

**TECNOLÓGICO NACIONAL DE MÉXICO**

**Instituto Tecnológico de Oaxaca**

**Gestión de Proyectos de Software**



**2.6 Segundo Avance del Proyecto Integrador**

**Unidad II**

**Docente:**

Dra. Eva Rafael Pérez

**Equipo 2:**

Solano Ramos Eduardo 22161254

Mendoza Chávez Jesús Abraham 22161168

Porras Avendaño Sergio Ezequiel 22161208

Miranda Virgen Sabas Mijaíl 22161171

Placido Martínez Cristian Gerardo 22161206

Fernández Crisanto Luis Gael 22161056

**Grupo:** 8SA

Oaxaca de Juárez a 27 de Octubre de 2025.

# **Capítulo II. MARCO TEÓRICO**

**2.1 Gestión de proyectos de software:**

Es un conjunto de actividades que permiten aplicar los procesos, métodos y herramientas de la ingeniería de software de manera controlada y medible, haciendo hincapié en la planificación detallada, la gestión de riesgos proactiva, la estimación y el aseguramiento de la calidad (pressman, 2010).

**2.2 Metodologías de proyectos de software:**

**2.2.1 ¿Qué es una metodología de gestión de proyectos de software?**

Una metodología de gestión de proyectos de software constituye un marco de trabajo estructurado que define sistemáticamente los procesos, prácticas, roles y artefactos necesarios para planificar, ejecutar, controlar y finalizar un proyecto de desarrollo de software. Según el Project Management Institute (2017), un proyecto es un esfuerzo temporal emprendido para crear un producto, servicio o resultado único, y la metodología proporciona el "cómo" gestionar dicho esfuerzo de manera estandarizada. Esta estructura sirve como un plano integral que guía al equipo a lo largo de todo el ciclo de vida del desarrollo, optimizando el uso de recursos y aumentando la predictibilidad y las probabilidades de éxito.

**2.2.2 Objetivos de una metodología de proyectos de software:**

1. Maximizar la Eficiencia y la Productividad: La estandarización de procesos busca optimizar el uso de recursos humanos, económicos y temporales para obtener los mejores resultados posibles (Pressman & Maxim, 2020).
2. Entregar Valor al Cliente: El objetivo primordial es asegurar que el producto final satisfaga los requisitos y expectativas del cliente, resolviendo efectivamente su problema o necesidad subyacente.
3. Mantener el Control del Proyecto: La metodología provee mecanismos para monitorear de cerca el progreso, el presupuesto y el alcance, permitiendo al gestor tomar acciones correctivas de manera oportuna (Project Management Institute, 2017).
4. Gestionar Proactivamente los Riesgos: Un marco estructurado facilita la identificación, análisis y mitigación continua de los riesgos potenciales que puedan impactar negativamente en el proyecto (Pressman & Maxim, 2020)..De
5. Garantizar la Calidad del Producto: Establece procesos sistemáticos de aseguramiento de la calidad para verificar que el software sea confiable, funcione correctamente y cumpla con los estándares definidos (Pressman & Maxim, 2020)..
6. Facilitar la Comunicación y Colaboración: Al definir roles, responsabilidades y canales de comunicación claros para todo el equipo y las partes interesadas), se minimizan los malentendidos y se fomenta el trabajo en equipo (Project Management Institute, 201

**2.3 - Tipos de Metodología de Gestión de Proyectos (**[**clic aquí para ver cuadro**](https://docs.google.com/document/d/1wvkvSjZOuXj8haZh0bWfM166Av-y5hZQQYQsc3QlXio/edit?tab=t.gz26agbq99yv#heading=h.evc3kdtczrjm)**)**

**2.3.1 Metodología adoptada: PM²:**

La metodología PM² fue seleccionada para la gestión de este proyecto debido a que garantiza un marco metodológico robusto y diseñado específicamente para facilitar la gestión eficaz de proyectos en diversos tipos de organizaciones. Se centra en responder a la naturaleza interdisciplinaria del proyecto, que integra diversos contextos como la epidemiología, la ciencia de datos y el desarrollo de software, ofreciendo un enfoque estructurado pero lo suficientemente ágil para adaptarse a esta complejidad. Asimismo, PM² proporciona una gobernanza clara mediante la definición explícita de roles, procesos y resultados. Esta metodología permite una gestión efectiva de las expectativas de los distintos interesados, incluyendo académicos, técnicos y potenciales usuarios del sector salud, asegurando una comunicación y alineación continuas. Además, PM² facilita el control del ciclo de vida completo del proyecto, desde la conceptualización del modelo predictivo hasta el despliegue funcional de la aplicación web. Finalmente, los "Mindsets" orientados a resultados que incorpora PM² promueven mentalidades y prácticas que favorecen la obtención de resultados tangibles y de valor.

**2.4 Metodología PM²:**

**2.4.1 Definición de la metodología PM²:**

PM² es una metodología de gestión de proyectos desarrollada por la Comisión Europea. Su objetivo es permitir a los Directores de Proyecto (DP) ofrecer soluciones y beneficios a sus organizaciones mediante una gestión eficaz a lo largo del ciclo de vida de su proyecto. PM² ha sido creada teniendo en cuenta las necesidades de los proyectos e instituciones de la Unión Europea, pero es transferible a los proyectos de cualquier organización. PM² es una metodología sencilla y fácil de implementar que los equipos de proyecto pueden adaptar a sus necesidades específicas. PM² se encuentra respaldada por un programa de capacitación integral (que incluye talleres y sesiones de coaching), documentación en línea y una Comunidad de Práctica activa (actualmente sólo disponible dentro de la Comisión Europea y para diversas Instituciones Europeas afiliadas). PM² incorpora elementos de numerosas mejores prácticas de gestión de proyectos globalmente aceptadas, plasmadas en estándares y metodologías. Su desarrollo también ha sido influenciado por la experiencia operativa en proyectos tanto dentro de las instituciones de la Unión Europea como en organismos externos (Comisión Europea Centro de Excelencia en PM2 (CoEPM2), 2021).

**2.4.2 Características de PM²:**

**La Metodología PM² proporciona:**

* Una estructura de gobernanza del proyecto.
* Directrices de procesos.
* Plantillas de artefactos.
* Directrices para usar los artefactos. x Un enfoque eficaz.

**PM² mejora la efectividad de la gestión de proyectos porque:**

* Mejora la comunicación y la difusión de información.
* Aclara las expectativas del proyecto de manera rápida y temprana.
* Define el ciclo de vida del proyecto (desde el Inicio hasta el Cierre).
* Proporciona pautas para la planificación del proyecto.
* Introduce actividades de seguimiento y control.
* Propone la gestión de actividades y resultados (planes, reuniones, decisiones).
* Proporciona un enlace a prácticas ágiles (PM² Ágil).

(Comisión Europea Centro de Excelencia en PM2 (CoEPM2), 2021).

**2.4.3 Fases de PM²:**

El ciclo de vida de los proyectos PM² tiene cuatro fases, con un tipo diferente de actividad predominante en cada fase (p. ej. las actividades de inicio son predominantes en la Fase de Inicio, etc.). Sin embargo, mientras que las actividades relacionadas con una fase alcanzan su punto máximo en términos de esfuerzo durante esa fase específica, las actividades de este tipo también se pueden ejecutar durante las fases contiguas (p. ej. las actividades de planificación se dan también en la Fase de Ejecución). Un proyecto pasa a la siguiente fase cuando se considera que se han alcanzado los objetivos de la fase actual, tras una revisión formal (o menos formal) de fin de fase.

1. Inicio: Define los resultados deseados. Crea un Caso de Negocio. Define el alcance del proyecto. Dota al proyecto de un buen comienzo.
2. Planificación: Asigna el Equipo Central del Proyecto (ECP). Desarrolla el alcance del proyecto. Planifica el trabajo
3. Ejecución: Coordina la ejecución de los planes del proyecto. Produce entregables.
4. Cierre: Coordina la aceptación formal del proyecto. Informa sobre el progreso del proyecto. Captura las lecciones aprendidas y las recomendaciones post-proyecto. Cierra el proyecto administrativamente.

(Comisión Europea Centro de Excelencia en PM2 (CoEPM2), 2021).

**2.5 Plan de calidad**

**2.5.1 Definición:**

Un Plan de Calidad, también conocido como Plan de Aseguramiento de la Calidad, es un documento formal que describe las actividades, estándares, recursos y secuencias de eventos específicos necesarios para garantizar que un producto o servicio cumpla con los requisitos establecidos (Project Management Institute, 2017). En el contexto del desarrollo de software, este plan funciona como un modelo operativo que detalla los procesos de calidad a seguir, los criterios de aceptación y las responsabilidades del equipo para asegurar que el software final sea confiable y satisfactorio para el usuario final. Según Pressman y Maxim (2020), el plan actúa como la base para el aseguramiento de la calidad, estableciendo los procedimientos técnicos y de gestión que se aplicarán durante todo el ciclo de vida del proyecto.

**2.5.2 Características:**

1. Es un documento vivo y dinámico: No es estático; se actualiza a lo largo del proyecto para reflejar cambios en el alcance, los requisitos o los riesgos identificados.
2. Es específico y medible: Define métricas y criterios de aceptación cuantificables para evaluar la calidad del producto y los procesos, evitando ambigüedades.
3. Establece responsabilidades claras: Asigna roles y responsabilidades específicas para cada actividad de calidad, desde la gestión hasta la ejecución de pruebas, asegurando la rendición de cuentas.
4. Está alineado con los objetivos del negocio: Las metas de calidad definidas en el plan deben apoyar directamente los objetivos generales del proyecto y las necesidades del cliente.
5. Es integral y estructurado: Aborda todos los aspectos de la calidad, incluyendo la calidad de los procesos (cómo se desarrolla el software) y la calidad del producto final (ISO, 2015)..

**2.5.3 Fases:**

La implementación de un Plan de Calidad se despliega a lo largo de las distintas fases del ciclo de vida del software, integrando actividades específicas en cada una de ellas:

1. Fase de planificación: En esta fase inicial, el Plan de Calidad es creado. Se definen los objetivos de calidad, se seleccionan los estándares aplicables (ej., ISO 25010), se establecen las métricas y se identifican las herramientas a utilizar.
2. Fase de Análisis y Diseño: El plan guía las revisiones de los requisitos y los diseños arquitectónicos para asegurar su claridad, completitud y consistencia antes de comenzar la codificación.
3. Fase de Implementación (Codificación): Se ejecutan actividades como la revisión de código (code reviews) para verificar el cumplimiento de estándares de codificación y se realizan pruebas unitarias para validar el correcto funcionamiento de componentes individuales.
4. Fase de Pruebas: Es donde el plan se ejecuta de forma más intensiva. Se llevan a cabo las pruebas de integración, sistema y aceptación definidas en el plan, utilizando los criterios de aceptación preestablecidos para determinar si el software está listo para su liberación.
5. Fase de Liberación y Mantenimiento: Tras el despliegue, el plan puede incluir la monitorización del software en producción para recoger datos sobre su rendimiento y confiabilidad. Además, establece los procedimientos para gestionar la calidad de las actualizaciones y correcciones durante el ciclo de vida del producto.

**2.6 Estándares de calidad MOPROSOFT y CMMI**

**2.6.1 MOPROSOFT**

**2.6.1.1 Definición**

MoProSoft es un Modelo de Referencia de Procesos conformado por un conjunto de buenas prácticas, métodos y procesos de gestión e ingeniería de software (Oktaba et al., 2005). El modelo es una Norma Mexicana (NMX-I-059-NYCE) diseñado específicamente para que las organizaciones, particularmente las Pequeñas y Medianas Empresas (PyMEs) mexicanas dedicadas al desarrollo y/o mantenimiento de software, mejoren su forma de trabajar, eleven su capacidad y aumenten su competitividad (NYCE, 2016).

**2.6.1.2 Características**

* Enfoque específico: Centrado en el desarrollo y mantenimiento de software.
* Alcance regional: Modelo mexicano (Norma NMX-I-059-NYCE), ideal para el mercado local.
* Público objetivo: Diseñado para la estructura y recursos de las PyMEs mexicanas.
* Estructura concisa: Se basa en solo 9 procesos agrupados en 3 categorías (Alta Dirección, Gerencia, Operación).

**2.6.1.3 Ventajas**

* Bajo costo de adopción: Está diseñado para ser significativamente más económico de implementar y evaluar que modelos globales como CMMI, lo que lo hace ideal para PyMEs.
* Facilidad de comprensión y adopción: Su estructura concisa (9 procesos) y su lenguaje son más sencillos y prácticos, acelerando la curva de aprendizaje en empresas pequeñas.
* Alineación con la estructura organizacional: Las categorías (Alta Dirección, Gerencia, Operación) se ajustan a la estructura funcional típica de las organizaciones mexicanas.
* Compatibilidad normativa: Facilita el cumplimiento simultáneo de requisitos de otros modelos y normas internacionales, como ISO 9001 o ISO/IEC 29110.
* Mejora de la calidad local: Eleva la capacidad y calidad del *software.* Desarrollado a nivel local, mejorando la competitividad en el mercado nacional.

**2.6.1.4 Niveles de madurez**

MoProSoft utiliza Niveles de Capacidad para evaluar sus procesos, siguiendo el estándar ISO/IEC 15504 (SPICE).

* Nivel 0: Incompleto
  + El proceso no está implementado o no logra su propósito.
* Nivel 1: Realizado
  + El proceso se implementa y alcanza su propósito de manera básica.
* Nivel 2: Administrado
  + El proceso realizado es planificado, supervisado y sus productos de trabajo son controlados.
* Nivel 3: Establecido
  + El proceso gestionado se ejecuta usando un proceso definido y estandarizado a nivel organizacional.
* Nivel 4: Predecible
  + El proceso establecido se ejecuta dentro de límites definidos y su desempeño es cuantitativamente predecible.
* Nivel 5: Optimizado
  + El proceso predecible se mejora de forma continua para cumplir con los objetivos estratégicos.

**2.6.1.5 Áreas de procesos**

MoProSoft organiza sus 9 procesos en tres categorías funcionales que reflejan la estructura de una organización de desarrollo de software:

### 1. Categoría: Alta Dirección (DIR)

Se centra en la gestión estratégica del negocio.

* DIR.1: Gestión de negocio

### 2. Categoría: Gerencia (GER)

Se centra en la provisión y administración de los recursos y la infraestructura para los proyectos.

* GER.1: Gestión de procesos
* GER.2: Gestión de proyectos
* GER.3: Gestión de recursos
  + *Subprocesos:*
    - Recursos Humanos y Ambiente de Trabajo
    - Bienes, servicios e infraestructura
    - Conocimiento de la organización

### 3. Categoría: Operación (OPE)

Se centra en la ejecución y entrega de los proyectos de software.

* OPE.1: Administración de proyectos específicos
* OPE.2: Desarrollo y mantenimiento de software

**2.6.2 CMMI**

**2.6.2.1 Definición**

CMMI es un marco de mejora de procesos que provee a las organizaciones los elementos esenciales para un proceso efectivo. Se define como un modelo que contiene las mejores prácticas que ayudan a las organizaciones a mejorar su desempeño en el desarrollo de productos y servicios (Chrissis et al., 2007). Es administrado por el CMMI Institute (ISACA) y es reconocido internacionalmente como una guía para la mejora de la capacidad organizacional (CMMI Institute, s.f.).

**2.6.2.2 Características**

* Enfoque integrador: Fusiona múltiples disciplinas (Desarrollo, Servicios, Adquisición) en un solo marco.
* Alcance global: Es reconocido y utilizado a nivel internacional.
* Complejidad y rigurosidad: Es un modelo detallado y riguroso, mejor adaptado a grandes organizaciones.
* Estructura Amplia: Se basa en un conjunto extenso de Áreas de Proceso (ej. 22 PAs en CMMI-DEV).
* Nomenclatura: Utiliza Niveles de Madurez (Inicial, Administrado, Definido, Cuantitativamente Administrado, Optimizado).
* Representación Dual: Ofrece la opción de representación Escalonada (rígida) o Continua (flexible).
* Costo: Alto costo de implementación y evaluación.

**2.6.2.3 Ventajas**

* Reconocimiento Global: Ofrece un prestigio internacional que es crucial para acceder a grandes clientes y licitaciones a nivel mundial.
* Mayor fiabilidad: Aumenta la certeza en las estimaciones de proyectos, lo que resulta en el mejor cumplimiento de fechas y presupuestos.
* Reducción de Costos a Largo Plazo: La mejora de procesos conduce a una reducción significativa en la tasa de defectos y, por ende, en los costos de retrabajo.
* Visibilidad de Procesos: Proporciona un control y seguimiento rigurosos, haciendo que el proceso de desarrollo y el estado del producto sean transparentes para la gerencia.
* Alineación Estratégica: Ayuda a alinear los procesos técnicos y de gestión directamente con los objetivos estratégicos del negocio.
* Mejora Medible: En los niveles superiores (4 y 5), introduce el control estadístico, permitiendo la mejora continua basada en datos y métricas objetivas.

**2.6.2.4 Desventajas**

* Alto costo financiero: La implementación de CMMI, desde la capacitación inicial hasta la evaluación formal (SCAMPI), es notablemente costosa en comparación con modelos alternativos (como MoProSoft).
* Larga duración de la implementación: Alcanzar los niveles más altos (Nivel 4 y 5) requiere un compromiso de tiempo extenso (a menudo de varios años), lo que puede generar fatiga en el equipo.
* Complejidad y rigidez: El modelo es muy detallado y extenso, lo que puede dificultar su comprensión y adopción, especialmente en Pequeñas y Medianas Empresas (PyMEs) o en organizaciones que utilizan metodologías ágiles.
* Resistencia al cambio: La implementación formal de CMMI a menudo implica una reingeniería profunda de los procesos existentes, lo que puede provocar una fuerte resistencia por parte del personal operativo si no hay una gestión de cambio adecuada.
* Riesgo de "Certificación por Papeles": Existe el riesgo de que la organización se centre únicamente en generar la documentación requerida por el modelo para aprobar la auditoría, en lugar de internalizar y mejorar genuinamente los procesos.
* Necesidad de recursos especializados: Requiere la contratación o dedicación de personal con conocimientos específicos en CMMI (consultores, evaluadores certificados, y grupos de mejora de procesos como el SEPG), aumentando la dependencia de expertos externos.

**2.6.2.5 Niveles de madurez**

CMMI (Capability Maturity Model Integration) define 5 Niveles de Madurez para clasificar a una organización en función de su capacidad para gestionar y mejorar sus procesos.

* Nivel 1: Inicial (Initial)
  + Enfoque: Caótico e impredecible.
  + Descripción: Los procesos son *ad hoc* (improvisados). El éxito depende de los "héroes" y no de los procesos definidos. El ambiente es reactivo.
* Nivel 2: Administrado (Managed)
  + Enfoque: Gestionado a nivel de proyecto.
  + Descripción: Los proyectos se planifican, ejecutan, se supervisan y controlan. Se gestionan los requisitos, la configuración y se realizan estimaciones básicas.
* Nivel 3: Definido (Defined)
  + Enfoque: Estandarizado a nivel organizacional.
  + Descripción: Los procesos son estandarizados, documentados y se adaptan a partir de los procesos estándar de la organización (conjunto de procesos organizacionales).
* Nivel 4: Cuantitativamente Administrado (Quantitatively Managed)
  + Enfoque: Medido y control estadístico.
  + Descripción: La organización utiliza métricas estadísticas y objetivos cuantitativos para controlar el rendimiento de los procesos. El desempeño es predecible.
* Nivel 5: Optimizado (Optimizing)
  + Enfoque: Mejora continua e innovación.
  + Descripción: La organización se enfoca en la mejora continua del desempeño del proceso mediante la aplicación de análisis de causas de variación y la introducción de tecnologías y métodos innovadores.

**2.6.2.6 Áreas de procesos**

### Gestión de proyectos (Project Management): Estas áreas se enfocan en la planificación, monitoreo y control de los proyectos específicos.

* PP (Project Planning) - Planeación de Proyectos
* PMC (Project Monitoring and Control) - Supervisión y Control de Proyectos
* SAM (Supplier Agreement Management) - Gestión de Acuerdos con Proveedores
* I PM (Integrated Project Management) - Gestión Integrada de Proyectos (Nivel 3)
* RSKM (Risk Management) - Gestión de Riesgos

### Soporte (Support): Áreas que están dedicadas a prácticas que dan soporte al proyecto y a la organización en general.

* CM (Configuration Management) - Gestión de configuración
* MA (Measurement and Analysis) - Medición y Análisis
* PPQA (Process and Product Quality Assurance) - Aseguramiento de calidad de proceso y producto
* DAR (Decision Analysis and Resolution) - Análisis y Resolución de Decisiones (Nivel 3)
* CAR (Causal Analysis and Resolution) - Análisis Causal y Resolución (Nivel 5)

### Categoría: Ingeniería (Engineering): Estas cubren el desarrollo y el mantenimiento del producto o servicio, desde los requisitos hasta la entrega.

* RD (Requirements Development) - Desarrollo de requisitos
* REQM (Requirements Management) - Gestión de requisitos
* TS (Technical Solution) - Solución Técnica
* PI (Product Integration) - Integración de producto
* VER (Verification) - Verificación
* VAL (Validation) - Validación

### Categoría: Gestión de procesos (Process Management): Se enfocan en la definición, despliegue y mejora de los procesos a nivel organización.

* OPF (Organizational Process Focus) - Enfoque en el proceso organizacional
* OPD (Organizational Process Definition) - Definición del Proceso Organizacional
* OT (Organizational Training) - Formación Organizacional
* OPP (Organizational Process Performance) - Rendimiento del Proceso Organizacional (Nivel 4)
* QPM (Quantitative Project Management) - Gestión Cuantitativa de Proyectos (Nivel 4)
* OID (Organizational Innovation and Deployment) - Innovación y Despliegue Organizacional (Nivel 5)

**2.7 Estimaciones**

**2.7.1 Definición**

Las estimaciones en proyectos son un cálculo aproximado del tiempo, costo y recursos necesarios para completar un proyecto, y se realizan durante la fase de planificación para crear presupuestos y cronogramas realistas. Este proceso ayuda a gestionar expectativas, planificar la capacidad y los recursos, y a evaluar la viabilidad del proyecto desde sus inicios. Una estimación precisa es fundamental para el éxito del proyecto y se apoya en diversas técnicas y en la consideración de factores como los riesgos y las contingencias.

**2.7.2 Tipos**

* Estimación de esfuerzo (personal/trabajo): Predicción de la cantidad total de trabajo necesaria para completar el alcance del proyecto. Se mide generalmente en horas-persona o días-persona, su uso principal es la base para determinar la necesidad de personal y la duración del proyecto. Es el primer paso en muchas metodologías.
* Estimación de Duración (Tiempo): Predicción del tiempo calendario total que tomará completar el proyecto de principio a fin. Se mide en días, semanas o meses; finalmente, ayuda a la creación del cronograma del proyecto y establecimiento de la fecha de entrega (*deadline*).
* Estimación de costo (presupuesto): Predicción de la cantidad total de dinero que se requerirá para ejecutar el proyecto. Incluye salarios, *hardware*, *software*, licencias y otros gastos, gracias a ello se da establecimiento del presupuesto y gestión financiera del proyecto.
* Estimación de Recursos (Materiales y Equipos): Predicción de los activos no humanos necesarios para la ejecución del proyecto, como servidores, herramientas de desarrollo, licencias de software y materiales de oficina, con la intención de planificar la logística e infraestructura esencial para el equipo de desarrollo.

**2.8 Factibilidad del proyecto**

La factibilidad del proyecto es el análisis sistemático que evalúa la viabilidad de una iniciativa desde los puntos de vista técnico, económico, operativo y legal, con el fin de determinar si es posible, conveniente y sostenible su ejecución (PMI, 2017).

#### **2.8.1 Factibilidad Económica**

La factibilidad económica evalúa si los beneficios financieros esperados superan los costos totales del proyecto, analizando retorno de inversión (ROI), valor presente neto (VPN), tasa interna de retorno (TIR) y fuentes de financiamiento (Kerzner, 2017).

#### **2.8.2 Factibilidad Operativa**

La factibilidad operativa analiza si el proyecto puede integrarse y operar eficientemente dentro de la estructura organizacional existente, considerando recursos humanos, procesos, infraestructura y capacidad de mantenimiento (Schwalbe, 2019).

#### **2.8.3 Factibilidad Técnica**

La factibilidad técnica determina si se cuenta con la tecnología, herramientas, conocimientos y experiencia técnica necesarios para desarrollar y desplegar el sistema con los estándares de calidad y funcionalidad requeridos (Sommerville, 2016).

#### **2.8.4 Factibilidad Legal**

La factibilidad legal evalúa el cumplimiento del proyecto con las leyes, normativas, regulaciones y estándares éticos aplicables, incluyendo protección de datos, propiedad intelectual, contratos y permisos (Project Management Institute, 2017).

**2.9 Metodologías de desarrollo de software**

**2.9.1 Definición**

Las metodologías de desarrollo de software son directrices estructuradas que guían el proceso de creación de aplicaciones, desde la concepción original hasta su entrega y mantenimiento. El objetivo de estas metodologías es optimizar la productividad, minimizar los riesgos y asegurar que el producto final sea de calidad. Extreme Programming (XP) es una de las metodologías ágiles más conocidas, la cual pone el foco en la colaboración, la flexibilidad y el suministro ininterrumpido de software que funcione (Beck, 2004).

**2.9.2 Programación extrema (XP)**

XP es una metodología ágil que se enfoca en la adaptación a las modificaciones, el contacto continuo con el cliente y la optimización constante del software a través de breves etapas de desarrollo llamadas iteraciones. Sus principios fundamentales incluyen:

* Desarrollo iterativo e incremental: entrega de funcionalidades en ciclos cortos.
* Pruebas constantes: todas las funcionalidades deben ser verificadas mediante pruebas unitarias.
* Programación en parejas (Pair Programming): dos desarrolladores trabajan juntos en una misma estación de trabajo.
* Refactorización continua: mejora del código sin alterar su funcionalidad para mantener calidad y legibilidad.
* Integración continua: Se combina el software y se comprueba a menudo para identificar errores con rapidez.

**2.9.3 Tipos de metodologías de desarrollo**

Las metodologías de desarrollo se pueden categorizar en:

* Tradicionales o predictivas: como el modelo de cascada, en el que cada etapa se termina antes de iniciar la siguiente.
* Ágiles o adaptables: como Scrum, Kanban y XP, que posibilitan una rápida iteración y una reacción ante cambios constantes.
* Híbridas: fusionan componentes de técnicas ágiles y predictivas para adaptarse a requerimientos particulares del proyecto.

**2.9.4 Cuadro comparativo de metodologías (**[**clic para ver cuadro**](https://docs.google.com/document/d/1wvkvSjZOuXj8haZh0bWfM166Av-y5hZQQYQsc3QlXio/edit?tab=t.xm7t01hwefek#heading=h.f5j32qc6kclu)**)**

**2.9.5 Detalle de la metodología utilizada**

El proyecto actual optó por la programación extrema (XP) porque puede gestionar cambios constantes en los requisitos y se concentra en la calidad del software a través de pruebas continuas y refactorización. Se llevaron a cabo iteraciones cada dos semanas, en las que cada entrega contiene funcionalidades completas y testadas, asegurando así un producto funcional en cada ciclo. La programación en parejas fomenta el trabajo en equipo y la revisión continua del código, a la vez que la integración continua permitió identificar y corregir fallos de manera inmediata.

XP, al ser ágil, también posibilitó que el cliente diera su opinión de manera continua, lo cual permitió que se modificaran los requerimientos y mejorará la satisfacción del usuario final. Esto es esencial en contextos donde los requerimientos cambian rápidamente.

**2.10 Base de datos.**

“Una base de datos es una colección organizada de datos estructurados, almacenados electrónicamente y gestionados mediante un sistema de gestión de bases de datos (SGBD), que permite la creación, consulta, actualización y administración de los datos de forma eficiente, segura y concurrente.” (Codd, 1970, p. 377).

**2.10.1 Diagrama Entidad-Relación**

El Modelo Entidad-Relación (ER) es un modelo conceptual de datos utilizado para representar la estructura lógica de una base de datos de forma gráfica y abstracta. Fue propuesto por Peter Chen en 1976 y se basa en tres componentes principales:

* **Entidades**: Objetos o conceptos del mundo real con existencia independiente (ej. *Estudiante*, *Curso*).
* **Atributos**: Propiedades que describen a las entidades (ej. *Nombre*, *Edad*).
* **Relaciones**: Asociaciones entre entidades (ej. *Matrícula*, *Imparte*).

Se representa mediante diagramas ER (rectángulos para entidades, óvalos para atributos y rombos para relaciones).

**2.10.2 Diagrama Relacional**

El Modelo Relacional es un modelo de datos formal basado en la teoría de conjuntos y la lógica de primer orden, que representa los datos como tablas (relaciones) compuestas por tuplas (filas) y atributos (columnas). Fue propuesto por Edgar F. Codd en 1970 y se fundamenta en:

* Relaciones con claves primarias y foráneas.
* Integridad referencial y de entidad.
* Operaciones algebraicas (selección, proyección, unión, etc.).

Es la base de los sistemas de gestión de bases de datos relacionales (RDBMS) como MySQL, PostgreSQL, Oracle, etc.

**2.11 - gestores de base de datos (**[**clic para ver cuadro comparativo**](https://docs.google.com/document/d/1wvkvSjZOuXj8haZh0bWfM166Av-y5hZQQYQsc3QlXio/edit?tab=t.nh3det4vp32v#heading=h.zf5wvt47gmoj)**)**

**2.12 - frameworks(**[**clic para ver cuadro comparativo)**](https://docs.google.com/document/d/1wvkvSjZOuXj8haZh0bWfM166Av-y5hZQQYQsc3QlXio/edit?tab=t.muk61m9wcb7c)

**2.13 - Tecnologías a utilizar**

Justificación:

Se elige React por su eficiencia en el manejo del DOM virtual, su facilidad para crear interfaces dinámicas y su compatibilidad con bibliotecas externas. Permite desarrollar aplicaciones modulares y reutilizables, lo que acelera la productividad en proyectos de front-end. Además, su comunidad activa proporciona soporte y actualizaciones constantes.

Desarrollo:

React facilita la construcción de componentes reutilizables que representan partes de la interfaz como botones, formularios o tarjetas. A través del Virtual DOM, solo se actualizan los elementos necesarios en pantalla, mejorando el rendimiento. Combinando React con herramientas como Vite o Next.js, se obtienen entornos de desarrollo más rápidos y adaptados al SEO.

**CAPÍTULO III: DESARROLLO DE ACTIVIDADES**

**3.1. Introducción**

La metodología PM² será empleada para la gestión y desarrollo del proyecto integrador titulado "Aplicación Web para la Predicción del Comportamiento de Enfermedades Virales". PM² es un marco metodológico desarrollado por la Comisión Europea, diseñado para facilitar la gestión eficaz de proyectos dentro de diferentes tipos de organizaciones, favoreciendo la claridad en roles, procesos y resultados.

La elección de PM² no es casual; responde directamente a la naturaleza interdisciplinaria y la complejidad inherente de este proyecto. Al fusionar dominios tan distintos como la epidemiología, la ciencia de datos (específicamente el aprendizaje automático) y el desarrollo de software (aplicaciones web), se genera una alta incertidumbre técnica y de requisitos.

PM², con su enfoque estructurado pero ágil y sus "Mindsets" orientados a resultados, proporciona el andamiaje necesario para navegar esta complejidad. Permitirá establecer una gobernanza clara (definiendo quién toma las decisiones), gestionar las expectativas de los *stakeholders* (académicos, técnicos y potenciales usuarios del área de salud) y controlar el ciclo de vida completo, desde la conceptualización del modelo predictivo hasta el despliegue funcional de la aplicación web.

## **Modelo y autor de PM²:**

La metodología PM² fue desarrollada por la Comisión Europea y puesta a disposición pública para mejorar la gestión de proyectos tanto dentro de la UE como en otros entornos. Su primera publicación formal se realizó en 2007, con varias actualizaciones y adaptaciones posteriores para incluir perspectivas ágiles e híbridas.

## 

## **3. Etapas de la metodología PM²**

El ciclo de vida de un proyecto PM² es un marco estructurado diseñado para guiar un proyecto desde su concepción hasta su finalización exitosa. Se divide en cuatro fases principales, cada una concluida por un "punto de transición" (Gate) que requiere aprobación formal para avanzar, asegurando que el proyecto se mantenga alineado con sus objetivos y su Caso de Negocio.

* Inicio: Definición de la justificación (Caso de Negocio), los resultados deseados y el alcance preliminar (Acta de Constitución). El objetivo es asegurar que el proyecto es viable y vale la pena.
* Planificación: Asignación de roles, desarrollo del alcance detallado, desglose del trabajo (WBS) y creación de todos los planes subsidiarios (cronograma, costes, riesgos).
* Ejecución: Coordinación del equipo y producción de los entregables del proyecto (el software) según lo planificado.
* Cierre: Aceptación formal de los entregables, registro de lecciones aprendidas y cierre administrativo y contractual.

Durante todo el ciclo, existe un proceso omnipresente de Seguimiento y Control, que se encarga de medir el progreso contra la línea base de planificación, gestionar cambios y comunicar el estado a los interesados.

## **3.2. Fase de Inicio (PM²)**

El propósito de esta fase es definir el "qué" y el "por qué" del proyecto, asegurando que existe una justificación clara y un entendimiento común entre los interesados clave antes de comprometer recursos significativos.

### **3.2.1. Creación de la Solicitud de Inicio del Proyecto**

* Solicitante (Project Owner Requestor): El Instituto Tecnológico de Oaxaca (ITO), a través de su Departamento de Sistemas y Computación.
* Necesidad de Negocio: La necesidad de contar con una herramienta tecnológica funcional que permita analizar y predecir el comportamiento de enfermedades virales, aplicando y validando los conocimientos en ciencia de datos y desarrollo web adquiridos en el plan de estudios.
* Resultados Deseados: Un software funcional (aplicación web) basado en modelos predictivos que integre diversas fuentes de datos históricos, ofrezca visualización de resultados mediante interfaces interactivas y genere alertas tempranas para apoyar la toma de decisiones en salud pública a nivel simulado.

### **3.2.2. Creación del Caso de Negocio**

Este artefacto justifica la inversión de tiempo y recursos en el proyecto.

* Contexto del Negocio: El Instituto Tecnológico de Oaxaca (ITO), por medio del departamento de Sistemas y Computación, requiere que los estudiantes desarrollen proyectos integradores que apliquen los conocimientos de la carrera en soluciones del mundo real, demostrando competencias técnicas, metodológicas y profesionales.
* Descripción del problema: Los proyectos académicos a menudo carecen de una aplicación práctica tangible y de una metodología de gestión formal, lo que limita su impacto educativo y reduce la preparación profesional de los estudiantes para enfrentar proyectos complejos.
* Descripción del proyecto: Desarrollo de una aplicación web full-stack utilizando Python para el backend (análisis y API del modelo), JavaScript (framework moderno) para el frontend, y modelos de machine learning para predecir el comportamiento de enfermedades virales. Todo el proceso será gestionado con la metodología PM².
* Análisis de soluciones alternativas:
  1. Proyecto teórico (Investigación Documental): No cumple con los objetivos de integración de conocimientos prácticos de desarrollo de software.
  2. Proyecto práctico sin metodología (Desarrollo Ad-hoc): Alto riesgo de desorganización, "scope creep" (desviación del alcance) y fracaso en la entrega de un producto funcional.
  3. Proyecto con metodología PM² (Solución Seleccionada): Asegura una gestión profesional, visibilidad del progreso, control de calidad y el cumplimiento de los objetivos académicos y técnicos.
* Beneficios Esperados: Entrega de un producto de software validado, cumplimiento del 100% de los requisitos académicos, y generación de experiencia práctica en gestión (PM²) y desarrollo (XP) para el equipo.
* Coste y cronograma: Los costes directos se asocian a servicios de hosting en la nube y licencias de desarrollo (si aplican). Las horas de trabajo del equipo son el recurso principal. El cronograma está alineado con la duración del semestre académico (16 semanas).

### **3.2.3. Creación del Acta de Constitución del Proyecto (Project Charter)**

Documento que autoriza formalmente el proyecto y otorga la autoridad al Director del Proyecto.

* **Objetivos SMART:**
  1. Specific: Desarrollar una aplicación web para la predicción de enfermedades virales.
  2. Measurable: Lograr una precisión mínima del 85% en las predicciones del modelo sobre el conjunto de datos de prueba.
  3. Achievable: Utilizando datos públicos disponibles y las herramientas de software estándar (Python, JS, MySQL).
  4. Relevado: Cumplir con el 100% de los requisitos del Proyecto Integrador.
  5. Time-bound: Entregar la aplicación funcional y la documentación completa antes del 12 de diciembre del 2025.
* **Alcance de alto nivel:**
  1. Incluye: Diseño de diagramas UML, modelo Entidad-Relación, implementación de base de datos en MySQL, desarrollo de API REST (Backend), desarrollo de interfaces de usuario (Frontend) y entrenamiento/validación de modelos predictivos.
  2. Excluye: Recopilación de datos primarios en campo, diagnóstico médico individualizado, o uso clínico real para la toma de decisiones (el proyecto es una prueba de concepto académica).
* **Organización del proyecto:**
  1. Patrocinador: ITO (Departamento de Sistemas y Computación).
  2. Director del Proyecto: Eduardo Solano Ramos.
  3. Equipo Central del Proyecto: Sergio Ezequiel Porras Avendaño, Jesus Abraham Mendoza Chavez, Cristian Gerardo Placido Martinez, Luis Gael Fernandez Crisanto, Sabas Mijail Miranda Virgen.
  4. Comité de Dirección: Asesor académico (M.T.I Eva Rafael Perez).
* **Riesgos Iniciales:**
  1. Datos: Baja calidad o disponibilidad limitada de datos públicos sobre enfermedades.
  2. Técnico: Complejidad en la integración del modelo de ML con la API del backend.
  3. Recursos: Gestión del tiempo académico del equipo (cruce con otras materias).
* **Supuestos:** Se asume que los datos públicos necesarios están disponibles y son suficientes para entrenar un modelo. Se asume la disponibilidad semanal del Asesor para las revisiones.

### **3.2.4. Revisión y Aprobación**

El Comité de Dirección del Proyecto (M.T.I Eva Rafael Perez) y el Patrocinador (ITO) revisaron y aprobaron el Caso de Negocio y el Acta de Constitución. Se autoriza formalmente al Director del Proyecto a consumir recursos para la Fase de Planificación.

## **3.3. Fase de Planificación (PM²)**

El objetivo de esta fase es detallar el "cómo". Se desglosa el trabajo definido en el Acta de Constitución en un plan de acción concreto y realista, creando la línea base del proyecto.

### **3.3.1. Realización de la Reunión de Inicio de Planificación**

Se llevó a cabo una reunión de trabajo con el Equipo Central del Proyecto (ECP) y el Comité de Dirección (CDP) para:

* Comunicar oficialmente el inicio de la fase.
* Revisar a fondo los documentos aprobados (Acta de Constitución y Caso de Negocio) para asegurar un entendimiento total del alcance.
* Establecer los objetivos, metodologías y herramientas específicas para la planificación (p.ej., "Usaremos Trello para el backlog y GitHub Projects para el WBS").
* Asignar responsabilidades para el desarrollo de los distintos componentes del Plan de Trabajo.

### **3.3.2. Creación del Manual del Proyecto**

Este documento centraliza las "reglas del proyecto" para la gestión y ejecución del proyecto.

* **Modelo de Gobernanza (Gestión):** Uso del ciclo de vida PM² para la gestión general, artefactos, comunicación con el CDP y gestión de riesgos.
* **Modelo de Desarrollo (Ingeniería):** Se adopta un modelo híbrido que utiliza Extreme Programming (XP) de *Kent Beck* para la ingeniería de software. PM² gestionará el "qué" (gobernanza) y XP gestionará el "cómo" (desarrollo).
* **Prácticas de XP a Implementar:**
  + El Juego de la Planificación: Las historias de usuario se crearán y priorizarán en Trello.
  + Entregas pequeñas: Se realizarán despliegues incrementales en el entorno de pruebas.
  + Programación en Parejas: Se utilizará en módulos complejos (p.ej., la integración de la API del modelo).
  + Desarrollo Guiado por Pruebas: Se exigirán pruebas unitarias para los endpoints críticos del backend.
  + Integración Continua (CI): Se configurará GitHub Actions para ejecutar pruebas automáticamente en cada push.
* **Roles y responsabilidades:** Confirmación detallada de los roles y responsabilidades del Director del Proyecto (Eduardo Solano Ramos), ECP (Sergio Ezequiel Porras Avendaño, Jesus Abraham Mendoza Chavez, Cristian Gerardo Placido Martinez, Luis Gael Fernandez Crisanto, Sabas Mijail Miranda Virgen) y el CDP (M.T.I. Eva Rafael Pérez).
* **Herramientas de Gestión:**
  + Control de versiones: GitHub.
  + Gestión de Tareas (Backlog): Trello.
  + Comunicación Síncrona: Discord (canal del equipo).
  + Documentación: Google Workspace / Notion.
* Estándares: Diagramas UML, Estándar de codificación (PEP 8 para Python), Normalización de base de datos (Tercera Forma Normal - 3FN).

### **3.3.3. Desarrollo del Plan de Trabajo del Proyecto.**

Se elaboró el plan detallado que guiará la ejecución.

* **Desglose del Trabajo:** Se desglosó el proyecto en paquetes de trabajo principales:
  + 1.0. Gestión del Proyecto (PM²)
  + 2.0. Análisis de Requerimientos (Historias de Usuario)
  + 3.0. Diseño (UML, Modelo E-R, Figma UI)
  + 4.0. Análisis de Datos y Modelado (ML)
    - 4.1. Obtención y Limpieza de Datos
    - 4.2. Análisis Exploratorio de Datos (EDA)
    - 4.3. Entrenamiento y Validación de Modelos
    - 4.4. Creación de API del Modelo
  + 5.0. Desarrollo Backend (API REST y BD)
  + 6.0. Desarrollo Frontend (Interfaces y Visualización)
  + 7.0. Pruebas e Integración
  + 8.0. Despliegue y Puesta en Producción (Configuración de servidor, despliegue con PM2/Gunicorn)
* **Esfuerzo y Costes:** Los costes se limitaron a recursos digitales (dominio, hosting cloud) y licencias de software de desarrollo. El esfuerzo se basa en las horas-estudiante asignadas.
* **Cronograma:** Se desarrolló un diagrama de Gantt que asigna las tareas a lo largo del semestre académico, definiendo hitos clave como la finalización del modelo de Machine Learning, la integración backend-frontend y la revisión final antes de la entrega ([revisar aquí](#_iq4d14noon4i)).

### **3.3.4. Preparación de la Matriz de Partes Interesadas**

Se actualizó y detalló el registro de interesados, definiendo estrategias de comunicación:

* Alto Poder / Alto interés: Comité de Dirección (M.T.I. Eva Rafael Perez), Equipo Central del Proyecto.
  + Estrategia: Gestionar de cerca, involucrarse activamente.
* Alto Poder / Bajo interés: Departamento Académico del ITO (evaluadores finales).
  + Estrategia: Mantener satisfechos, reportar hitos clave.
* Bajo Poder / Alto interés: Instituciones de salud (potenciales beneficiarios de la idea).
  + Estrategia: Mantener informados, usar como fuente de requisitos no funcionales.
* Bajo Poder / Bajo interés: Potenciales usuarios públicos.
  + Estrategia: Monitorear (no se requiere comunicación activa).

### **3.3.5. Creación de otros Planes Importantes**

* Plan de Gestión de las Comunicaciones: Se definieron canales (Discord para el día a día, reuniones semanales de 30 min con el Asesor), formatos de reporte (Reportes de Avance del Proyecto Integrador, actas de reunión) y responsables.
* Plan de Implementación (Despliegue): Se planificó la entrega final, detallando el proceso técnico para mover la aplicación de "desarrollo" a "producción" en un servidor en la nube, incluyendo la configuración de la base de datos y el dominio.
* Plan de Transición: Dado el contexto académico, este plan se centró en la entrega formal (el handover) de la aplicación y su documentación al Departamento de Sistemas y Computación del ITO para su evaluación final.

### **3.3.6. Revisión y Aprobación**

Al finalizar la Fase de Planificación, el Director del Proyecto (Eduardo Solano) presentó el Plan de Trabajo del Proyecto completo (línea base de alcance, tiempo y coste) al Comité de Dirección (M.T.I. Eva Perez). Tras su revisión y aprobación, se autorizó formalmente el inicio de la Fase de Ejecución.

# Cronograma

**3.4 Fase de Ejecución**

En esta fase, el Equipo Central del Proyecto (ECP) realiza el trabajo definido en el Plan de Trabajo para crear los entregables (la aplicación web) usando el modelo de desarrollo XP. El rol del Director del Proyecto (DP) se centra en coordinar, monitorear y controlar este trabajo.

**3.4.1 Etapa 1 de XP. Planeación:**

#### **3.4.1.1. Requerimientos Funcionales**

Módulo de Gestión de Datos

* **RF-01:** El sistema debe permitir la carga de archivos en formato CSV provenientes de fuentes oficiales del sector salud. Esta función debe permitir seleccionar el archivo, verificar su estructura y confirmar los datos a importar antes de procesarlos.
* **RF-02:** El sistema debe validar la integridad y formato de los datos durante la carga, comprobando que los campos contengan el tipo de información correcto y que no existan registros incompletos o duplicados.
* **RF-03:** El sistema debe ejecutar procesos automáticos de limpieza y normalización de datos, corrigiendo valores inconsistentes, eliminando duplicados y estandarizando nombres y formatos.
* **RF-04**: El sistema debe almacenar los datos procesados en una base de datos MySQL con estructura normalizada, conservando registros históricos para consultas y análisis posteriores.

Módulo de Modelado Predictivo

* **RF-05:** El sistema debe implementar un algoritmo de regresión lineal que permita generar predicciones a partir de los datos almacenados.
* **RF-06:** El sistema debe permitir el uso de regresión lineal múltiple para analizar la relación entre varias variables independientes y una variable dependiente.
* **RF-07:** El sistema debe calcular métricas de precisión (como error cuadrático medio o coeficiente de determinación) para evaluar el rendimiento de los modelos predictivos.
* **RF-08:** El sistema debe permitir comparar distintos modelos predictivos, mostrando resultados comparativos de precisión y eficiencia para seleccionar el mejor modelo.

Módulo de Visualización

* **RF-09:** El sistema debe generar gráficos de tendencia temporal interactivos, permitiendo al usuario explorar la evolución de los datos a lo largo del tiempo.
* **RF-10:** El sistema debe mostrar un panel de indicadores en tiempo real, donde se reflejen los valores más recientes de los principales indicadores de salud.
* **RF-11:** El sistema debe contar con un sistema de semáforo de alertas, con tres niveles (verde, amarillo y rojo) que indiquen el estado de riesgo o tendencia.
* **RF-12:** El sistema debe permitir filtrar la información visualizada por enfermedad, región geográfica y rango de fechas, para facilitar el análisis específico.

Módulo de Reportes y Alertas

* **RF-13:** El sistema debe generar reportes descargables en formatos PDF y Excel, con información resumida y visualizaciones gráficas.
* **RF-14:** El sistema debe enviar notificaciones automáticas por correo electrónico cuando se detecten alertas, nuevos reportes o actualizaciones relevantes.

Módulo de Administración

* **RF-15:** El sistema debe permitir la gestión de usuarios, asignando diferentes perfiles y permisos según el rol (administrador, analista, invitado, etc.).
* **RF-16:** El sistema debe registrar una bitácora de actividades, donde se almacenen acciones relevantes del sistema, como accesos, modificaciones o cargas de datos.
* **RF-17:** El sistema debe garantizar la confidencialidad y seguridad de la información, mediante autenticación de usuarios y control de acceso a los datos.

#### **3.4.1.2. Requerimientos No Funcionales**

Rendimiento

* **RNF-01:** El sistema debe responder a las consultas de predicción y visualización de datos en un corto tiempo, asegurando una experiencia fluida para el usuario. Las respuestas deben generarse de forma casi inmediata, optimizando el uso de recursos y minimizando la latencia.
* **RNF-02:** El sistema debe soportar la conexión y operación simultánea de más de dos usuarios, garantizando un rendimiento estable y sin interrupciones durante el uso concurrente.
* **RNF-03:** El procesamiento y carga de archivos en formato CSV de hasta 100 MB debe completarse en un tiempo máximo de 2 minutos, incluyendo las etapas de validación, limpieza y almacenamiento de datos.

Seguridad

* **RNF-04:** El sistema debe implementar un mecanismo de autenticación segura, utilizando contraseñas encriptadas y almacenadas de forma protegida para evitar accesos no autorizados.
* **RNF-05:** El sistema debe cumplir con las normativas de protección de datos de salud, garantizando la confidencialidad, integridad y disponibilidad de la información sensible de los usuarios y registros médicos.
* **RNF-06:** El sistema debe registrar una bitácora completa de accesos y actividades críticas, incluyendo inicio de sesión, cargas de datos, generación de reportes y configuraciones administrativas.

Usabilidad

* **RNF-07:** La interfaz de usuario debe ser clara, intuitiva y fácil de usar, permitiendo la navegación fluida tanto para usuarios técnicos (analistas, administradores) como no técnicos (personal operativo o médico).
* **RNF-08:** El sistema debe incluir ayuda contextual y un manual de usuario accesible desde la interfaz, proporcionando orientación sobre el uso de las principales funcionalidades y procedimientos.

Fiabilidad

* **RNF-09:** Los modelos predictivos implementados deben alcanzar una precisión mínima del 85% en las pruebas realizadas con datos históricos, garantizando resultados confiables para la toma de decisiones.

Compatibilidad

* **RNF-10:** El sistema debe ser compatible con los principales navegadores web modernos, incluyendo Google Chrome, Mozilla Firefox y Microsoft Edge en sus versiones actuales, sin pérdida de funcionalidad o diseño.
* **RNF-11:** El backend debe ser desarrollado utilizando Python con el framework Flask, integrándose con herramientas web modernas y una base de datos MySQL versión 8.0 o superior, asegurando estabilidad, escalabilidad y mantenimiento eficiente.

**3.4.1.3 Historias de usuario**

**Épica 1 :** Gestión de Datos Epidemiológicos

|  |  |
| --- | --- |
| **Historia De Usuario** | |
| **Número:** 1 | **Usuario:** Administrador del sistema |
| **Nombre de la historia:** Cargar archivos CSV con datos históricos de enfermedades virales. | |
| **Prioridad en el negocio:** Alta | **Riesgo en desarrollo:** Medio |
| **Puntos estimados:** 5 | **Iteración asignada:** 1 |
| Programador responsable: | |
| **Descripción:** Como administrador del sistema, quiero cargar archivos CSV con datos históricos de enfermedades virales para que el sistema pueda procesarlos y almacenarlos correctamente. | |
| Validación:   * El sistema acepta archivos CSV con columnas: fecha, enfermedad, casos confirmados y región. * Se valida el formato del archivo y se muestran errores específicos si hay inconsistencias. * La carga exitosa muestra un resumen con el número total de registros importados. | |

|  |  |
| --- | --- |
| **Historia De Usuario** | |
| **Número:** 2 | **Usuario:** Administrador del sistema |
| **Nombre de la historia:** Limpieza y normalización automática de datos cargados. | |
| **Prioridad en el negocio:** Alta | **Riesgo en desarrollo:** Alto |
| **Puntos estimados:** 5 | **Iteración asignada:** 1 |
| Programador responsable: | |
| **Descripción:** Como administrador del sistema, quiero que los datos cargados sean limpiados y normalizados automáticamente para garantizar la calidad de la información. | |
| Validación:   * El sistema elimina registros duplicados correctamente. * Se completan los valores faltantes utilizando promedio móvil. * Se estandarizan formatos de fechas y nombres de regiones. | |

### 

**Épica:** Modelado Predictivo

|  |  |
| --- | --- |
| **Historia De Usuario** | |
| **Número:** 3 | **Usuario:** Administrador del sistema |
| **Nombre de la historia:** Entrenamiento de modelos de predicción lineal y múltiple. | |
| **Prioridad en el negocio:** Alta | **Riesgo en desarrollo:** Alto |
| **Puntos estimados:** 8 | **Iteración asignada:** 2 |
| Programador responsable: | |
| **Descripción:** Como administrador del sistema, quiero entrenar modelos de predicción lineal y múltiple utilizando los datos históricos para generar proyecciones de contagios. | |
| Validación:   * El sistema permite seleccionar variables independientes. * Se muestran métricas de precisión cómo R² y RMSE. * Se generan gráficos comparativos entre valores reales y predichos. | |

|  |  |
| --- | --- |
| **Historia De Usuario** | |
| **Número:** 4 | **Usuario:** Institución de salud |
| **Nombre de la historia:** Ejecución de predicciones a corto plazo. | |
| **Prioridad en el negocio:** Alta | **Riesgo en desarrollo:** Medio |
| **Puntos estimados:** 6 | **Iteración asignada:** 2 |
| Programador responsable: | |
| **Descripción:** Como institución de salud, quiero ejecutar predicciones para un periodo de tiempo próximo (en semanas) con el fin de planificar recursos sanitarios. | |
| Validación:   * El sistema permite seleccionar enfermedad y región. * Las predicciones se muestran en tabla y gráfico temporal. * Los resultados se pueden exportar en PDF y Excel. | |

### 

**Épica:** Visualización e Informes

|  |  |
| --- | --- |
| **Historia De Usuario** | |
| **Número:** 5 | **Usuario:** Usuario final |
| **Nombre de la historia:** Visualización de tendencias temporales de enfermedades. | |
| **Prioridad en el negocio:** Media | **Riesgo en desarrollo:** Medio |
| **Puntos estimados:** 4 | **Iteración asignada:** 3 |
| Programador responsable: | |
| **Descripción:** Como usuario final, quiero visualizar en un dashboard las tendencias temporales de las enfermedades para monitorear el comportamiento viral. | |
| Validación:   * Se muestran gráficos de línea con las tendencias históricas. * Se incluyen proyecciones futuras diferenciadas por color. * Se permite filtrar por fecha y tipo de enfermedad. | |

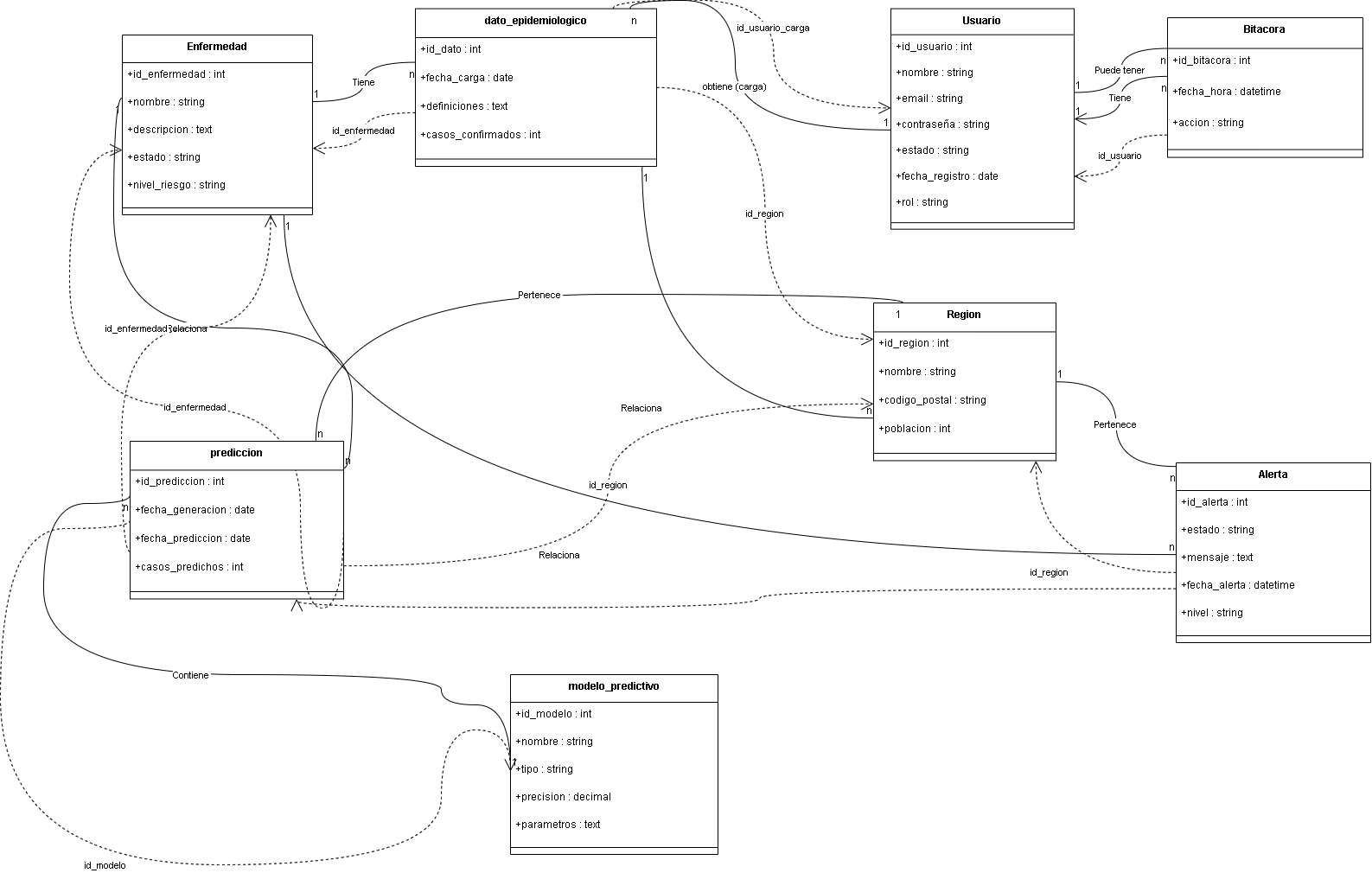
|  |  |
| --- | --- |
| **Historia De Usuario** | |
| **Número:** 6 | **Usuario:** Institución de salud |
| **Nombre de la historia:** Generación automática de reportes de alertas tempranas. | |
| **Prioridad en el negocio:** Alta | **Riesgo en desarrollo:** Medio |
| **Puntos estimados:** 5 | **Iteración asignada:** 3 |
| Programador responsable: | |
| **Descripción:** Como institución de salud, quiero generar reportes automatizados de alertas tempranas para poder tomar decisiones oportunas. | |
| Validación:   * El sistema genera un semáforo de alertas (bajo, medio, alto). * Se envían notificaciones por correo cuando cambia el tipo de alerta. * Los reportes incluyen recomendaciones específicas según el nivel de riesgo. | |

**Épica:** Administración del Sistema

|  |  |
| --- | --- |
| **Historia De Usuario** | |
| **Número:** 7 | **Usuario:** Administrador del sistema |
| **Nombre de la historia:** Gestión de usuarios y permisos de acceso. | |
| **Prioridad en el negocio:** Alta | **Riesgo en desarrollo:** Alto |
| **Puntos estimados:** 6 | **Iteración asignada:** 4 |
| Programador responsable: | |
| **Descripción:** Como administrador, quiero gestionar usuarios y permisos de acceso para garantizar la seguridad de la información del sistema. | |
| Validación:   * El sistema permite crear, editar y desactivar usuarios. * Se pueden asignar roles (administrador, epidemiólogo, consultor). * Se registra una bitácora de accesos y acciones del sistema. | |

### 

**3.4.2 Etapa 2 de XP. Diseño:**



**Explicación del diagrama UML**

Un usuario ingresa datos epidemiológicos de una enfermedad en una región.

Con estos datos, se entrena un modelo predictivo.

El modelo predictivo genera predicciones para combinaciones de enfermedad y región.

Si una predicción supera un límite, se emite una alerta para la región y enfermedad correspondientes.

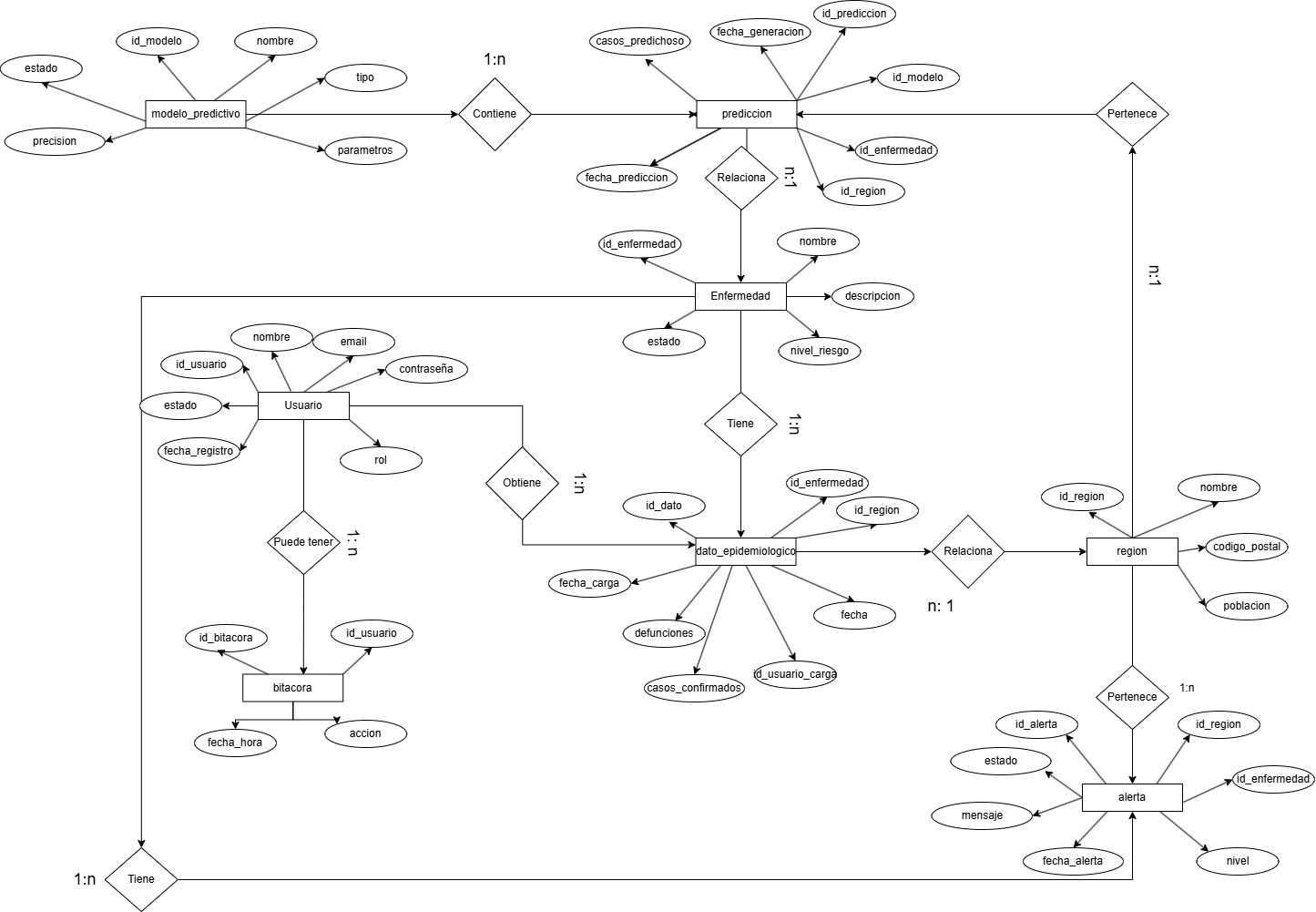
Todas las acciones del Usuario se registran en la Bitácora.

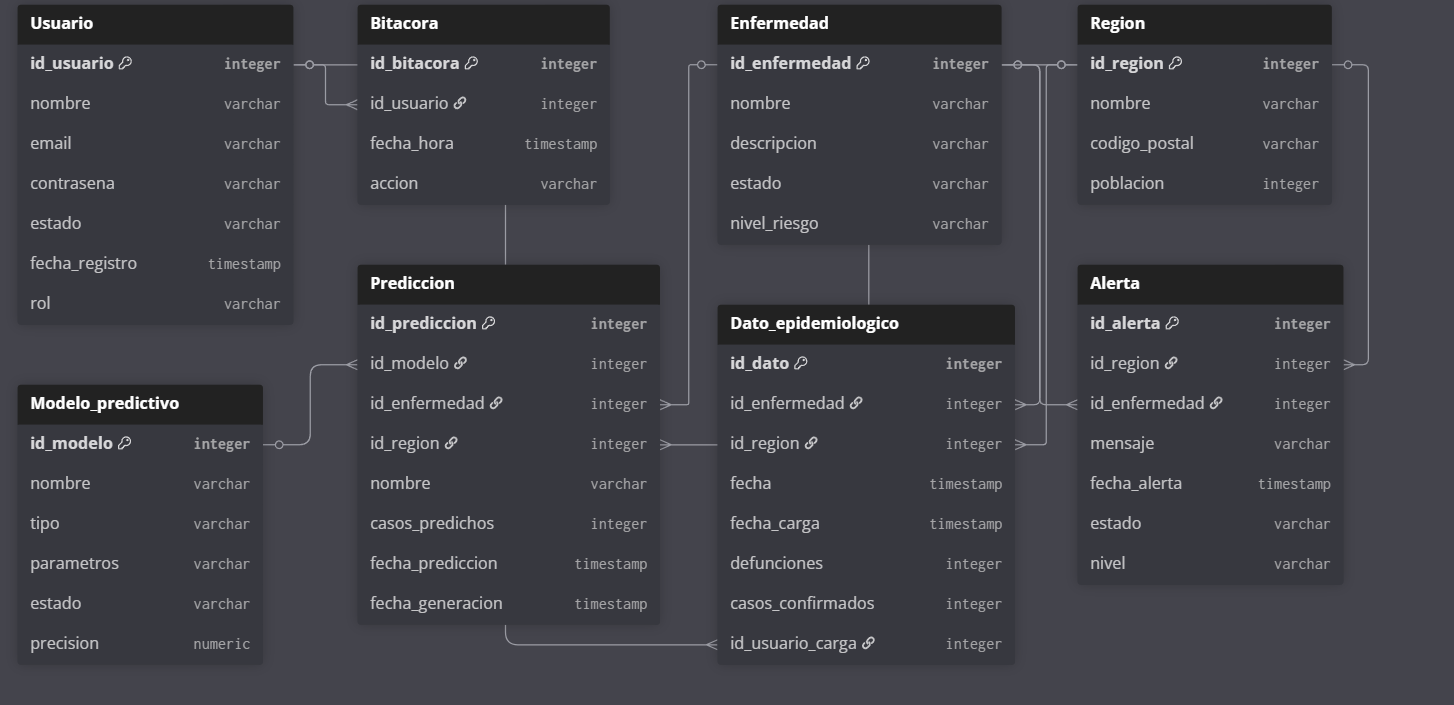
Para este Sistema de Predicción de Enfermedades, el diagrama de clases es importante porque:

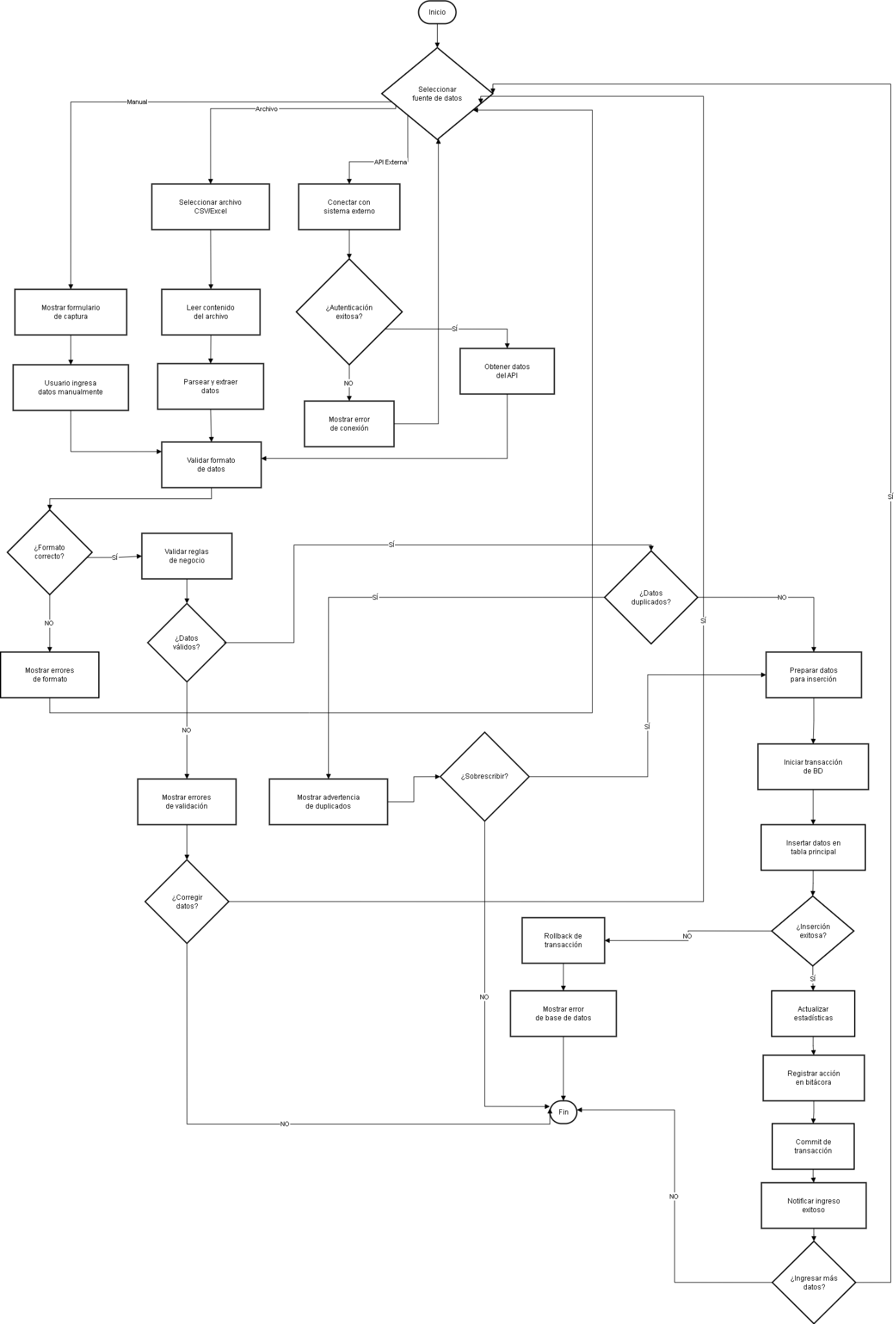
1. Maneja información sensible (datos de salud pública).
2. Integra múltiples fuentes (usuarios, sistemas externos, modelos de ML).
3. Genera información crítica (predicciones y alertas que afectan decisiones de salud).
4. Requiere trazabilidad (bitácora de quién hizo qué y cuándo).
5. Tiene relaciones complejas (modelos, predicciones, regiones, enfermedades, alertas).

Sin este diagrama, sería muy fácil:

* Perder datos importantes
* Crear predicciones sin trazabilidad
* No poder vincular alertas con sus fuentes.
* Tener inconsistencias entre regiones y datos.

Entidad-Relación



**Diagrama de flujo:** 

**Explicación diagrama de flujo**

### La lógica de flujo general abarca puntos muy importantes:

Datos malos = Predicciones malas = Alertas incorrectas = Decisiones equivocadas de salud pública.

También abarca la parte de los datos; ya que pueden provenir de diferentes fuentes:

* Manual: Epidemiólogos ingresando casos individuales.
* Archivo: Importación masiva de datos históricos.
* API externa: Integración con otros sistemas de salud.

Cada fuente tiene sus propios desafíos, formatos y procesos de validación.

Para que los datos de salud pública sean asumidos de buena manera. Se deben validar aspectos clave como:

* ¿Los números de casos son positivos y coherentes?
* ¿Las fechas corresponden al rango permitido?
* ¿La región o localidad existe en el sistema?
* ¿La enfermedad está dentro del catálogo oficial?
* ¿No hay duplicados en los registros cargados?

Estas validaciones aseguran integridad, precisión y confianza sobre el sistema.

### Puntos fuertes del diseño

1. Tres fuentes de datos → Flexibilidad del sistema.
2. Validación en capas → Mayor calidad de la información.
3. Manejo de duplicados → Integridad garantizada.
4. Transacciones de base de datos → Consistencia total.
5. Opciones de recuperación → Experiencia de usuario amigable.

**Referencias:**

Pressman, R. S., & Maxim, B. R. (2020). Ingeniería del software: Un enfoque práctico (9a ed.). McGraw-Hill Education.

Beck, K. (2004). *Extreme Programming Explained: Embrace Change* (2.ª ed.). Addison-Wesley.

Project Management Institute. (2017). Una guía a los fundamentos para la dirección de proyectos (Guía del PMBOK) (6a ed.). Project Management Institute, Inc.

Comisión Europea Centro de Excelencia en PM2 (CoEPM2). (2021). Metodología de Gestión de Proyectos PM2: Guía 3.0.1. Comisión Europea, DIGIT Centre of Excellence in PM2. 2.Guide\_.v3.0.Espagnol.Spanish.Translation.ES\_.v3.0.1.pdf"><https://www.pm2alliance.eu/wp-content/uploads/2021/08/Guia.Metodologia.PM2.Guide_.v3.0.Espagnol.Spanish.Translation.ES_.v3.0.1.pdf>

ISO. (2015). ISO 9001:2015 Sistemas de gestión de la calidad - Requisitos. International Organization for Standardization.

AXELOS. (2017). *Managing Successful Projects with PRINCE2* (6th ed.). TSO (The Stationery Office).

Harry, M. J., & Schroeder, R. (2000). Six Sigma: The Breakthrough Management Strategy Revolutionizing the World's Top Corporations.

Ladas, C. (2008). Scrumban: Essays on Kanban Systems for Lean Software Development. Modus Cooperand Press.

Project Management Institute. (2017). *A Guide to the Project Management Body of Knowledge (PMBOK® Guide)* (6th ed.). Project Management Institute.

Poppendieck, M., & Poppendieck, T. (2003). *Lean software development: An agile toolkit*. Addison-Wesley.

EALDE Business School. (2014). *Guía para Project Managers: Claves del PMBOK Guide 7 para la dirección de proyectos*

*Rodríguez, A., & Yepes, V. (2010). Análisis del método de la cadena crítica vs método del camino crítico. Viabilidad y conceptos.*

*Goldratt, E. M. (1997). Critical Chain. North River Press.*

*Leach, L. P. (2000). Critical Chain Project Management. Artech House.*

*Royce, W. (1970). Managing the Development of Large Software Systems. Proceedings of IEEE WESCON.*

.

Norma Mexicana (Fuente Oficial y Actual): Normalización y Certificación (NYCE), S.C. (2016). *NMX-I-059/02-NYCE-2016. Tecnologías de la Información—Software—Modelos de Procesos y Evaluación para Desarrollo y Mantenimiento de Software*.<https://www.nyce.org.mx/nmx-i-059-02-nyce-2016-moprosoft/>

Documento Base del Modelo (Definición y Autores Originales): Oktaba, H., Alquicira Esquivel, C., Su Ramos, A., Martínez Martínez, A., Quintanilla Osorio, G., Ruvalcaba López, M., López Lira Hinojo, F., Rivera López, M. E., Orozco Mendoza, M. J., Fernández Ordóñez, Y., & Flores Lemus, M. Á. (2005). *Modelo de Procesos para la Industria de Software: MoProSoft* (Versión 1.3). Secretaría de Economía de México.<https://www.researchgate.net/publication/267028000_Modelo_de_Procesos_para_la_Industria_de_Software_MoProSoft>

Codd, E. F. (1970). A relational model of data for large shared data banks. *Communications of the ACM*, *13*(6), 377–387.<https://doi.org/10.1145/362384.362685>

Chen, P. P.-S. (1976). The entity-relationship model—Toward a unified view of data. *ACM Transactions on Database Systems (TODS)*, *1*(1), 9–36.<https://doi.org/10.1145/320434.320440>

Kerzner, H. (2017). *Project management: A systems approach to planning, scheduling, and controlling* (12.ª ed.). John Wiley & Sons.

Project Management Institute. (2017). *A guide to the Project Management Body of Knowledge (PMBOK® Guide)* (6.ª ed.). Project Management Institute.

Schwalbe, K. (2019). *Information technology project management* (9.ª ed.). Cengage Learning.

Sommerville, I. (2016). *Software engineering* (10.ª ed.). Pearson.

Belle, A., Thiagarajan, R., Soroushmehr, S. M. R., Navidi, F., Beard, D. A., & Najarian, K. (2015). Big data analytics in healthcare. *BioMed Research International, 2015*, 370194. <https://doi.org/10.1155/2015/370194>

Fielding, R. T. (2000). *Architectural styles and the design of network-based software architectures* (Tesis doctoral). University of California, Irvine.

Flanagan, D. (2020). *JavaScript: The definitive guide* (7th ed.). O'Reilly Media.

Mitchell, T. M. (1997). *Machine learning*. McGraw-Hill.

Porta, M. (Ed.). (2014). *A dictionary of epidemiology* (6th ed.). Oxford University Press.

Sánchez-Vizcaíno, J. M. (2019). *Enfermedades virales de los animales*. Ediciones Díaz de Santos.

Tufte, E. R. (2001). *The visual display of quantitative information* (2nd ed.). Graphics Press.

*¿Qué es la minería de datos? La minería de datos, explicada - AWS*. (s. f.). Amazon Web Services, Inc. <https://aws.amazon.com/es/what-is/data-mining>

World Health Organization. (2021). *Global surveillance of COVID-19: A guide to principles and practice for development and implementation.* WHO. <https://www.who.int/health-topics/coronavirus#tab=tab_1>

Mozilla Developer Network (MDN). (2025). Introducción a los frameworks web. Recuperado de <https://developer.mozilla.org/es/docs/Learn/Tools_and_testing/Client-side_JavaScript_frameworks>

React. (2025). React – A JavaScript library for building user interfaces. Meta Platforms, Inc. Recuperado de <https://react.dev/>

Angular. (2025). Angular – The modern web developer’s platform. Google LLC. Recuperado de <https://angular.io/>

Vue.js. (2025). The Progressive JavaScript Framework. Evan You y colaboradores. Recuperado de <https://vuejs.org/>

Django Software Foundation. (2025). Django Documentation. Recuperado de <https://docs.djangoproject.com/>

Laravel. (2025). Laravel – The PHP Framework for Web Artisans. Taylor Otwell. Recuperado de <https://laravel.com/>

W3Schools. (2025). CSS Flexbox. Recuperado de <https://www.w3schools.com/css/css3_flexbox.asp>

MDN Web Docs. (2025). CSS Positioning. Recuperado de <https://developer.mozilla.org/en-US/docs/Web/CSS/position>