Plan de Gestión de Software - Asistente virtual para Ciencia Abierta

Patricia A. Loto

2025-08-25

Tabla de contenidos

1 Introducción

Este documento presenta el Plan de Gestión de Software para el proyecto de investigación [**NOMBRE DEL PROYECTO**]. El objetivo es establecer las mejores prácticas para el desarrollo, documentación, preservación y difusión del software desarrollado durante la investigación.

---

2 1. Descripción del Software

2.1 Nombre del proyecto

[Escribe el nombre del software o del proyecto de investigación]

2.2 Identificación

[Describe de forma breve qué hace el software, su propósito y los problemas que resuelve.]

Tabla 1: Información General del Software

Aspecto	Detalle
Nombre del Software	AS Ciencia Abierta
Versión Inicial	[A completar]
Propósito Principal	[A completar]
Usuarios Objetivo	[A completar]
Dominio de Aplicación	[A completar]

2.3 Lenguaje(s) de programación y tecnologías

[Enumera los lenguajes, librerías, frameworks y otras tecnologías que se utilizarán]

Ejemplos:

- R (versión 4.3+)
- Python 3.9+
- Librerías principales: tidyverse, ggplot2, shiny
- Base de datos: PostgreSQL
- Frameworks: Quarto, RMarkdown

2.4 Estado de desarrollo

☐ Software nuevo
☐ Basado en proyecto existente

☐ Versión beta
☐ En producción
☐ Otro: _____

---

3 2. Requisitos de Desarrollo

3.1 Control de versiones

Sistema: [Git / SVN / Mercurial]
Repositorio principal: [URL del repositorio]

Tabla 2: Estrategia de Ramas en Git

Rama	Propósito
main	Código estable de producción
develop	Integración de nuevas características
feature/*	Desarrollo de características específicas
hotfix/*	Correcciones urgentes

3.2 Entorno de desarrollo

[Describe cómo se gestionará la reproducibilidad del entorno]

Herramientas recomendadas: - `renv` para gestión de dependencias en R - Docker para containerización - `conda` environments para Python - Virtual environments

```
# Ejemplo de configuración de entorno R
install.packages("renv")
renv::init()
renv::snapshot()
```

3.3 Gestión de dependencias

[Menciona cómo se registrarán las dependencias del proyecto]

- **R:** Archivo `renv.lock`
- **Python:** `requirements.txt` o `environment.yml`
- **Sistema:** Dockerfile

---

4 3. Calidad y Pruebas

4.1 Pruebas del código

[Describe el plan de pruebas]

Tabla 3: Estrategia de Pruebas

Tipo.de.Prueba	Herramienta	Frecuencia
Pruebas Unitarias	testthat (R), pytest (Python)	Cada commit
Pruebas de Integración	Integración continua (GitHub Actions)	Antes de cada merge
Pruebas de Sistema	Pruebas manuales y automatizadas	Antes de cada release
Pruebas de Rendimiento	Benchmarking y profiling	Mensual o por milestone

4.2 Documentación del código

[Explica cómo se documentará el código internamente]

Estándares: - Uso de docstrings/roxygen2 en cada función - Comentarios inline para lógica compleja - Documentación de APIs - Ejemplos de uso

```
#' Función ejemplo con documentación roxygen2
#'
#' @param x Vector numérico de entrada
#' @param method Método de cálculo ("mean" o "median")
#' @return Valor calculado
#' @examples
#' calcular_estadistica(c(1,2,3,4,5), "mean")
#' @export
calcular_estadistica <- function(x, method = "mean") {
  # Implementación de la función
}
```

4.3 Documentación para el usuario

[Detalla cómo se creará la documentación para usuarios]

Componentes:

- README.md detallado
- Guías de instalación
- Tutoriales paso a paso
- Documentación de API
- FAQ y troubleshooting

---

5 4. Colaboración y Acceso

5.1 Licencia de software

Licencia seleccionada: [MIT / GPL v3 / Apache 2.0 / BSD / Otra]

Tabla 4: Comparación de Licencias de Software

Licencia	Permisividad	Uso.Comercial	Requiere.Atribución
MIT	Muy permisiva	Sí	Sí
Apache 2.0	Permisiva	Sí	Sí
GPL v3	Copyleft	Sí*	Sí
BSD 3-Clause	Permisiva	Sí	Sí

5.2 Repositorio público

Plataforma: [GitHub / GitLab / Bitbucket]
URL: [URL del repositorio público]

5.3 DOI para citación

Servicio: [Zenodo / figshare / Otros]
DOI: [Se asignará al momento de la publicación]

---

6 5. Preservación y Difusión

6.1 Plan de archivo a largo plazo

[Describe estrategias de preservación]

Tabla 5: Estrategia de Preservación

Estrategia	Frecuencia	Responsable
Repositorio institucional	Cada release mayor	[Nombre]
Zenodo	Cada release	[Nombre]
Software Heritage	Automático	Automático
Backup local	Semanal	[Nombre]

6.2 Plan de difusión

[Enumera estrategias de difusión]

Canales de difusión:

- ☐ Publicación en JOSS (Journal of Open Source Software)
- ☐ Presentación en conferencias científicas
- ☐ Redes sociales académicas (ResearchGate, Academia.edu)
- ☐ Blog posts y artículos divulgativos
- ☐ Workshops y seminarios
- ☐ Colaboraciones con otros investigadores

---

7 6. Cronograma y Hitos

Tabla 6: Cronograma de Desarrollo

Hito	Fecha.Objetivo	Estado
Configuración inicial del proyecto	[Fecha]	[ ] Pendiente
Primera versión funcional (MVP)	[Fecha]	[ ] Pendiente
Implementación de pruebas	[Fecha]	[ ] Pendiente
Documentación completa	[Fecha]	[ ] Pendiente
Primera release pública	[Fecha]	[ ] Pendiente
Publicación científica	[Fecha]	[ ] Pendiente

---

8 7. Recursos y Presupuesto

8.1 Recursos humanos

- **Desarrollador principal:** [Nombre y dedicación]
- **Colaboradores:** [Lista de colaboradores]
- **Revisores de código:** [Nombres]

8.2 Recursos técnicos

- **Hardware:** [Especificaciones necesarias]
- **Software:** [Licencias y herramientas]
- **Servicios cloud:** [Si aplica]

Tabla 7: Estimación de Recursos

Concepto	Costo.Estimado	Notas
Tiempo de desarrollo	[Horas × tarifa]	Tiempo del investigador principal
Servicios cloud	[/mes]||*GitHubPro, servidores*||*Si aplica*||*Licencias software propietario*||[/año]	
Hardware adicional	[$ total]	Hardware especializado si es necesario

---

9 8. Evaluación y Métricas

9.1 Métricas de calidad

- Cobertura de pruebas: $> 80\%$
- Documentación: Todas las funciones públicas documentadas
- Issues abiertas: $< 10\%$ del total
- Tiempo de resolución de bugs: < 1 semana

9.2 Métricas de impacto

- Número de descargas/clones
- Citas académicas
- Contribuciones externas
- Uso en otros proyectos

---

10 Notas y Observaciones

[Espacio para notas adicionales, consideraciones especiales, o información relevante específica del proyecto]

Fecha de última actualización: 2025-08-25
Versión del documento: 1.0
Próxima revisión: [Fecha de próxima revisión programada]

Este documento es un documento vivo que debe ser revisado y actualizado regularmente durante el ciclo de vida del proyecto.