

**TALLER INTEGRACIÓN EN
ANÁLISIS DE DATOS**

Semana 5
Diseño de la propuesta

ÍNDICE

INTRODUCCIÓN	3
RESUMEN	4
PALABRAS CLAVE	4
PREGUNTAS GATILLANTES	4
1.DISEÑO DE LA PROPUESTA DE MEJORA	5
1.1DEFINICIÓN DE RECURSOS	5
1.1.1 DEFINICIÓN DE SOFTWARE	8
1.2 DESARROLLO DE LA PROPUESTA DE SOLUCIÓN(DASHBOARD)	10
1.3 IMPLEMENTACIÓN DE LA PROPUESTA DE MEJORA	11
COMENTARIO FINAL	21
REFERENCIAS	22

INTRODUCCIÓN

Una vez definidos los aspectos generales del proyecto de investigación, incluyendo sus fundamentos teóricos y metodológicos, es esencial abordar los aspectos técnicos que darán forma a los resultados del estudio. Para desarrollar una propuesta de mejora efectiva basada en el análisis de datos, es fundamental comprender a fondo las necesidades y objetivos del proyecto. Esta fase se centra en el diseño del dashboard analítico como solución clave para el problema identificado, estableciendo directrices claras y los recursos necesarios para su implementación.

Este contenido se enfoca en el diseño detallado de la propuesta de mejora, estableciendo las bases para integrar todos los elementos de la investigación: contexto, objetivos y metodología. Un enfoque integral garantiza que el dashboard no solo facilite la toma de decisiones basada en datos, sino que también impulse nuevas soluciones en el campo del análisis de datos, promoviendo la innovación y la mejora continua.

El desarrollo del dashboard debe ir más allá de la simple visualización de datos, permitiendo generar información valiosa que optimice los procesos de gestión y fomente la adaptación a tendencias emergentes en la analítica de datos y la inteligencia de negocios.

RESUMEN

Dentro de las especificaciones técnicas de la propuesta, es esencial definir los recursos de hardware y software requeridos, así como los requisitos funcionales y no funcionales del dashboard. También es importante detallar el proceso de desarrollo de la propuesta de mejora, destacando especialmente el diseño del sistema. En esta etapa, se modelan y crean los diversos componentes que conforman el dashboard, incluyendo visualizaciones, modelos de datos, bases de datos y los requisitos mínimos de operación, todos los cuales son fundamentales para alcanzar los resultados deseados.

El diseño del dashboard debe orientarse a lograr objetivos específicos que permitan al investigador resolver el problema planteado y aportar al conocimiento en el área del análisis de datos. Estos objetivos deben estar alineados con los resultados esperados al implementar la propuesta, asegurando que el dashboard proporcione información útil y procesable para la toma de decisiones.

Además, la elección de la metodología de desarrollo afecta directamente el diseño del dashboard. Se pueden emplear metodologías ágiles, que permiten iteraciones rápidas y flexibles, o metodologías tradicionales como el modelo en cascada, dependiendo de las características del proyecto y del problema a resolver. La metodología elegida determinará la forma en que se estructuran los procesos de diseño, implementación y mejora continua del dashboard.

PALABRAS CLAVE

- Diseño del dashboard
- Desarrollo del dashboard
- Integración de la propuesta
- Recursos de hardware y software
- Elementos o Artefactos a entregar

PREGUNTAS GATILLANTES

1. ¿Qué implica el diseño de un dashboard analítico en un proyecto de investigación?
2. ¿Cómo se deben integrar los diferentes elementos del dashboard para garantizar una implementación efectiva de la propuesta de mejora?
3. ¿Qué requisitos son necesarios para llevar a cabo la implementación del dashboard en la organización que lo solicita?

1.DISEÑO DE LA PROPUESTA DE MEJORA

El diseño de la propuesta de mejora establece la base para orientar todas las etapas del desarrollo del dashboard. En esta fase, se seleccionan los métodos, enfoques técnicos y herramientas que permitirán gestionar eficazmente el tiempo y los recursos. Además, se identifican y analizan los requisitos técnicos y funcionales del dashboard, asegurando que la solución no solo cumpla con las necesidades actuales, sino que también tenga la capacidad de escalar y adaptarse a futuros requerimientos.

Para IACC (2020) una vez establecidos los objetivos del proyecto y definidas claramente las especificaciones del dashboard, es crucial planificar su desarrollo. Esto implica garantizar que el diseño se alinee con los requerimientos de los usuarios, las restricciones operativas y las mejores prácticas en el análisis de datos. Para lograrlo, es fundamental adoptar un enfoque estructurado que asegure la correcta integración de fuentes de datos, la optimización de la interfaz de usuario y la implementación de visualizaciones interactivas que faciliten la toma de decisiones.

En esta etapa, se definen los artefactos relevantes que conformarán la solución, como modelos de datos, esquemas de bases de datos, visualizaciones de indicadores y reportes automatizados.

Ejemplo

Propuesta: Desarrollo de un dashboard analítico que optimice la gestión de cobranzas en finrec solutions.

En un contexto de cobranzas, un dashboard diseñado para optimizar la gestión financiera incluiría visualizaciones de indicadores clave de rendimiento (KPIs), reportes de pagos y segmentación de clientes según el riesgo de morosidad. Además, se establecerían los requisitos mínimos de operación y se planificaría el desarrollo de funcionalidades esenciales, como la integración con sistemas de facturación, la predicción de pagos y la segmentación de clientes.

El diseño del dashboard debe enfocarse en proporcionar información útil y procesable, comprobando que los usuarios puedan monitorear métricas relevantes, detectar patrones y tomar decisiones estratégicas basadas en datos en tiempo real. Finalmente, la elección de una metodología de desarrollo adecuada, ya sea ágil o tradicional, influirá directamente en la forma en que se estructuran los procesos de implementación y mejora continua del sistema.

1.1DEFINICIÓN DE RECURSOS

Al evaluar las diversas opciones para el diseño e implementación de un dashboard, es necesario llevar a cabo un análisis exhaustivo de cada alternativa, considerando tanto los componentes de hardware y software esenciales, como los sistemas operativos, gestores de bases de datos y aplicaciones disponibles en el mercado. Dado que no existe una solución técnica única que garantice el éxito, cada opción debe ser examinada cuidadosamente.

Según IACC (2021), la especificación de las alternativas debe incluir una descripción concisa de cada opción, una lista de los requisitos que debe cumplir y una justificación clara para su aceptación o rechazo como solución viable.

Se recomienda un proceso estructurado para evaluar las diferentes opciones, que comprenda las siguientes etapas, para IACC (2022):

1. Identificación de alternativas relevantes.
2. Preselección de las opciones más prometedoras.
3. Definición de las características clave a evaluar en cada opción.
4. Evaluación técnica detallada de cada opción, considerando su rendimiento, funcionalidad y capacidad de integración.
5. Evaluación de los proveedores de software y hardware, incluyendo su reputación, soporte técnico y costos.
6. Selección final de la alternativa óptima que mejor se adapte a las necesidades y requisitos del proyecto de dashboard.

Ejemplo

Propuesta: Desarrollo de un dashboard analítico para la gestión de cobranzas en finrec solutions.

Para evaluar las herramientas y tecnologías necesarias para el desarrollo del dashboard, se deben seguir las siguientes directrices:

a) Identificar las ventajas y características que debe cumplir cada componente. Esto incluye tanto el software (lenguaje de programación, herramientas de visualización, base de datos) como el hardware (servidores, capacidad de procesamiento).

b) Definir un rango de valores para calificar cada característica. Por ejemplo:

- Características: facilidad de uso, rendimiento, escalabilidad, seguridad, costo.
- Calificaciones: bajo: 0, Medio: 1, Alto: 2.

c) Asignar un peso a cada característica según su importancia relativa. Las ponderaciones pueden ser establecidas en rangos como 1 a 3 (normal: 1, importante: 2, muy importante: 3) o en rangos mayores, como 1 a 5 o 1 a 10.

Al definir la importancia de las características, es probable que se identifiquen requisitos esenciales, es decir, aquellos cuyo incumplimiento hará que la opción sea descartada de inmediato.

Se puede utilizar una tabla comparativa para evaluar las características de software y hardware, que permita analizar las ventajas y desventajas de cada elemento (hardware, software y aplicación) considerado en la selección.

Basándose en las directrices proporcionadas, se puede estructurar un cuadro comparativo de la siguiente manera, como por ejemplo:

Cuadro comparativo de alternativas							
Nro.	Características	Característica imprescindible	Rango		Ponderación	Máxima calificación	Calificación obtenida
			Mín.	Máx.			

Tabla 1. Cuadro evaluativo de alternativas (opciones) a implementar
Fuente: elaboración propia a partir de IACC (2025)

Ejemplo: Evaluación de la herramienta de visualización Tableau para el dashboard de finrec solutions.

Cuadro comparativo de alternativas							
Nro.	Características	Característica imprescindible	Rango		Ponderación	Máxima calificación	Calificación obtenida
			Mín.	Máx.			
1	Facilidad de uso	Sí	1	3	3	9	8
2	Rendimiento	Sí	1	3	2	6	5
3	Escalabilidad	Sí	1	3	2	6	6
4	Seguridad	Sí	1	3	3	9	7
5	Costo	No	1	3	1	3	2
6	Integración con Base de Datos	Sí	1	3	3	9	9

Tabla 2. Cuadro evaluativo de la herramienta Tableau
Fuente: elaboración propia a partir de IACC (2025)

En la Tabla 2 se presenta una evaluación sistemática de las características clave de Tableau, herramienta de visualización de datos considerada para el desarrollo del dashboard de FinRec Solutions.

Las características evaluadas en la tabla abarcan aspectos críticos para el éxito del dashboard, incluyendo:

- **Facilidad de uso:** es fundamental para garantizar que los analistas y usuarios de FinRec Solutions puedan interactuar con el dashboard sin dificultades. La calificación obtenida es 8 de 9, lo que indica que Tableau es fácil de usar, aunque podría haber algunas áreas que necesiten ajustes o capacitación adicional para usuarios menos técnicos.
- **Rendimiento:** es importante para asegurar que el dashboard pueda manejar grandes volúmenes de datos de cobranzas y generar visualizaciones rápidamente. La calificación obtenida es 5 de 9, mostrando que Tableau ofrece un rendimiento adecuado para la mayoría de los casos, pero podría haber limitaciones en la velocidad y la eficiencia al procesar conjuntos de datos extremadamente grandes o generar informes muy complejos. Se recomienda realizar pruebas exhaustivas con los volúmenes de datos reales de la empresa para identificar posibles cuellos de botella y optimizar el rendimiento.
- **Escalabilidad:** es esencial para asegurar que el dashboard pueda crecer y adaptarse a las necesidades futuras de FinRec Solutions, a medida que aumente la cantidad de datos y usuarios. La calificación obtenida es 6 de 9, indicando que Tableau es escalable, pero se deben considerar cuidadosamente las limitaciones de la licencia y la infraestructura para garantizar un rendimiento óptimo a largo plazo.
- **Integración con BD:** es crucial para asegurar que el dashboard pueda conectarse y extraer datos de las bases de datos existentes en FinRec Solutions. La calificación obtenida es 9 de 9, lo que demuestra que Tableau se integra perfectamente con una amplia variedad de bases de datos, incluyendo las que utiliza FinRec Solutions.
- **Costo:** aunque no es un requisito, el costo de la licencia y el mantenimiento de la herramienta de visualización es un factor importante. La calificación obtenida es 5 de 9, sugiriendo que Tableau tiene un costo relativamente alto, pero se debe evaluar en el contexto del presupuesto disponible y el valor que aporta al análisis de datos y la toma de decisiones en FinRec Solutions.
- **Seguridad:** es un aspecto crítico para proteger la confidencialidad de los datos de cobranzas de FinRec Solutions. La calificación obtenida es 7 de 9. Tableau ofrece diversas características de seguridad, como el cifrado de datos, la autenticación de usuarios y los permisos de acceso granular. Sin embargo, es importante configurar y administrar correctamente estas características para garantizar la protección adecuada de la información sensible. Se recomienda realizar una auditoría de seguridad exhaustiva y establecer políticas claras para el acceso y uso de los datos en el dashboard.

Justificación

Tableau cumple adecuadamente con los requisitos de facilidad de uso, rendimiento e integración con bases de datos, siendo una opción sólida para el desarrollo del dashboard analítico de FinRec Solutions. La escalabilidad es un aspecto que requiere una planificación cuidadosa, y el costo debe ser evaluado en el contexto del presupuesto disponible y el retorno de inversión esperado.

En general, la herramienta se presenta como una herramienta valiosa para mejorar la gestión de cobranzas en FinRec Solutions, siempre y cuando se consideren los aspectos mencionados y se implementen las estrategias adecuadas para optimizar su rendimiento y escalabilidad.



Recuerda que

En la tabla evaluativa de Tableau, se utilizaron los siguientes parámetros para evaluar cada característica:

- **Máxima calificación:** es el valor máximo que puede alcanzar una característica en la evaluación. Se calcula según un rango y ponderación predefinidos. En este caso, se utilizó un rango de 1 a 3 y una ponderación que varía según la importancia de la característica (de 1 a 3). Por lo tanto, la máxima calificación posible para una característica con ponderación 3 es 9 (3×3) y para una característica con ponderación 2 es 6 (3×2).
- **Calificación obtenida:** es el puntaje específico que recibe Tableau en cada característica, basado en cómo cumple con los criterios de evaluación. Esta calificación se encuentra dentro del rango de 1 a 3 y se multiplica por la ponderación correspondiente para obtener la calificación final.

Tableau es una herramienta de visualización intuitiva (calificación: 8/9), pero algunos usuarios podrían requerir capacitación. Su rendimiento es adecuado (5/9), aunque podría tener limitaciones con grandes volúmenes de datos. Es escalable (6/9) y se integra perfectamente con diversas bases de datos (9/9). Ofrece seguridad robusta (7/9) con la configuración adecuada. Su costo es relativamente alto (2/9).

1.1.1 DEFINICIÓN DE SOFTWARE

El **software para un dashboard** abarca diversas categorías, cada una con un papel definido en el procesamiento, visualización y gestión de los datos.

1. **Sistema Operativo:** es la base sobre la que se ejecuta todo el software. Debe ser estable, seguro y compatible con las demás herramientas. Ejemplos comunes incluyen Windows Server, Linux (varias distribuciones) o macOS Server.
2. **Gestor de Base de Datos (DBMS):** almacena y gestiona los datos que alimentan el dashboard. Debe ser capaz de manejar grandes volúmenes de datos, realizar consultas complejas y garantizar la integridad de los datos. Opciones populares incluyen MySQL, PostgreSQL, Microsoft SQL Server, Oracle y soluciones en la nube como Amazon Redshift o Google BigQuery.
3. **Herramientas de Extracción, Transformación y Carga (ETL):** facilitan la integración de datos de diversas fuentes, su limpieza y transformación a un formato adecuado para el análisis, y su carga en el DBMS. Ejemplos incluyen informática PowerCenter, Talend, Apache NiFi y herramientas de código abierto como Pentaho Kettle.

4. **Herramientas de visualización de datos:** permiten crear representaciones gráficas interactivas y fáciles de entender de los datos. Deben ofrecer una amplia gama de gráficos, la capacidad de personalizar las visualizaciones y la posibilidad de crear dashboards interactivos. Algunas opciones populares son Tableau, Power BI, Qlik Sense, Looker y bibliotecas de visualización de datos en lenguajes de programación como D3.js (JavaScript) o Matplotlib y Seaborn (Python).
5. **Lenguajes de programación (opcional):** pueden ser necesarios para tareas de procesamiento de datos más complejas, análisis estadístico avanzado o personalización de visualizaciones. Python y R son lenguajes ampliamente utilizados en el análisis de datos.
6. **Software de servidor web (opcional):** si el dashboard se va a acceder a través de la web, se requerirá un servidor web (como Apache o Nginx) para alojar la aplicación y gestionar las solicitudes de los usuarios.

Los **requisitos de hardware** para un dashboard dependen del volumen de datos, la complejidad de los análisis, el número de usuarios y las necesidades de rendimiento.

1. **Servidores:** son el corazón del proyecto. Deben tener suficiente capacidad de procesamiento (CPU), memoria RAM y almacenamiento en disco para manejar los datos y las consultas. Se recomienda utilizar servidores de alto rendimiento con capacidad de escalabilidad para adaptarse al crecimiento futuro.
2. **Almacenamiento:** el almacenamiento debe ser lo suficientemente grande para contener los datos, las copias de seguridad y otros archivos del sistema. Se pueden utilizar discos duros locales, almacenamiento en red (NAS o SAN) o soluciones en la nube.
3. **Red:** la red debe ser rápida y confiable para permitir una comunicación fluida entre los servidores, la base de datos y los usuarios. Se recomienda utilizar una red de alta velocidad con suficiente ancho de banda.
4. **Equipos de Usuario:** los usuarios necesitarán equipos (computadoras de escritorio, laptops, tablets) con capacidad suficiente para acceder al dashboard y visualizar los datos. Los requisitos de hardware para los equipos de usuario dependerán de la complejidad del dashboard y las funcionalidades que ofrezca.



Ejemplo

Propuesta: Desarrollo de un dashboard analítico que optimice la gestión de cobranzas en Finrec solutions.

Para la implementación del dashboard se requiere, por un lado, una selección adecuada de **software** que garantice un rendimiento óptimo, compatibilidad con los sistemas existentes y escalabilidad futura. A continuación, se presentan las opciones recomendadas:

- Sistema Operativo: Windows Server 2022 o superior.
- Gestor de Base de Datos (DBMS): Microsoft SQL Server 2019 o superior.
- Herramienta de Visualización de Datos: Tableau Desktop y Tableau Server.
- Software de ETL (Extracción, Transformación y Carga): Talend Studio (versión Open Source).
- Lenguaje de Programación (Opcional): Python, con bibliotecas como Pandas y NumPy para procesamiento de datos.
- Software de Servidor Web: Internet Information Services (IIS).

Por otra lado, el **hardware** debe garantizar un procesamiento eficiente de grandes volúmenes de datos y un desempeño fluido en la generación de reportes analíticos. A continuación, se describen los recursos recomendados:

1. Servidor para Tableau Server:

- Procesador: Intel Xeon Silver 4314 o equivalente.
- Memoria RAM: 64 GB o superior.
- Almacenamiento: 1 TB SSD, con opción de almacenamiento en red (NAS o SAN).

2. Servidor para la Base de Datos (SQL Server):

- Se utilizará la infraestructura existente de FinRec Solutions, garantizando compatibilidad y rendimiento óptimo.

3. Equipos de Usuario (Analistas y Usuarios del Dashboard):

- Procesador: Intel Core i5 o equivalente.
- Memoria RAM: 16 GB o superior.
- Almacenamiento: 256 GB SSD o superior.

1.1.3 Desarrollo propuesta de solución (dashboard).

En la fase de desarrollo, el programador convertirá el diseño conceptual del dashboard en una herramienta funcional. Este diseño es una representación visual y estructural del dashboard que incluye la identificación de las variables a mostrar, la organización de la información, la selección de los tipos de gráficos y la

interacción con el usuario. Para ello, se elaboran wireframes y mockups que servirán como guía para la construcción del dashboard.

Desarrollo Ágil con Scrum

En un entorno de desarrollo ágil basado en Scrum, el equipo trabaja en ciclos cortos o sprints, entregando versiones funcionales e incrementales del dashboard. Cada sprint se enfoca en un conjunto de funcionalidades priorizadas, con el objetivo de aportar valor continuo al usuario.

El desarrollo del dashboard bajo esta metodología aplica principios del Diseño Centrado en el Usuario (DCU), lo que implica considerar las necesidades, expectativas y características de los usuarios finales en cada etapa del proceso. Para ello, se llevan a cabo:

- Investigación de usuarios.
- Creación de "personas" (perfiles representativos de los usuarios típicos).
- Pruebas de usabilidad.
- Incorporación de retroalimentación en cada iteración.

Responsabilidades del Desarrollador en la Fase de Desarrollo

- 1. Comprender los requisitos:** analizar los requerimientos del dashboard, incluyendo los datos a visualizar, funcionalidades necesarias y preferencias de los usuarios.
- 2. Seleccionar herramientas y tecnologías:** elegir las herramientas y tecnologías adecuadas según los requerimientos del proyecto, habilidades del estudiante y recursos disponibles.
- 3. Diseñar la arquitectura:** planificar la estructura del dashboard, definiendo la conexión con las fuentes de datos, procesamiento de información y organización de la interfaz de usuario.
- 4. Desarrollar el código:** implementar las funcionalidades necesarias, incluyendo la integración con las fuentes de datos, transformación de la información y generación de visualizaciones.
- 5. Pruebas y correcciones:** realizar pruebas exhaustivas para detectar y corregir errores, asegurando compatibilidad en distintos dispositivos y navegadores.
- 6. Documentar el código:** incluir comentarios claros y concisos para facilitar el mantenimiento y comprensión del código.

Implementación técnica

1. Herramientas y Lenguajes

El **stack tecnológico** dependerá de los requerimientos del proyecto. Algunas opciones comunes incluyen:

- Lenguajes de programación: Python (Dash, Flask), JavaScript (React, D3.js).
- Bases de datos: PostgreSQL, MongoDB.
- Frameworks de visualización: Power BI, Tableau, Plotly.

- Infraestructura y despliegue: Docker, AWS, Google Cloud.

Ejemplo:

A continuación, se presenta un fragmento de código en Python utilizando Dash para construir un dashboard básico de cobranzas:

```
import dash
from dash import dcc, html
import plotly.express as px
import pandas as pd

# Datos de ejemplo
data = pd.DataFrame({
    'Cliente': ['A', 'B', 'C', 'D'],
    'Monto Cobrado': [5000, 3000, 7000, 2000]
})

fig = px.bar(data, x='Cliente', y='Monto Cobrado', title='Cobranza por Cliente')

# Inicialización de la app
app = dash.Dash(__name__)
app.layout = html.Div([
    html.H1('Dashboard de Cobranzas'),
    dcc.Graph(figure=fig)
])

if __name__ == '__main__':
    app.run_server(debug=True)
```

Este código genera una visualización interactiva de los montos cobrados por cliente, permitiendo análisis dinámico en la gestión de cobranzas. No presenta el ejemplo filtros interactivos ni conexiones con base de datos.

Un dashboard de cobranzas típico podría incluir las siguientes visualizaciones:

- Gráfico de barras: Representa el monto total de cobranzas por mes, comparando el mes actual con los anteriores.
- Gráfico de líneas: Muestra la evolución de la cartera de cobranzas a lo largo del tiempo, incluyendo saldo pendiente, pagos recibidos y deudas vencidas.
- Tabla: Presenta el detalle de las cobranzas por cliente, con información como nombre, monto adeudado, fecha de vencimiento y estado de la deuda.

- Indicadores clave: Destacan métricas importantes como porcentaje de cobranza, promedio de días de pago y monto total de deuda vencida.



Importante

¿Qué es un stack tecnológico?

Un stack tecnológico es el conjunto de tecnologías utilizadas para desarrollar una solución digital, como un dashboard o una aplicación web. Incluye:

- Frontend: tecnologías para la interfaz de usuario (ejemplo: HTML, CSS, JavaScript, React).
- Backend: lenguajes y frameworks para la lógica del servidor (ejemplo: Python con Django, Node.js).
- Base de datos: sistemas de almacenamiento (ejemplo: MySQL, PostgreSQL, MongoDB, entre otros).
- Infraestructura y despliegue: servidores y servicios en la nube (ejemplo: AWS, Docker, Kubernetes).

Factores para elegir un stack

La elección del stack depende de los requerimientos del proyecto, considerando:

- Objetivo (visualización de datos, reportes, análisis).
- Escalabilidad y rendimiento (volumen de datos, usuarios).
- Compatibilidad con otros sistemas.
- Costo y recursos disponibles.

Seleccionar el stack adecuado es muy importante para el éxito y eficiencia del proyecto.

1.2 IMPLEMENTACIÓN DE LA PROPUESTA DE MEJORA

En un análisis de datos, la selección de la metodología de trabajo, **los entregables o artefactos por fases o sprints**, así como las actividades, dependerán de las características específicas de cada proyecto. La explicación se debe presentar de manera teórica y técnica lo más completa posible para proporcionar un panorama claro de las acciones a realizar. En el caso de usar una metodología ágil, cada sprint puede tener uno o más entregables; la elección de estos recae en el investigador que lidera el trabajo.

La **implementación de un dashboard** se centra en el **diseño, desarrollo e implementación** de una herramienta interactiva que optimiza la visualización y el análisis de datos. La adopción de la metodología Scrum y el enfoque de Diseño Centrado en el Usuario (DCU) garantizan que el producto final satisfaga las necesidades de los usuarios y facilite una toma de decisiones más efectiva basada en datos.

A continuación, se detallan las actividades realizadas durante la fase de implementación, proporcionando una descripción clara y completa de cada una para visualizar con precisión el trabajo llevado a cabo y los entregables correspondientes.

Fases de Implementación

- **Planificación y Diseño del Dashboard (Sprint 1)**

El **objetivo** es establecer los requisitos detallados y el diseño preliminar del dashboard. Las **actividades a realizar** son:

1. **Investigación de usuarios:** llevan a cabo entrevistas, encuestas y sesiones de observación para identificar las necesidades, expectativas y puntos débiles de los usuarios en relación con el acceso, la interpretación y el uso de los datos.
2. **Análisis de requisitos:** analizan los datos recopilados para determinar las funcionalidades esenciales del dashboard, los indicadores clave de rendimiento (KPI) y las métricas específicas que utilizarán para el seguimiento y la evaluación.
3. **Diseño de la estructura:** define la arquitectura de la información del dashboard, incluyendo la organización de los módulos, la navegación y la jerarquía de la información. Crean diagramas de flujo y mapas de sitio para visualizar la estructura del dashboard.
4. **Creación de Wireframes y Prototipos:** generan wireframes de baja fidelidad para representar la estructura básica de las pantallas del dashboard. Luego, desarrollan prototipos interactivos de alta fidelidad para simular la funcionalidad y la interacción del usuario.
5. **Revisión con usuarios:** realizan pruebas de usabilidad con usuarios finales para validar los prototipos, recopilar retroalimentación y realizar ajustes en el diseño y la funcionalidad.
6. **Presentación de resultados del sprint:** llevan a cabo una revisión formal del sprint con los stakeholders para presentar los avances, discutir los desafíos y obtener la aprobación para continuar con la siguiente fase.

- **Desarrollo de Funcionalidades Iniciales (Sprint 2)**

El objetivo es implementar las funcionalidades básicas del dashboard priorizadas en el Product Backlog. Las **actividades a realizar** son:

1. **Análisis de flujos de trabajo:** definen los flujos de trabajo detallados para cada funcionalidad, incluyendo las interacciones del usuario, el procesamiento de datos y las respuestas del sistema.
2. **Desarrollo del módulo de gestión de datos:** crea la capa de procesamiento de datos que encargará de la extracción, transformación y carga (ETL) de los datos desde las fuentes de datos relevantes. Implementan las consultas a la base de datos y la lógica de negocio necesaria para el procesamiento de los datos.
3. **Pruebas de integración inicial:** realizan pruebas unitarias y pruebas de integración para verificar la correcta conexión entre los diferentes módulos del dashboard y la base de datos.
4. **Implementación de visualizaciones iniciales:** crean los primeros paneles y gráficos del dashboard, utilizando herramientas de visualización de datos. Seleccionan los tipos de gráficos más adecuados para representar cada métrica y KPI.
5. **Mejoras según retroalimentación:** realizan ajustes y mejoras en el diseño y la funcionalidad del dashboard en función de la retroalimentación recibida durante las pruebas y revisiones.
6. **Demostración del módulo funcional:** realizan una demostración del módulo funcional a los stakeholders para presentar los avances y obtener retroalimentación adicional.

- **Iteración y Refinamiento del Dashboard (Sprint 3)**

El objetivo es optimizar las funcionalidades existentes y mejorar la usabilidad del dashboard. Las **actividades a realizar** son:

1. **Análisis de flujos de trabajo y comportamiento del usuario:** analizan los datos de uso del dashboard y el feedback de los usuarios para identificar áreas de mejora en los flujos de trabajo y la experiencia del usuario.
2. **Iteración de diseño basado en feedback:** realizan ajustes en el diseño de la interfaz y la funcionalidad del dashboard en función del feedback de los usuarios y los resultados del análisis de uso.
3. **Implementación de funcionalidades avanzadas:** implementan funcionalidades avanzadas, como filtros interactivos, opciones de personalización, alertas y capacidades de exportación de datos.

4. **Pruebas de usabilidad con datos reales:** realizan pruebas de usabilidad con usuarios reales utilizando datos de producción para evaluar el rendimiento del dashboard y detectar posibles errores o cuellos de botella.
5. **Ajustes de diseño:** realizan ajustes finales en el diseño y la funcionalidad del dashboard para perfeccionar la interfaz y garantizar una experiencia de usuario óptima.
6. **Presentación y retroalimentación:** realizan una presentación final del dashboard a los stakeholders para obtener su aprobación antes del despliegue.

- **Optimización y Pruebas Finales (Sprint 4)**

El objetivo es optimizar el rendimiento, la estabilidad del dashboard y realizar pruebas finales exhaustivas.

Las **actividades a realizar** son:

1. **Optimización de rendimiento:** realizan optimizaciones técnicas en el código, las consultas a la base de datos y la configuración del servidor para reducir los tiempos de carga y mejorar la capacidad de respuesta del dashboard.
2. **Pruebas de usabilidad:** realizan pruebas de usabilidad finales con un grupo representativo de usuarios para identificar cualquier problema restante y asegurar que el dashboard cumple con los requisitos de usabilidad y accesibilidad.
3. **Ajustes finales:** implementan ajustes finales en el dashboard en función de la retroalimentación recibida durante las pruebas finales. La presentación del producto en donde se realizan una revisión final del dashboard con los stakeholders para obtener la aprobación antes del despliegue.
4. **Implementación y despliegue:** una vez completadas las pruebas y obtenido la aprobación de los stakeholders, proceden a la implementación del dashboard en el entorno de producción. Esto incluye:

Preparación del entorno de producción: configuran el servidor, la base de datos y demás componentes de software necesarios para el funcionamiento del dashboard en el entorno de producción.

Despliegue del dashboard: despliegan la aplicación del dashboard en el servidor de producción y realizan pruebas finales para asegurar su correcto funcionamiento.

5. **Capacitación de usuarios:** imparten talleres de capacitación a los usuarios finales para asegurar que comprenden cómo utilizar el dashboard de manera efectiva y aprovechar al máximo sus funcionalidades.
6. **Estrategia de mantenimiento y soporte:** establecen un plan de mantenimiento y soporte técnico para el dashboard, incluyendo la definición de responsables, procesos de actualización y resolución de problemas.
7. **Evaluación de impacto:** definen un plan de evaluación para medir el impacto del dashboard en el análisis de datos y la toma de decisiones, incluyendo el seguimiento de métricas, la recopilación de feedback de los usuarios y la identificación de áreas de mejora continua.



Importante:

En el contexto de proyectos de desarrollo de dashboards, los términos **entregables** y **artefactos a entregar** se usan de manera intercambiable. Ambos hacen referencia a los elementos específicos que deben presentarse al finalizar cada fase del proyecto.

- **Entregables:** son los resultados finales entregados a los usuarios o partes interesadas, como el dashboard funcional, la documentación técnica o los reportes de validación.
- **Artefactos a entregar:** se refiere a los productos específicos generados a lo largo del desarrollo, como wireframes, prototipos, consultas a la base de datos, código fuente y documentación de diseño, que conforman los entregables finales.

Ambos términos resaltan el carácter tangible del trabajo realizado en cada iteración del proyecto.

EJEMPLO

Propuesta: Desarrollo de un dashboard analítico que optimice la gestión de cobranzas en FinRec Solutions.

Fases de Implementación

1. Planificación y Diseño del Dashboard (Sprint 1)

En esta fase, se establecieron los requisitos y el diseño preliminar del dashboard para optimizar la gestión de cobranzas en FinRec Solutions.

La investigación de usuarios incluyó entrevistas y encuestas para identificar sus necesidades. A partir de estos datos, se definieron las funcionalidades clave, los KPI y las métricas esenciales.

Se diseñó la estructura del dashboard, organizando módulos y navegación, y se crearon wireframes y prototipos interactivos para visualizar su funcionamiento. Luego, fueron validados mediante pruebas de usabilidad para recopilar retroalimentación y realizar ajustes. Finalmente, el avance del sprint fue presentado a los stakeholders para su aprobación y continuidad del proyecto.

Requisitos principales:

- Conexión con bases de datos para obtener información en tiempo real sobre cobranzas.
- Visualización de indicadores clave (KPI) como montos pendientes, pagos recibidos y tasas de recuperación.
- Filtros dinámicos por cliente, periodo y estado de cobranza.
- Interfaz intuitiva con gráficos interactivos para facilitar el análisis de datos.
- Accesibilidad y seguridad en el acceso a la información.

Sprint 2: Desarrollo de funcionalidades iniciales

- **Módulo de gestión de datos:** La **figura 1** ilustra la integración del chatbot con el dashboard, facilitando la identificación de los datos clave a capturar y su visualización.
- **Implementación de visualizaciones iniciales:** Sirve como referencia para diseñar los primeros paneles del dashboard relacionados con la actividad del chatbot en la gestión de cobranzas.

Sprint 3: Iteración y refinamiento del dashboard

- **Análisis de flujos de trabajo y comportamiento del usuario:** Permite evaluar la utilidad de la información proporcionada por el chatbot a los agentes y determinar si requieren datos adicionales o una mejor presentación.
- **Pruebas de usabilidad con datos reales:** Esta imagen ayuda a verificar si el dashboard refleja correctamente la información del chatbot y si los usuarios pueden acceder fácilmente a los datos relevantes.
- **Revisión y ajustes finales de diseño:** La retroalimentación de usuarios y stakeholders sobre este dashboard específico puede derivar en mejoras en diseño y funcionalidad.

Sprint 4: Optimización y pruebas finales

- **Pruebas de usabilidad finales:** Se verifica que el dashboard cumpla con las expectativas en cuanto a la información del chatbot y su impacto en la gestión de cobranzas.

- **Revisión y presentación del producto final:** Este dashboard se presenta como parte de la solución final, destacando el papel del chatbot en la optimización del proceso de cobranza.

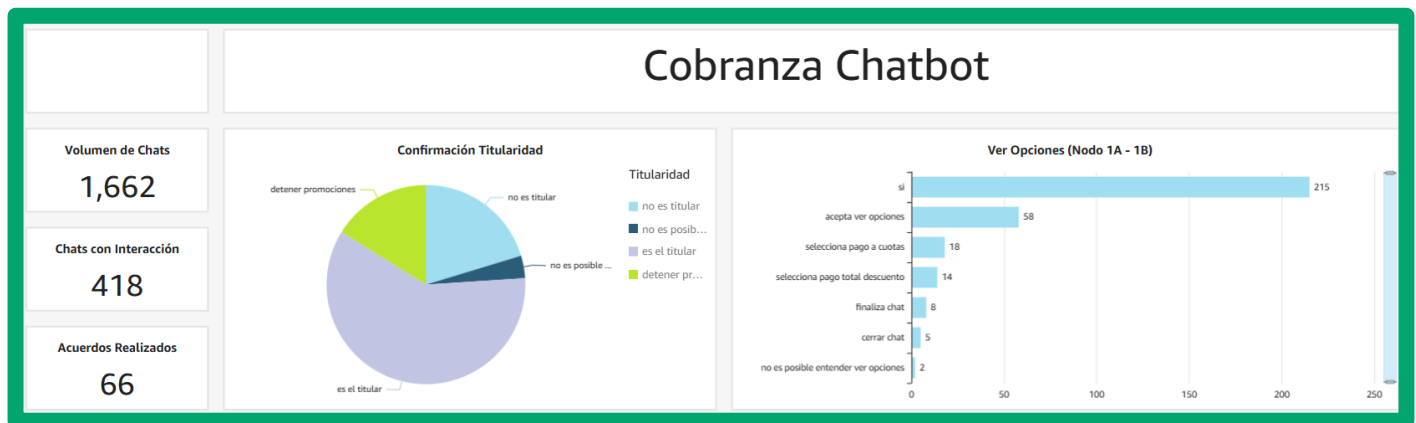


Figura 1. Ejemplo de visualización de datos del chatbot en el dashboard de cobranzas

Fuente: Elaboración propia

La **Figura 1** muestra el dashboard de cobranza diseñado. Se identifican las siguientes secciones:

- **Volumen de Chats:** indica la cantidad total de conversaciones iniciadas en el chatbot.
- **Chats con Interacción:** muestra el número de chats donde hubo una interacción significativa (presumiblemente con un agente o sistema).
- **Acuerdos Realizados:** registra la cantidad de acuerdos de pago logrados a través del chatbot.
- **Confirmación Titularidad:** gráfico que muestra la distribución de la confirmación de titularidad (si la persona que interactúa es el titular de la deuda).
- **Ver Opciones (Nodo 1A-1B):** detalle de las opciones que se le ofrecen al usuario en un punto específico de la conversación (ejemplo: ver promociones, seleccionar pago en cuotas, entre otros).

Visualicen el análisis del Dashboard de contactabilidad por hora



Figura 2. Ejemplo de dashboard de contactabilidad por hora en cobranzas

Fuente: Elaboración propia

La **Figura 2** muestra un dashboard que visualiza la contactabilidad por hora en el área de cobranzas. Los datos están organizados en una tabla donde se registran los intentos de contacto por fecha y rango horario.

Principales características:

- **Estructura de la tabla:** Ordenada por fecha y franjas horarias, permitiendo analizar los momentos de mayor actividad.
- **Datos numéricos:** Cada celda indica la cantidad de contactos o intentos de contacto en un período específico.
- **Comparación visual:** La disposición de los datos facilita la identificación de patrones y tendencias en la contactabilidad.

Uso en los sprints del proyecto**Sprint 3: Iteración y refinamiento del dashboard**

- **Análisis de flujos de trabajo:** Permite evaluar cómo los agentes de cobranza utilizan la información y detectar horarios óptimos para la gestión.
- **Pruebas de usabilidad con datos reales:** Facilita la validación del diseño del dashboard, garantizando que los usuarios puedan interpretar los datos eficientemente.

- **Ajustes finales de diseño:** Basado en la retroalimentación, se pueden realizar mejoras visuales, como la incorporación de gráficos o indicadores que resalten patrones de contactabilidad.

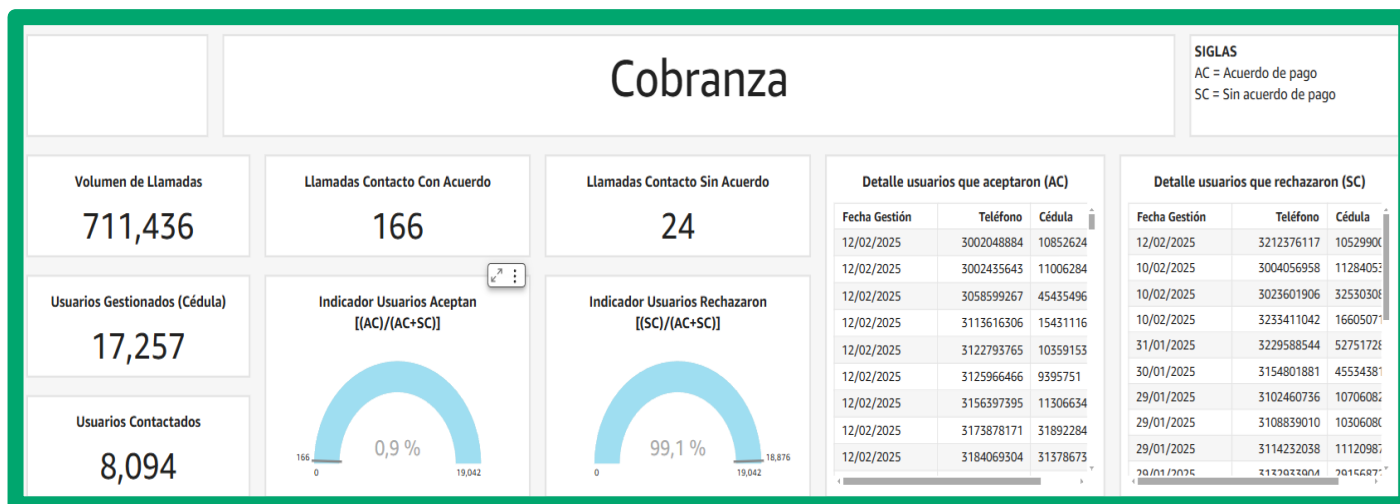


Figura 3. Dashboard con indicadores clave para la gestión de cobranzas.

Fuente: Elaboración propia

La **Figura 3** presenta un dashboard con métricas clave para la gestión de cobranzas, incluyendo:

- Volumen de Llamadas: Total de interacciones realizadas.
- Llamadas con acuerdo y sin acuerdo: Diferenciación entre gestiones exitosas y sin éxito.
- Detalle de usuarios: Información detallada de los clientes contactados.
- Usuarios gestionados y contactados: Cantidad de personas atendidas.
- Indicadores de acuerdo/rechazo: Porcentajes de aceptación o rechazo de acuerdos de pago.

Uso en los sprints del proyecto

Sprint 3: Iteración y refinamiento del dashboard

- **Análisis de flujos de trabajo:** Permite evaluar la claridad y utilidad de la información presentada.
- **Pruebas de usabilidad con datos reales:** Verifica si el dashboard muestra la información correctamente.
- **Ajustes finales de diseño:** Posibles mejoras en la visualización para optimizar su uso.

Sprint 4: Optimización y pruebas finales

- **Pruebas de usabilidad finales:** Asegura que el dashboard cumple con los requerimientos del usuario.

- **Presentación del producto final:** Muestra a los stakeholders cómo el dashboard optimiza la gestión de cobranzas.



Importante:

Cada metodología de desarrollo, junto con el criterio del investigador, determinará los diagramas que sean más relevantes o necesarios para el proyecto.

Fase de Diseño y entregables del dashboard de cobranzas: Un ejemplo

En la fase de diseño del dashboard de cobranzas, se definen las visualizaciones clave y la organización de la información para garantizar una gestión eficiente. Para ello, se utilizan diversas herramientas y metodologías, incluyendo:

- **Definición de KPIs y métricas clave:** identificación de indicadores esenciales, como volumen de interacciones, acuerdos logrados y tasas de contacto.
- **Estructura del dashboard:** organización de módulos y secciones para una navegación intuitiva.
- **Diseño de visualizaciones:** selección de gráficos y tablas que representen datos de forma clara y comprensible (ejemplo. gráficos de barras, tablas dinámicas y mapas de calor).
- **Wireframes y prototipos interactivos:** creación de versiones preliminares del dashboard para evaluar funcionalidad y facilidad de uso.
- **Pruebas de usabilidad:** validación con usuarios para asegurar que la presentación de datos facilite la toma de decisiones.

Resumen de entregables del proyecto

- **Prototipos y diseño:** Wireframes, diagramas de navegación y diseño final de la interfaz.
- **Dashboard funcional:** implementación en la herramienta seleccionada con KPIs, interactividad y filtros.
- **Base de datos:** modelo de datos y proceso de actualización.
- **Pruebas y validaciones:** evaluaciones de usabilidad y ajustes según retroalimentación.

La implementación de un dashboard requiere una planificación estructurada que abarque la capacitación del personal, la implementación técnica y un seguimiento continuo para garantizar su éxito. Primero, es importante capacitar a los usuarios en el uso, verificando que comprendan sus funcionalidades, la interpretación de los datos y la generación de informes. Además, se debe proporcionar soporte continuo para resolver dudas y optimizar su uso.

La implementación técnica implica configurar servidores, conectar fuentes de datos, instalar el software necesario y realizar pruebas para asegurar el correcto funcionamiento del sistema antes de su despliegue definitivo.

Por último, el seguimiento y la evaluación permiten detectar oportunidades de mejora, resolver problemas técnicos y medir el impacto del dashboard en la toma de decisiones y la eficiencia operativa.

Una implementación efectiva requiere formación adecuada, un despliegue técnico riguroso y una evaluación constante para garantizar que el dashboard cumpla con sus objetivos.



Importante:

Entregables técnicos clave en el desarrollo de un dashboard

El desarrollo de un dashboard requiere una planificación estructurada y la entrega de componentes clave en cada fase. A continuación, se presentan los entregables más importantes en cada etapa del proceso:

1. Planificación

- **Requerimientos funcionales y técnicos:** documento que define las necesidades del negocio, los objetivos del dashboard y las restricciones técnicas.
- **Identificación de KPIs:** selección de métricas clave que serán visualizadas en el dashboard.
- **Análisis de fuentes de datos:** evaluación de los datos disponibles y su integración con el sistema.

2. Diseño

- **Diagramas de arquitectura:** representación visual de la estructura del dashboard y sus componentes.
- **Wireframes:** esquemas de baja fidelidad que muestran la disposición de los elementos en las pantallas.
- **Mockups:** diseños de alta fidelidad que simulan la apariencia final del dashboard.
- **Prototipos interactivos:** versiones funcionales preliminares que permiten probar la navegación y la usabilidad.
- **Especificaciones de diseño:** documento detallado con requisitos visuales, funcionales y de usabilidad.

3. Desarrollo

- **Código fuente:** implementación del dashboard, incluyendo lógica de negocio, interfaz de usuario y conexión con la base de datos.
- **Módulos funcionales:** componentes del sistema que implementan funcionalidades específicas.
- **Documentación técnica:** manuales y guías sobre el funcionamiento del código y la arquitectura del sistema.
- **Scripts de base de datos:** código para la creación, actualización y gestión de la base de datos.

4. Pruebas

- **Casos de prueba:** documentos que describen los escenarios de prueba y los resultados esperados.
- **Informes de pruebas:** registro de los errores detectados y las soluciones aplicadas.
- **Pruebas de rendimiento:** evaluación del tiempo de respuesta y capacidad del dashboard.
- **Pruebas de usabilidad:** análisis de la experiencia del usuario para mejorar la interfaz y la navegación.

5. Implementación

- **Dashboard desplegado:** versión final instalada en el entorno de producción.
- **Documentación de usuario:** manuales y guías para el uso del dashboard por los usuarios finales.
- **Plan de mantenimiento:** estrategia para la actualización y mejora del dashboard a largo plazo.
- **Evaluación de impacto:** informe sobre el efecto del dashboard en la toma de decisiones y la eficiencia operativa.

COMENTARIO FINAL

El desarrollo de un dashboard como solución analítica requiere una planificación estratégica que garantice su efectividad en la toma de decisiones. La entrega de componentes clave, como la definición de KPIs, la integración de fuentes de datos, el diseño de visualizaciones intuitivas y la validación con usuarios, permite construir una herramienta funcional y orientada a las necesidades del negocio.

La implementación de un prototipo inicial facilita la iteración y refinamiento del dashboard, asegurando que los datos se presenten de manera clara y útil para los usuarios finales. A través de un proceso estructurado que incluye diseño, desarrollo, pruebas y ajustes, se optimiza la experiencia del usuario y se mejora la calidad de los análisis obtenidos.

Además, la asignación de recursos adecuados, tanto humanos como tecnológicos, es fundamental para garantizar el éxito del proyecto. La correcta identificación de roles y responsabilidades permite una ejecución eficiente y alineada con los objetivos estratégicos de la organización.

En conclusión, un dashboard bien diseñado y desarrollado no solo soluciona el problema inicial de visualización de datos, sino que también impulsa la optimización de procesos y la generación de nuevas oportunidades de mejora en el análisis de información.

REFERENCIAS

IACC (2020). *Propuesta técnica para el desarrollo de proyecto informático*. Taller de Integración Informática. Semana 5.

IACC (2021). *Implementación de la metodología seleccionada*. Taller de Integración y Programación. Semana 5.

IACC. (2022). *Desarrollo del sistema*. Taller de Integración de Redes. Semana 5.

PARA REFERENCIAR ESTE DOCUMENTO, CONSIDERE:

IACC (2025). *Diseño de la propuesta*. Taller de Integración en Análisis de Datos. Semana 5.