

Código de buenas prácticas en el desarrollo de software público¹

Guía técnica para el desarrollo sustentable de software en la Administración Pública

¿Qué encontrarás en este documento?

Este documento contiene lineamientos generales y recomendaciones específicas de la [ONTI](#) para el desarrollo de software en la Administración Pública. Vas a encontrar varias referencias al [Decálogo Tecnológico ONTI](#) porque ambos lineamientos se basan en los mismos principios rectores.

El mismo propone ser un marco de referencia para los organismos acerca de las mejores prácticas en el desarrollo de software público. Su confección es el resultado de un esfuerzo colaborativo entre distintos referentes y organismos en varias instancias incluyendo los eventos de la [Comunidad Digital AR](#). Dado que las tecnologías se van mejorando y aparecen nuevas y más potentes herramientas la guía de buenas prácticas será revisada y actualizada al menos 2 veces por año. Invitamos a toda la comunidad de agentes técnicos a sumarse a esta actualización comunicándose con el equipo de software público: softwarepublico@modernizacion.gob.ar

¿Cómo leer esta guía?

La guía técnica para las buenas prácticas de desarrollo de software contiene 7 (siete) puntos principales con algunas subguías en forma de hipervínculo para aquellos temas que requieren mayor profundidad. A su vez los 7 puntos están divididos en tres niveles de acción: El *Ideal* es el rumbo aspiracional de mediano-largo plazo que la ONTI recomienda a las áreas de sistemas en la Administración Pública en donde se encuentran las mejores prácticas y tendencias en la industria TIC a nivel regional y global. El nivel *Mínimo* consiste en un punto de transición hacia el *Ideal* a partir del cual se pretende que las direcciones de sistemas logren alcanzar en el mediano plazo a través de buenas prácticas. *Por dónde empezar* describe los primeros pasos a seguir para comenzar el camino a la innovación y sustentabilidad tecnológica en el desarrollo de software.

¹El software público es aquella solución tecnológica de software desarrollada, usada, modificada y distribuida para facilitar el cumplimiento de objetivos gubernamentales y sociales por el Estado y para el Estado.

1. Entendé las necesidades de los usuarios

Entender los contextos y las necesidades reales de los usuarios es fundamental para que tus sistemas cumplan su función y para que puedas evaluar mejor las prioridades de desarrollo.

Ideal

- Creá [mapas de historias de usuario](#) para entender la experiencia de todos los distintos [tipos de usuarios](#), incluyendo personas con discapacidad
- Desarrollá [prototipos](#) para probar el servicio con usuarios reales
- Tomá decisiones basadas en la recolección y análisis de datos objetivos, incluyendo [tests A/B](#), métricas de uso de los servicios y [pruebas de usabilidad](#)
- Diseñá teniendo en cuenta toda la experiencia del usuario incluyendo las acciones que realiza fuera de los sistemas digitales
- Evaluá tu servicio con las [herramientas de la W3C](#)

Mínimo

- Cumplí las [normativas de accesibilidad](#) para garantizar igualdad real de oportunidades y trato a todas las personas
- Conocé los [estándares de la Dirección Nacional de Servicios Digitales](#)
- Investigá desde qué tipo de dispositivos acceden tus usuarios, para poder priorizar el diseño y desarrollo de una interfaz y experiencia acorde

Por dónde empezar

- Consultá el [Decálogo Tecnológico ONTI #1 - Conocé tu proyecto](#)
- Consultá el [Decálogo Tecnológico ONTI #7 - Asegurá que tus soluciones sean accesibles](#)
- Creá un primer [mapa de historia de usuario](#) para entender la experiencia de tus usuarios más frecuentes

2. Aprovechá la nube y la virtualización

Alojar tus aplicaciones y tomar ventaja de las herramientas y soluciones de infraestructura que ofrece la nube aumenta la flexibilidad, escalabilidad y seguridad, a su vez facilita la reducción de tiempos y costos.

Ideal

- Considera utilizar una solución de Nube para implementar tu servicio antes que cualquier otra opción.
- Las características de tu proyecto podrán indicar como la mejor opción una solución de Nube pública, privada o incluso híbrida. En cualquier caso utilizando entornos virtualizados para tomar ventaja de la economía, flexibilidad y dinamicidad que estos aportan.
- Aprovechá los entornos de desarrollo en la nube pública²
- Optimizá tareas comunes con herramientas de automatización³
- Utilizá contenedores para el despliegue de aplicaciones y el mantenimiento⁴
- Gestioná y automatizá la configuración de despliegues con herramientas de *DevOps*⁵

Mínimo

- Opta por un entorno de Nube para tu desarrollo, aprovechando sus facilidades y herramientas. Si tu aplicación consiste en una combinación de funcionalidades comunes (formularios, ABM, gráficos) entonces considerá las herramientas para desarrollo de aplicaciones en la nube (ofrecidas como SaaS) antes de iniciar un nuevo desarrollo⁶
- Minimizá las diferencias entre los ambientes de desarrollo y producción para favorecer un despliegue continuo de los cambios a tus sistemas

Por dónde empezar

- Consultá el [Decálogo Tecnológico ONTI #3 - Preferí soluciones que utilicen la Nube](#) y buscá cuál es la mejor opción de Nube para desarrollar e implementar el servicio o solución.

² Por ejemplo [Cloud9](#) o [CodeAnywhere](#)

³ Por ejemplo [Zapier](#) o [IFTTT](#)

⁴ Por ejemplo [Docker](#), [Kubernetes](#), [CloundFoundry](#)

⁵ Por ejemplo, [Puppet](#), [Chef](#), [Ansible](#)

⁶ Por ejemplo [Zoho Creator](#) o [Google Apps](#)

3. Reutilizá y compartí software y datos

Desarrollar nuevos productos digitales que reutilicen software y datos permiten aprovechar el trabajo realizado por otros, optimiza recursos, reduce costos y facilita la interoperabilidad entre servicios favoreciendo a la eficiencia y la transparencia

Ideal

- Compartí tu código en el [repositorio oficial de la República Argentina](#) y mantenelo actualizado siguiendo el proceso de [apertura y reutilización del software público \(en desarrollo\)](#)
- [Utilizá una licencia abierta y estándar](#), como [MIT](#), [GPL](#) o [Creative Commons](#)
- Cumplí con las [pautas técnicas de interoperabilidad](#) para el intercambio de información entre sistemas.
- Exponé tus datos mediante una [API](#) estilo [REST](#) que cumpla con las [especificaciones OpenAPI](#)
- Documentá y probá tu [API](#) con herramientas diseñadas para el caso⁷
- Diseñá tus aplicaciones como un [SaaS](#) para que otros organismos las puedan utilizar con facilidad⁸

Mínimo

- Utilizá plataformas comunes ya desarrolladas por el gobierno, como [GDE](#), [Argentina.gob.ar](#), [Mi Argentina](#), [Sistema de Identidad Digital](#) o [Autenticar](#)
- Elegí cómo encarar un [nuevo desarrollo de acuerdo a la necesidad, disponibilidad y sustentabilidad](#)
- Exponé tus datos mediante una [API](#) estilo [REST](#) que cumpla con las [especificaciones OpenAPI](#)
- Utilizá un [framework abierto y activo](#)
- Aplicá y conocé los [estándares de la Plataforma Web Abierta de la W3C](#)
- Favorecé las tecnologías y lenguajes de código abierto por sobre las propietarios para reducir costos y evitar la dependencia de oferentes⁹
- Utilizá tecnologías con una comunidad activa para contar con soporte y recibir actualizaciones periódicas de funcionalidad y seguridad.

⁷ Por ejemplo [Swagger](#)

⁸ Recomendamos la metodología [12factor](#) para cubrir todas las necesidades técnicas involucradas en el proceso

⁹ Por ejemplo el [MEAN stack](#) ([Express](#), [AngularJS](#) y [Node.js](#)), el [LAMP stack](#) (Linux, Apache, MySQL y PHP, Python, Java) o [ASP.NET Core](#).

- Gestioná las bases de datos de tus aplicaciones (relacionales y no relacionales) a modo de *cluster* para que puedan crecer de manera horizontal y favorecé las tecnologías abiertas¹⁰
- Asegurá una estrategia de mantenimiento y soporte en tus contrataciones.
- Desarrollá código modular para que los componentes también se puedan reutilizar
- Agregá tu software al [catálogo de software público](#)
- Abrió tus datos públicos en el [portal de datos públicos](#)

Por dónde empezar

- Consultá el [Decálogo Tecnológico ONTI #4 - Utilizá estándares abiertos y soluciones interoperables](#)
- Consultá el [Decálogo Tecnológico ONTI #5 - Elegí plataformas y soluciones comunes de Gobierno](#)
- Consultá el [Decálogo Tecnológico ONTI #6 - Desarrollá soluciones reutilizables y compartilas](#)
- Analizá cómo abrieron su código algunas agencias de la administración pública, como por ejemplo la Dirección Nacional de Servicios Digitales en cuanto a la implementación de [Poncho](#)
- Unite a la [Comunidad Digital](#) para compartir y consultar experiencias sobre soluciones y herramientas
- Contactá al [equipo de software público](#) para asesorarte sobre cómo abordar tus nuevos desarrollos

¹⁰ Relacionales: [PostgreSQL](#) 10x, [MySQL](#) 5.7x. No relacionales: [MongoDB](#) 3.6x, [Elasticsearch](#) 5.5x, [Redis](#) 5.0x

4. Probá el servicio de punta a punta

Realizar pruebas de tu sistema en todo momento del desarrollo te permite adaptarte fácilmente a los cambios y asegura que las herramientas cumplan las necesidades para las cuáles fueron diseñadas.

Ideal

- Integrá prácticas *DevOps* en el ciclo de producto para realizar despliegues continuos que te permitan realizar pruebas automatizadas en todos los ambientes de manera rápida y segura.
- Realizá pruebas en todo el proceso de desarrollo además de contemplarla como una fase del ciclo.
- Escribí pruebas unitarias como parte de tus tareas diarias de desarrollo.
- Asegurate que por cada unidad funcional exista una prueba automatizada como caso de fallo¹¹
- Reducí la deuda técnica refactorizando tu código regularmente.
- Incluí pruebas de seguridad dentro de los requerimientos del producto.
- Automatizá las pruebas funcionales y mantenelas actualizadas.
- Separá adecuadamente los ambientes de desarrollo, prueba, aceptación y producción y llevá a cabo pruebas de integración continua con herramientas de automatización¹²
- Escribí y mantené actualizado tu plan de pruebas unitarias, de integración y punta a punta, y aseguráte de integrarlos y ejecutarlos desde una herramienta de integración continua.

Mínimo

- Desarrollá tus sistemas orientados a pruebas o Test-Driven-Development
- Llevá a cabo pruebas unitarias para las funcionalidades de tu código
- Reducí la deuda técnica refactorizando tu código regularmente
- Incluí pruebas de seguridad dentro de los requerimientos del producto

Por dónde empezar

- Realizá un Plan de Pruebas y registra su ejecución y resultados para cada unidad funcional
- Escribí casos de prueba unitarios para tus nuevas funcionalidades
- Incorporá la refactorización como parte de tus tareas diarias de desarrollo

¹¹ Recomendamos utilizar frameworks de pruebas automatizadas como [Cucumber](#), [Selenium](#) y [JUnit](#)

¹² Por ejemplo, [Jenkins](#), [Circle CI](#), [Travis CI](#), [GitLab](#)

- Utilizó herramientas de inspección¹³ de código para aumentar la calidad semántica

5. Protegé al software y a los usuarios

Mantener la seguridad del software es una responsabilidad que sólo puede cumplirse con el esfuerzo conjunto de todas las partes involucradas en el ciclo de vida del mismo..

Ideal

- Involucrá tempranamente al equipo de seguridad informática en tu equipo u organismo
- Entendé los requerimientos de disponibilidad y rendimiento de tu software
- Revisá el diseño de tu software con la ayuda de un especialista de seguridad.
- Aplicá los principios de diseño seguro de software [Recomendaciones de Seguridad en el Desarrollo de Software](#)
- Considera las recomendaciones de la Guía de seguridad para aplicaciones
- Tené en cuenta la normativa vigente respecto a la [Política de Seguridad de la Información Modelo](#) (dispo ONTI 1/2015)
- Cumplí con la [Ley de Protección de los Datos Personales](#) vigente
- Publicá el código de tu aplicación, recibí mejoras de terceros.
- En tu desarrollo basado en pruebas, incluí pruebas de seguridad.
- Elegí herramientas y librerías con soporte activo y aptas para producción
- Planificá para mantener la seguridad durante toda la vida útil de tu software
- Ofrecé un canal para recibir reportes de vulnerabilidades de la comunidad experta

Mínimo

- Involucrá tempranamente al equipo de seguridad informática de tu equipo u organismo
- Entendé los requerimientos de disponibilidad y rendimiento de tu solución.
- Aplicá los principios de diseño seguro de software [Recomendaciones de Seguridad en el Desarrollo de Software](#)
- Cifrá las comunicaciones de tu software.
- Validá todas las entradas a tu aplicación.
- Realizá pruebas de seguridad sobre tu aplicación antes de entrar en producción..
- Minimiza la cantidad de información sensible de tus usuarios que almacenes

¹³ [Codeclimate](#), [Sonarqube](#)

Por dónde empezar

- Consultá el [Decálogo Tecnológico ONTI #8 - Protegé al sistema y a los usuarios](#).
- Consultá la Guía rápida [“Consideraciones importantes para la Seguridad de Aplicaciones”](#)

6. Utilizá metodologías ágiles e iterativas en todo el ciclo de vida del software

Un paradigma flexible y adaptable a los cambios permite abordar integralmente el desarrollo de productos digitales adecuados a las necesidades de los usuarios, los recursos y las capacidades de tu equipos maximizando la entrega de valor y minimizando los costos.

Ideal

- Empezá construyendo una visión compartida de tu producto con tu equipo, los interesados y comprendé las necesidades de tus usuarios
- Construí una hoja de ruta a partir de usuarios ideales que respondan a necesidades reales
- Definí en conjunto con el equipo el objetivo de cada finalización de ciclo de trabajo que oriente el desarrollo de tus productos
- Priorizá las entregas de valor y las tareas necesarias para desplegar tus productos y servicios
- Desarrollá [prototipos y versiones mínimas de tu producto para validar ideas](#)
- Planificá tus entregas y [reflexioná periódicamente sobre los procesos y los productos](#)
- Incorporá técnicas de [DevSecOps](#) y [DevOps](#) para [entregar valor frecuentemente](#) de forma segura
- Construí un equipo multidisciplinario y lo más autónomo posible para evitar dependencias
- Desarrollá las capacidades del equipo utilizando [programación por pares](#)
- Visibilizá las tareas y el flujo de trabajo para que puedan ser compartidos con tu equipo

Mínimo

- Entregá valor frecuentemente a tus usuarios desarrollando [prototipos y versiones mínimas de tu producto para validar ideas](#)
- Reflexioná sobre los [procesos y los productos que entrega tu equipo](#)
- Colaborá con tu equipo y otros miembros de la organización para generar una cultura de confianza

- Mejorá continuamente tus productos y servicios realizando experimentos planteándote hipótesis a validar con datos cualitativos y cuantitativos
- Construí un equipo multidisciplinario y lo más autónomo posible
- Visibilizá las tareas y el flujo de trabajo para que puedan ser compartidos con tu equipo
- Priorizá los requerimientos de tu software utilizando roles como el de encargado de producto y que conozca la visión del mismo
- Consensuá con tu equipo la definición de trabajo terminado para gestionar las expectativas de tus productos.

Por dónde empezar

- Hací los [cursos de Agilidad y Scrum en el INAP](#)
- Priorizá los requerimientos de tu software utilizando roles como el de encargado de producto y que conozca la visión del mismo
- Construí un equipo multidisciplinario y lo más autónomo posible
- Explorá herramientas de código abierto que trabajen en la nube que apoyen la construcción de software con dinámicas iterativas e incrementales
- Reflexioná sobre los [procesos y los productos que entrega tu equipo](#)

7. Documentá y compartí el conocimiento de tus sistemas

Documentar correctamente aumenta la sustentabilidad de aplicación, ya que facilita que otras personas la instalen, mantengan y utilicen.

Ideal

- Compartí toda la información sobre tus productos para que sea lo más clara posible y disponible en el [Repositorio Público](#)
- Creá documentación tanto para personal técnico como para usuarios nuevos
- Documentá el código mediante comentarios y entidades comprensibles
- Utilizá alguna herramienta de documentación automática¹⁴
- Automatizá el versionado de tu software
- Registrá los eventos de tus sistemas con herramientas abiertas¹⁵

¹⁴ Por ejemplo [Doxygen](#) o [Javadoc](#)

¹⁵ [ElasticSearch 5.5](#), [Logstash](#), [Kibana](#), [Sentry](#)

Mínimo

- Escribí instrucciones detalladas de instalación y probá que los pasos te lleven a instalar correctamente el sistema
- Si estás reutilizando un software, documentá exhaustivamente todos los cambios que realices al código fuente, para que tu equipo o algún equipo futuro los pueda reproducir en caso de actualizar el software
- Utilizá alguna herramienta de seguimiento de defectos¹⁶
- Incorporá algún sistema de versionado para tu software¹⁷
- Mantené informados a tus usuarios respecto a actualizaciones y cambios en el sistema

Por dónde empezar

- Mirá y toma como referencia la documentación de otros proyectos oficiales, como [Poncho](#) y [Autenticar](#)
- Visibilizá tu proyecto a través de una página pública con enlaces a otros recursos relevantes¹⁸
- Defini al comienzo del proyecto todos los entregables y documentos que deberá disponerse con la finalización del desarrollo del software.

¹⁶ Por ejemplo, utilizando [Taiga](#), [Redmine](#), [GitLab](#) o [GitHub](#) para software subido al [repositorio oficial en GitHub](#), o [Phabricator](#) para proyectos más grandes y complejos

¹⁷ Por ejemplo utilizando [GitHub](#), [BitBucket](#), [GitLab](#).

¹⁸ Puede ser una página en [Argentina.gov.ar](#), un README en el [repositorio oficial en GitHub](#), o una [GitHub Page](#)