer las capas que se encuentren a un nivel inferior a ellas.


### 5.3.2 Atributos de calidad

#### 1. Disponibilidad

Antes de pasar un sistema a producción, cada proyecto de software deberá entregar un documento donde especifique los parámetros tenidos en cuenta para garantizar la disponibilidad del sistema.

#### 2. Capacidad de mantenimiento

Para llevar a cabo una modificación en cualquier sistema de información del IDARTES, debe realizar las actividades propuestas en el Procedimiento de Mantenimiento y Desarrollo de Software, el cual puede ser consultado en el mapa de procesos de la intranet.

 <p>ALCALDÍA MAYOR DE BOGOTÁ D.C. CULTURA, RECREACIÓN Y DEPORTE Instituto Distrital de las Artes</p>	<b>GESTIÓN DE TECNOLOGÍAS DE LA INFORMACIÓN Y LAS COMUNICACIONES - TIC</b>	Código: GTIC-POL-03
		Fecha: 10/05/2021
	<b>POLÍTICA DE DESARROLLO DE SOFTWARE</b>	Versión: 01
		Página 18 de 29

Cada producto de software elaborado por la entidad debe ser capaz de aceptar la implementación de correcciones y/o mejoras a nivel de código, cambios en su entorno y especificaciones de requerimientos funcionales. Durante esta modificación, la aplicación debe conservar su criterio de estabilidad para así evitar efectos inesperados en su ejecución.

### **2.1 Capacidad de ser analizado**

Todo proyecto de software que vaya a ser implementado en producción debe tener un sistema de log de errores a nivel de software (no de servidor de aplicaciones) que sea visible y fácil de consultar.

### **2.2 Portabilidad**

Toda aplicación desarrollada en la entidad debe tener la capacidad de cambiar de ambiente de producción de una manera ágil, sin afectar su funcionalidad inicial.


## **3. Eficiencia**

Cada aplicación, hará un adecuado uso de los recursos ofrecidos para su entorno de producción de acuerdo a los que usa en condiciones específicas. También se requiere tener en cuenta el recurso utilizado en la interacción con otros productos de software para alguna funcionalidad en específico. De esta manera los tiempos de respuesta y procesamiento de la aplicación serán los apropiados.

## **4. Funcionalidad**

Los productos de software desarrollados deben cumplir y proveer las funciones para satisfacer las necesidades consignadas explícitamente en el Formato de Levantamiento de Requerimientos Funcionales GTIC-F-15 previamente socializado y aprobado por los líderes funcionales.

### **4.1 Interoperabilidad**

	<b>GESTIÓN DE TECNOLOGÍAS DE LA INFORMACIÓN Y LAS COMUNICACIONES - TIC</b>	Código: GTIC-POL-03
		Fecha: 10/05/2021
	<b>POLÍTICA DE DESARROLLO DE SOFTWARE</b>	Versión: 01
		Página 19 de 29

Todo desarrollo de software debe tener la capacidad de intercambiar datos bajo la tecnología REST por medio del formato JSON, respetando los aspectos de seguridad establecidos en este documento.


## 5. Seguridad

Toda aplicación del IDARTES tendrá la capacidad de proteger la información y los datos de tal forma que los usuarios y/o sistemas no autorizados no puedan acceder a ellos. Para conceder al sistema los principios de integridad, autenticación y disponibilidad al sistema, se proponen los siguientes métodos:

1. Antes de realizar el paso a producción, todo proyecto de software debe contar con el visto bueno del oficial de seguridad de la entidad o quien sus veces.
2. Es requisito para el paso a producción que los productos de software cuenten con un documento de análisis de riesgos cómo lo indica la guía de aseguramiento de aplicaciones.
3. El oficial de seguridad deberá gestionar las vulnerabilidades en el desarrollo del software, revisando con una frecuencia mensual los logs generados por las aplicaciones o por el servidor de aplicaciones. Validar herramientas que permitan la automatización de este proceso.
4. Se debe proteger la información sensible como (contraseñas, tokens, firmas digitales) en todos sus estados: tránsito, en reposo y en uso por medio de herramientas y técnicas criptográficas que garanticen la integridad y confidencialidad de la información.

## 6. Usabilidad

Todos los sistemas y sus componentes desarrollados deben ser entendidos y aprendidos por el usuario de manera sencilla y práctica.

	<b>GESTIÓN DE TECNOLOGÍAS DE LA INFORMACIÓN Y LAS COMUNICACIONES - TIC</b>	Código: GTIC-POL-03
		Fecha: 10/05/2021
	<b>POLÍTICA DE DESARROLLO DE SOFTWARE</b>	Versión: 01
		Página 20 de 29

Además, podrá reconocer si el sistema es adecuado y fácilmente entender cómo desarrollar las tareas dentro de él.

Todo producto de software deberá seguir parámetros de usabilidad como simplicidad, organización de los elementos y adaptación a diferentes dispositivos, navegación intuitiva y accesibilidad sin importar su sistema operativo y/o navegador preferido, estos parámetros son descritos en la guía de estilo y usabilidad.

Cada componente desarrollado debe incluir documentación guía para el usuario, como lo son por ejemplo los video tutoriales, diagramas de proceso y manuales.

## 7. Escalabilidad


Cada aplicación debe poder manejar una carga constante y creciente de datos y trabajo, en simultáneo con el crecimiento del IDARTES. A fin de configurar el entorno de la aplicación para la atención de peticiones de manera conjunta y distribuir su carga de trabajo se establece:

1. Se debe implementar un sistema de orquestador de contenedores por medio de Docker y Kubernetes para garantizar el balanceo de cargas y el auto mantenimiento ante fallos por peticiones o no disponibilidad del sistema, contemplando el uso de firewall web o físico, lo anterior para garantizar la recuperabilidad del sistema y escalabilidad horizontal.

## 8. Facilidad de pruebas

Toda aplicación del IDARTES permitirá realizar pruebas a los componentes desarrollados o modificados para así poder identificar si provoca efectos secundarios o adversos durante su ejecución.

A continuación, se expone cada uno de los tipos de pruebas propuestos para verificar y validar la aplicación:

	<b>GESTIÓN DE TECNOLOGÍAS DE LA INFORMACIÓN Y LAS COMUNICACIONES - TIC</b>	Código: GTIC-POL-03
		Fecha: 10/05/2021
	<b>POLÍTICA DE DESARROLLO DE SOFTWARE</b>	Versión: 01
		Página 21 de 29


1. Se deben realizar pruebas unitarias de código con la finalidad de garantizar el funcionamiento esperado de los métodos usados en el desarrollo.
2. Se aplicarán pruebas de integración sobre cada uno de los módulos y otros sistemas con los que se comunica la aplicación. De esta manera se podrá validar el comportamiento de los componentes presentes en el sistema y garantizar su funcionalidad.
3. Se deben realizar pruebas funcionales de sistema que validen su funcionamiento en condiciones específicas: sistemas operativos, navegadores, servidores, bases de datos o tipo de dispositivo en el que lo usen.
4. Posterior a una modificación y/o mejora de alguna funcionalidad en la aplicación, se realizarán pruebas de regresión para asegurar que estos cambios o adiciones no alteraron ni eliminaron funcionalidades existentes.

## 9. Facilidad de despliegue

Cada aplicación debe permitirse instalar y configurar en el entorno de producción sin afectar su integridad ni su total funcionalidad.

Esta etapa requiere generar los siguientes entregables, dando instrucción al procedimiento realizado para la puesta en producción de la aplicación:

1. Documento de instalación, configuración, integración y migración: Este documento incluye los insumos necesarios para los procesos de instalación, configuración, integración y migración del sistema, así como también la coordinación de tiempos y actividades según condiciones específicas. Consulte el FORMULARIO DE DESPLIEGUE PARA PROYECTOS DE SOFTWARE propuesto para esta actividad.

	<b>GESTIÓN DE TECNOLOGÍAS DE LA INFORMACIÓN Y LAS COMUNICACIONES - TIC</b>	Código: GTIC-POL-03
		Fecha: 10/05/2021
	<b>POLÍTICA DE DESARROLLO DE SOFTWARE</b>	Versión: 01
		Página 22 de 29

2. Manual de usuario: Documento en el cual se describe el uso del sistema por parte del usuario final.
3. Manual de administración: Documento técnico y de administración del sistema.
4. Archivos binarios: Instaladores de software base, archivos de base de datos, archivos ejecutables y de configuración.

## 10. Facilidad de desarrollo


Teniendo en cuenta que el desarrollo de software es propiamente una ingeniería, se establecieron una serie de reglas comunes para lograr que el desarrollador atienda la mayor parte de los requerimientos del proyecto en el menor tiempo posible. Así mismo, se busca garantizar un proyecto compuesto por código válido y de calidad haciendo uso de estructuras reconocidas como lo son los patrones de diseño y además que el código sea modelado de manera tal que ayude al desarrollador a explicar a sus colegas a entender cómo solucionar determinados problemas y justificar que esa era la mejor opción.

Para conocer los estándares de desarrollo para los proyectos de la entidad, consulte los siguientes documentos:

- GUÍA DE BUENAS PRÁCTICAS PARA EL DESARROLLO DE SOFTWARE.
- GUÍA ESTÁNDARES, HERRAMIENTAS Y TECNOLOGÍAS PARA PROYECTOS DE DESARROLLO DE SOFTWARE.
- GUÍA ARQUITECTURA HEXAGONAL.

### 5.3.3 Principios y patrones de diseño

1. Todo desarrollo de software debe seguir los principios de programación orientada a objetos APIE de la mano de los principios SOLID de forma obligatoria. ver ***guía de buenas prácticas para el desarrollo de software***.

	<b>GESTIÓN DE TECNOLOGÍAS DE LA INFORMACIÓN Y LAS COMUNICACIONES - TIC</b>	Código: GTIC-POL-03
		Fecha: 10/05/2021
	<b>POLÍTICA DE DESARROLLO DE SOFTWARE</b>	Versión: 01
		Página 23 de 29

2. Todo desarrollo de software debe implementar la arquitectura de software de referencia mencionada en esta política.
3. Cada desarrollador de software debe implementar patrones de diseño en los proyectos en los que intervenga de acuerdo con la **guía de buenas prácticas para el desarrollo de software**.
4. Cada proyecto de desarrollo debe incluir un ORM para el mapeo de base de datos, validando que dicha herramienta no posea vulnerabilidades de inyección. Dado el caso en que no se pueda implementar el ORM en módulos específicos del proyecto, se deben realizar una justificación y utilizar sentencias preparadas para prevenir ataques de SQL injection.


#### 5.3.4 Metodologías de desarrollo

Se debe establecer para cada proyecto, módulo o formulario la metodología de gestión SCRUM en la cual se oriente el ciclo de vida de desarrollo en los proyectos de software del IDARTES.

Para implementar este marco de trabajo existe el **Instructivo Scrum Para Proyectos De Desarrollo De Software** en comunicarte.

#### 5.3.5 Gestión y documentación de los sistemas de información

1. Todas las iniciativas de desarrollo de software y las nuevas aplicaciones que se implementen en el instituto, deben ser analizadas y gestionadas por la oficina asesora de planeación y tecnologías de la información únicamente.
2. La oficina asesora de planeación y tecnología debe disponer de un espacio en disco para que los equipos de desarrollo almacenen la información de documentación requerida.
3. Todo desarrollo que se realice por y para el IDARTES debe generar los siguientes entregables:
  - Plan de trabajo de acuerdo con la metodología de desarrollo ágil.
  - Historias de usuario.

	<b>GESTIÓN DE TECNOLOGÍAS DE LA INFORMACIÓN Y LAS COMUNICACIONES - TIC</b>	Código: GTIC-POL-03
		Fecha: 10/05/2021
	<b>POLÍTICA DE DESARROLLO DE SOFTWARE</b>	Versión: 01
		Página 24 de 29


- Diagramas de clases, secuencia y relacional.
  - Solicitud de cambios o ajustes a software.
  - Documento de casos de pruebas.
  - Manual de usuario.
  - Códigos fuentes debidamente documentados.
  - Paso a producción de software.
  - Diccionario De Modelo De Datos.
  - Diagrama de arquitectura de software.
4. El acceso a los archivos fuente de las aplicaciones está restringido al personal autorizado por la oficina asesora de planeación y tecnologías información.
  5. La oficina asesora de planeación y tecnología como encargada de los sistemas de información debe custodiar los archivos fuentes y las bases de datos de las aplicaciones propiedad del instituto.
  6. Todo contrato relacionado con el ciclo de vida de desarrollo de software o con la adquisición de software, debe ser supervisado y monitoreado por la oficina asesora de planeación y tecnologías de la información de IDARTES con el fin que se cumplan con los requerimientos propios de la entidad y de la política de desarrollo de software.
  7. Todo contrato suscrito entre el IDARTES y una persona sea natural y/o jurídica cuyo objeto esté centrado en el desarrollo de un producto de software a medida bien sea por ajuste a un sistema existente del IDARTES o un requerimiento nuevo, debe incluir la siguiente obligación:

***Ceder los acuerdos de licenciamiento, propiedad de los códigos fuente y derechos de propiedad intelectual relacionados con el contenido de los aplicativos entregados a IDARTES, registrando los cambios en el código fuente en los repositorios oficiales de la entidad.***


### **5.3.6 Estándares, herramientas y tecnologías para el desarrollo de software**

1. Para el manejo de nomenclatura se deben adoptar algunos estándares que aseguran un código legible, fácil de entender y mantener.



 <p>ALCALDÍA MAYOR DE BOGOTÁ D.C. CULTURA, RECREACIÓN Y DEPORTE Instituto Distrital de las Artes</p>	<b>GESTIÓN DE TECNOLOGÍAS DE LA INFORMACIÓN Y LAS COMUNICACIONES - TIC</b>	Código: GTIC-POL-03
		Fecha: 10/05/2021
	<b>POLÍTICA DE DESARROLLO DE SOFTWARE</b>	Versión: 01
		Página 25 de 29

2. Todo desarrollo de software debe especificar ciertas tecnologías con las cuales se desarrollará los aplicativos, módulos o sistemas, y deben ser un estándar para los equipos de trabajos que realicen dichos desarrollos.
3. Para garantizar seguridad de la información se deben establecer para cada proyecto la separación de ambientes de construcción, testing y producción cada uno representado en una rama del repositorio del proyecto, con el objetivo de controlar de mejor manera los cambios al sistema de información.
4. Todo desarrollo de software debe contar con 3 ambientes: desarrollo, pruebas y producción. Cada uno con diferente base de datos.
5. Todo desarrollo de software debe tener por lo menos 2 ramas en el repositorio de código, una para producción y una para desarrollo. Se recomienda tener una rama independiente para el ambiente de pruebas.
6. Se deben implementar las pruebas de integración que sean necesarias a los sistemas con la finalidad de asegurar que en las aplicaciones desarrolladas se ha implementado los requisitos de seguridad definidos antes de comenzar el desarrollo.
7. Se deben realizar pruebas de integración a las aplicaciones desarrolladas en conjunto con el dueño del sistema de información, haciendo uso del formato de casos de prueba, realizando la respectiva documentación a las pruebas realizadas y aprobando los pasos a producción
8. Realizar el merge de la rama de desarrollo a pruebas una vez finalizado el proceso de desarrollo y realizar el merge de la rama de pruebas a producción una vez validado y aprobado el proceso de pruebas integrales.
9. Las herramientas de desarrollo de software utilizadas deben estar licenciadas por el instituto y estar en custodia de la oficina asesora de planeación y tecnología únicamente.
10. Se debe hacer uso de un software de control de versiones el cual salvaguarde la integridad del código fuente desarrollado, la administración de los repositorios debe estar a cargo de un funcionario y/o cuenta oficial de IDARTES designado por la Oficina asesora de planeación y tecnologías de la información.
11. Mantener actualizada las versiones de lenguaje de programación, frameworks y librerías usadas en los proyectos de acuerdo con la tabla de versiones seguras, lo anterior para solventar los problemas de seguridad del lenguaje de las versiones anteriores.

	<b>GESTIÓN DE TECNOLOGÍAS DE LA INFORMACIÓN Y LAS COMUNICACIONES - TIC</b>	Código: GTIC-POL-03
		Fecha: 10/05/2021
	<b>POLÍTICA DE DESARROLLO DE SOFTWARE</b>	Versión: 01
		Página 26 de 29

12. Se debe identificar frameworks y/o librerías y de autores reconocidos con mantenimiento, actualización y desarrollo activo por parte de la comunidad y que sean ampliamente usados y por lo tanto validados. Los frameworks y/o librerías de terceros reconocidos que incluyen elementos de seguridad son primordiales para minimizar errores de implementación en componentes en los que el desarrollador no tenga el conocimiento y dominio necesario. Es de gran importancia realizar un inventario de dichos frameworks y librerías para tener un control de las actualizaciones y prevenir vulnerabilidad.


Para implementar estos estándares existe la **GUÍA DE ESTÁNDARES, HERRAMIENTAS Y TECNOLOGÍAS PARA PROYECTOS DE DESARROLLO DE SOFTWARE** en el mapa de procesos.

### 5.3.7 Lineamientos para terceros externos a la entidad

En el contexto que desde IDARTES estén interesados en contratar o implementar un desarrollo con terceros (persona jurídica) externos a la entidad, es necesario tener en cuenta **los siguientes escenarios**:

- Un producto nuevo de software desarrollado por un tercero.
- Un producto de software ya finalizado por un tercero y adquirido bajo licencia.
- Un producto de software existente desarrollado por un tercero y que se adapte a las necesidades del IDARTES.
- Un software libre que se desee adoptar e implementar en la entidad.
- Un producto de software desarrollado por IDARTES y compartido a otras entidades mediante las diferentes modalidades convencionales y/o contractuales.


Por medio de la tabla 1, se ilustra la aplicabilidad de los lineamientos en cada uno de los escenarios descritos anteriormente, en donde 'Si' hace referencia al lineamiento específico que le aplica y 'No', al no requerido:

	<b>GESTIÓN DE TECNOLOGÍAS DE LA INFORMACIÓN Y LAS COMUNICACIONES - TIC</b>	Código: GTIC-POL-03
		Fecha: 10/05/2021
	<b>POLÍTICA DE DESARROLLO DE SOFTWARE</b>	Versión: 01
		Página 27 de 29

Tipo de desarrollo	5.3.1 Arquitecturas de referencia y 5.3.3 principios y patrones de diseño	5.3.2 Estilos y usabilidad	5.3.4 Metodologías de desarrollo	5.3.5 Gestión y documentación de los sistemas	5.3.6 Estándares, herramientas y tecnologías para el desarrollo
Nuevo desarrollo por externos a la entidad	Si	Si	Si	Si	Si
Desarrollo finalizado por externos a la entidad	No	Si	No	Si	No
Adaptación de Software de Terceros	No	Si	No	Si	No
Adaptación de software libre	No	Si	No	Si	Si
Desarrollado por IDARTES compartido con otras entidades	Si	Si	Si	Si	Si

Tabla 1: Aplicabilidad de lineamientos de políticas de desarrollo

De igual forma, cualquiera que sea el caso es necesario que la Oficina Asesora de Planeación y Tecnologías de la Información realice un **análisis de viabilidad** para la adquisición, implementación o desarrollo de la solución de software.

	<b>GESTIÓN DE TECNOLOGÍAS DE LA INFORMACIÓN Y LAS COMUNICACIONES - TIC</b>	Código: GTIC-POL-03
		Fecha: 10/05/2021
	<b>POLÍTICA DE DESARROLLO DE SOFTWARE</b>	Versión: 01
		Página 28 de 29


### 5.3.8 Uso de Software del IDARTES por parte de terceros

En el evento en que un tercero requiera tener acceso a alguno de los productos de software del IDARTES, bien sea por medio de un API o por medio un perfil de usuario con usuario y contraseña. Se deberá llevar a cabo el siguiente procedimiento:

- La unidad de gestión que hace las veces de enlace con el tercero debe notificar a la oficina asesora de planeación y tecnologías de la información, acerca del tipo de acceso y uso que requiere el tercero.
- Los privilegios de la cuenta creada para el tercero deben mantener el principio del menor privilegio, es decir limitar el acceso a lo que es imprescindible, evitando al máximo las operaciones de tipo CREATE, UPDATE y DELETE sobre los registros de bases de datos.
- Todo API construido en el IDARTES debe seguir el estándar OpenAPI 3+ para establecer la interfaz de servicios a que serán consumidos por terceros por medio de una signatura o contrato.

### 5.4 Verificación de la Política

- a. El instituto distrital de las artes IDARTES velará por el cumplimiento de la presente política de desarrollo de software.
- b. La oficina asesora de planeación y tecnologías de la información realizará el monitoreo del cumplimiento de la política de desarrollo de software. El Área de Control Interno evaluará la política por medio de informes o auditorías de acuerdo con la programación aprobada en el Plan Anual de Auditorías.
- c. La presente política de desarrollo de software será de estricto cumplimiento y se enmarcará en las obligaciones de contratos y/o convenios relacionados con desarrollo y adquisición de software.

	<b>GESTIÓN DE TECNOLOGÍAS DE LA INFORMACIÓN Y LAS COMUNICACIONES - TIC</b>	Código: GTIC-POL-03
		Fecha: 10/05/2021
	<b>POLÍTICA DE DESARROLLO DE SOFTWARE</b>	Versión: 01
		Página 29 de 29

### 5.5 Aplicabilidad

- a) Esta política es aplicable para todos los procesos de construcción y adquisición de software tanto de los desarrollos realizados por contratistas o funcionarios del instituto como los adquiridos a terceros, además se aplica para todos los desarrollos compartidos por medio de convenios interadministrativos
- b) No es aplicable para los paquetes de software de aplicación licenciados.

### 6. REFERENCIAS

- a) Guía del dominio de Sistemas de Información del Mintic
- b) Arquitectura TI para el Distrito – Alta Consejería Distrital TIC
- c) Política de Seguridad de la información
- d) Decreto 1008
- e) Modelo SAMM
- f) Guía para el aseguramiento de aplicaciones
- g) Diccionario de lenguaje común de datos
- h) Procedimiento de mantenimiento de desarrollo de software
- i) Formato de levantamiento de requerimientos funcionales
- j) Casos de prueba funcionales a software
- k) Solicitud de desarrollo actualización o mantenimiento de software y/o formularios.