# ESTRATEGIAS DE DISEÑO DE ARQUITECTURAS DE SOFTWARE Y SU APLICACIÓN EN CASOS REALES

Somos una asociación de empresas dedicada a transmitir conocimiento sobre implementación de productos de software, usando plataforma open source empresarial a través de nuestros cursos de capacitación
Jr. Huascar 359 Ofc 501 - los Olivos. RUC: 20602629512
Telf: 994589783 - B. SOFT GROUP

**OBJETIVO:** El curso está orientado en capacitar a los participantes sobre diferentes perspectivas de arquitecturas de software estandarizadas, teniendo en cuenta los estándares definidos por: IEEE, ISO/IEC 25010, OWASP y GARTNER; para que a partir de ello puedan definir sus estrategias de diseño, basado en los escenarios del negocio que están evaluando. Vamos a entender las diferentes vistas evolutivas de las arquitecturas de software: modelos de arquitecturas evolutivas, niveles de madurez, tópicos tecnológicos, arquitecturas basado en capas, cuadro de congruencia de las diferentes vistas evolutivas; dimensionamiento de recursos como: memoria, procesador, ancho de banda, espacio en disco. Se va a diseñar cada una de las arquitecturas de referencia, según el proceso evolutivo, considerando los estándares, patrones, las estrategias de su implementación y recomendaciones. Vamos a diseñar y explicar con casos reales (3), la estrategia del diseño, usando como base los conceptos de las diferentes arquitecturas explicadas, según el contexto del escenario abordado.

Vamos a explicar y entregar los documentos oficiales que se definen como estándares y recomendaciones de arquitectura:

- ISO/IEC – International Standard 25010. System and Software Engineering - System and Software Quality – System and Software quality models.
- IEEE - Architecture, Design and Implementation.
- GARTNER - Software-Defined Architecture for Applications in Digital Business.
- OWASP - Open Web Application Security Project

## EJECUCIÓN

- ☑ Docentes: 3
  - Especialista Mainframe
  - Especialista Cliente Servidor Web y Mainframe
  - Especialista en Componentes Distribuidos web y Cloud
- ☑ Requisitos: Conocer a nivel teórico los procesos de desarrollo de software.
- ☑ Perfil del Participante: Analista, Desarrollador, Jefe de Proyecto, Funcional, Responsable de Equipo de Desarrollo, Analista Técnico, Analista Funcional

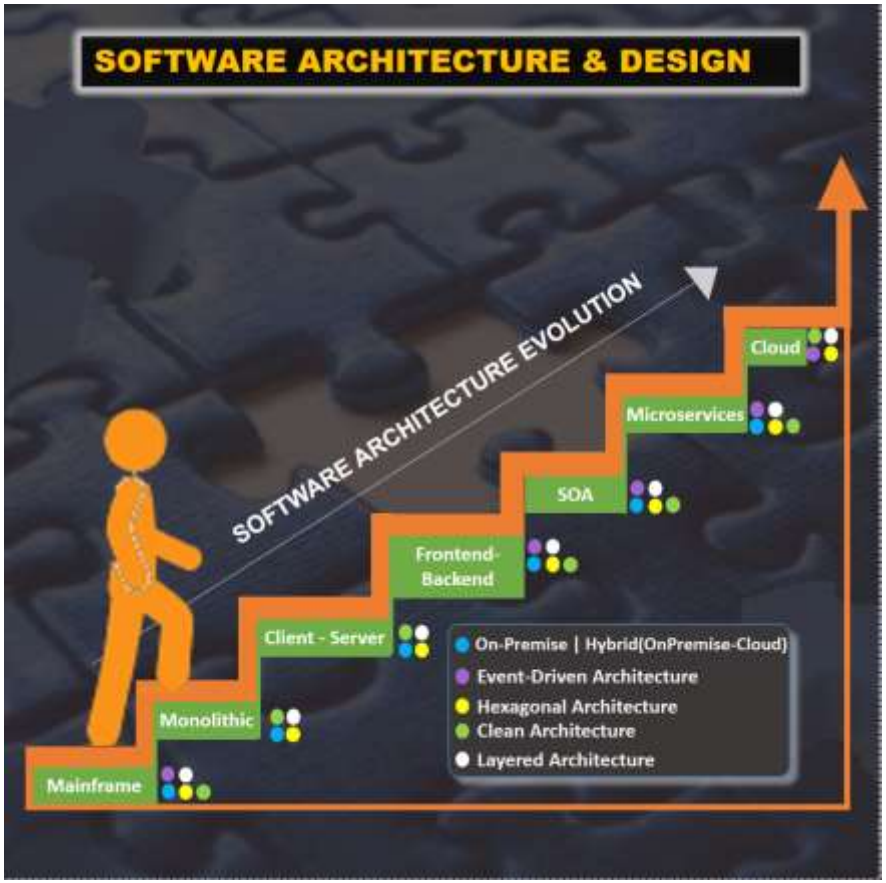
## RESULTADO DEL CURSO

El participante adquiere conocimientos necesarios para diseñar las diferentes vistas de arquitecturas de software definiendo los elementos necesarios para la implementación

- ☑ Entrega de Certificado al Cierre del Curso.
- ☑ Diseño de las arquitecturas de referencia, estrategias y recomendaciones.
- ☑ Guía Consolida del curso

## TEMARIO

1. **Arquitectura de Software - Perspectivas de Arquitecturas de Software.**




2. **Modelos de arquitecturas evolutivas - tópicos tecnológicos - madurez y capas)**
   - 2.1. Arquitectura Mainframe.
   - 2.2. Arquitectura monolítica y aisladas.
   - 2.3. Arquitectura Cliente Servidor
   - 2.4. Arquitectura Distribuida Web 3 y N capas.
   - 2.5. Arquitectura Orientadas a Servicios-SOA
   - 2.6. Arquitectura basada en Microservicios.
   - 2.7. Arquitectura de Full Apis Cloud Services.
   - 2.8. Recomendaciones y Tendencias.

3. **Vistas de Arquitecturas según el nivel evolutivo.**
   - 3.1. Vista Integral(Procesos).
   - 3.2. Vista Implementación(Desarrollo)
   - 3.3. Vista de Física.
   - 3.4. Vista de Lógica.
   - 3.5. Recomendaciones y Tendencias.

4. **Estándares y patrones de componentes(IEEE)**
   - 4.1. Facade Design Pattern
   - 4.2. Intercepting Filter Design Pattern
   - 4.3. Factory Design Pattern
   - 4.4. Singlenton Design Pattern
   - 4.5. Publish-Susbcribe Pattern
   - 4.6. Proxy Design Pattern
   - 4.7. Observer Design Pattern
   - 4.8. Decorator Design Pettern
   - 4.9. Composite Design Pattern
   - 4.10. Recomendaciones.

5. **Sizing y Software Base.**
   - 5.1. Esquemas de Desarrollo
   - 5.2. Componentes de desarrollo
   - 5.3. Dimensionar Memoria para la aplicación.
   - 5.4. Dimensionar procesador para la aplicación.
   - 5.5. Dimensionar ancho de banda.
   - 5.6. Seguridad Componentes.
   - 5.7. Balanceo - Software Base.
   - 5.8. **Software base y dimensionamiento basado en el nivel de arquitectura:** Bases de datos – Aplicación – Componentes Reusable – Middleware – Balanceo – Trazabilidad – Middleware – ESB – MQ – Socket – Protocolos de Comunicación – Web – BPM – Integradores.

5.9. Recomendaciones y conclusiones.

**6. Atributos de Software basado en ISO ISO/IEC 25010**

6.1. **Desempeño – Eficiencia:** Uso de Recursos (memoria, procesador, etc.) - Capacidad - Procesamiento

6.2. **Compatibilidad:** Interoperabilidad - Coexistencia.

6.3. **Usabilidad:** Protección frente a errores – Accesibilidad - Protección frente a errores - Operabilidad.

6.4. **Fiabilidad:** Madurez – Disponibilidad - Tolerancia a fallo. - Capacidad de recuperación.

6.5. **Seguridad:** Confidencialidad - Integridad - No Repudio – Autenticidad - Responsabilidad.

6.6. **Mantenibilidad:** Modularidad – Reusabilidad – Trazabilidad – Modificable – Validable.

6.7. **Portabilidad:** Adaptabilidad – Fácil Despliegue – Migrable.

6.8. Recomendaciones y Sugerencias.

**7. Stream Processsing(Procesamiento de Flujo)**

7.1. Data Ingestion

7.2. Forma de Ingestión de Datos

7.3. Uso de Ingestión de Datos.

7.4. Procesamiento de Data Distribuida

7.5. Arquitectura Lambda

7.6. Tendencias.

**8. Mobile Apps**

8.1. Nativo e Hibrido Code

8.2. Tipos de Mobile Apps

8.3. Mobile and Services

8.4. Tendencias.

**9. Estándares.**

9.1. ISO/IEC – International Standard 25010. System and Software Engineering - System and Software Quality – System and Software quality models.

9.2. IEEE - Architecture, Design and Implementation.

9.3. GARTNER - Software-Defined Architecture for Applications in Digital Business.

9.4. OWASP - Open Web Application Security Project

**10. CASOS REALES A DISEÑAR**

12.1. INTEGRACIÓN INTER - BANCARIA PARA PROCESOS DE TRANSFERENCIAS ENTRE ENTIDADES FINANCIERAS EN MODALIDAD ONLINE.

12.2. PROPUESTA DE DISEÑO Y ESTRATEGIA DE IMPLEMENTACIÓN DE UNA ARQUITECTURA BASE BASADA EN MICROSERVICIOS CONSIDERANDO ASPECTOS DE PROYECCIÓN DEL NEGOCIO.

12.3. DISEÑO “FRONT-END DE GESTIÓN DE TÍTULOS VALORES E INTEGRACIÓN AL SISTEMA CORE EMPRESARIAL DE ADMINISTRACIÓN DE TÍTULOS VALORES”

# INTEGRACIÓN INTER - BANCARIA PARA PROCESOS DE TRANSFERENCIAS ENTRE ENTIDADES FINANCIERAS EN MODALIDAD ONLINE.

**1. Contexto:** Un País X, cuenta con una institución que regula los procesos de los bancos y sus interacciones con los clientes, bajo temas normativos. Dicha institución ha definido que los bancos generen un **proceso online** para las transferencias interbancarias, las cuales a la fecha se manejan en modo diferido. Con ello el país espera dinamizar la economía haciendo que el flujo del dinero sea más rápida y segura.

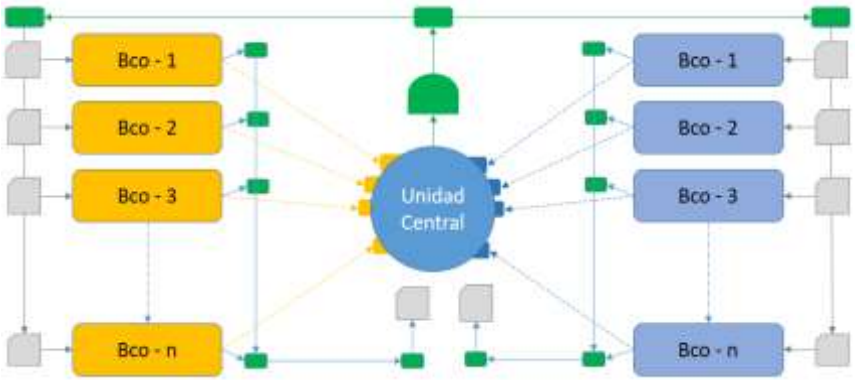
## 2. Consideraciones del negocio:

1. Todo el proceso de la transferencia online debe de pasar por una unidad centralizada que canaliza las transacciones. Es decir, los bancos no deben hacer transacciones directas.
2. Las transferencias online deben tener un tiempo máximo de 2 seg desde que se genera la transacción. Es decir, entre la unidad centralizadora y el banco destino los tiempos deben de sumar como máximo 2 s., entre la solicitud y respuesta:
   **T(Uc) + T(Bd) < 2s;** *Tc= Unidad centralizada, Bd = Bco Destino.*
3. Las transacciones tienen un monto máximo de 10 mil dólares/30 mil soles entre 5am – 10 pm. En otros casos un máximo de 5 mil dólares. 10 mil soles.
4. Los bancos y la unidad centralizada deben asegurar que el proceso sea seguro.
5. Para homologar transacciones cada banco debe de reportar en 4 eventos en el día, (12-16-20-24) horas, a la Unidad Centralizada toda la información de las transferencias en línea. Ante eventos de falla debe de activarse un proceso de contingencia para subsanar la inconsistencia.
6. Al cierre del día (1 am) la Unidad Centralizada debe homologar la información de las transferencias de todos los bancos y generar un reporte consolidado a cada banco de las transferencias que le generaron todos los bancos, dicho detalle debe contener.
   a. Id único Transacción(7)
   b. Dni del Ordenante(7)
   c. Dni del Beneficiario(7)
   d. Cuenta Origen(20)
   e. Cuenta Destino(20)
   f. Monto de la transacción(10,2)
   g. Fecha hora de inicio de la solicitud de cada unidad involucrada.(14)
   h. Fecha Hora de repuesta de cada entidad involucrada(14)
7. Todas las transacciones no homologadas deben de ser generadas en un archivo de pendientes y deben de ser reportada a cada banco, según sea el caso. El plazo de subsanación debe ser menor a 4 horas. Cada banco debe de informar a sus clientes sobre los motivos de retraso de sus transferencias.
8. Deben de controlarse los eventos de excepciones de modo que se asegure el proceso de la transacción.
9. Se espera 1000 transacciones por segundo en horas picos y en proceso normal 30 transacciones diarias y por normativa la información puede almacenarse por 6 meses

## 3. Consideraciones Técnicas:

1. Usar Red privada punto a punto con procesos de doble nateo de ips.
2. Encriptar el canal de transmisión y el contenido de la información.
3. Definir timeout entre los nodos interconectados banco origen – unidad centralizada – banco destino.
4. Definir el ancho de banda para el proceso de transmisión de los mensajes y la velocidad de transmisión entre los nodos.
5. Definir un esquema de desacoplamiento de componente, de modo que permita realizarse tareas en paralelo sin afectar el proceso crítico del negocio.
6. Definir el mejor esquema de transmisión de datos, a través de los diferentes protocolos de transmisión que existen.
7. Definir compontes y apis de homologación para la carga y procesamientos de la información reportada
8. Schedule los diferentes Apis de modo que se agilice la ejecución de las diferentes reglas del negocio.
9. Controlar los niveles de excepciones en cada punto involucrado desde que se origina la transacción hasta que se responde, pasando por la unidad centralizada y el banco destino en ambos flujos ida y vuelta.
10. El proceso de intercambio de archivo debe ser de modo seguro.
11. Dimensionar los recursos: disco, memoria, procesador. Basado en las condiciones del negocio.

## 4. Flujo de Procesos



# DISEÑO "FRONT-END DE GESTIÓN DE TÍTULOS VALORES E INTEGRACIÓN AL SISTEMA CORE EMPRESARIAL DE ADMINISTRACIÓN DE TÍTULOS VALORES"

**1. Contexto:** Una entidad del rubro financiero necesita implementar una solución web de tipo empresarial (front-end) para sus clientes PPJJ que negocian a través de la entidad sus títulos valores (facturas y letras). Esto como parte de su estrategia de competitividad, para brindar a su cartera de clientes la facilidad de realizar las gestiones desde sus oficinas o dispositivos móviles vía internet.

## 2. Consideraciones del negocio:

1. Diariamente se ingresan aproximadamente 12mil letras y 15mil facturas a través del canal tradicional (agencia).
2. La estrategia es derivar gradualmente en un lapso de 6 meses a 1 año el 75% del ingreso de estos valorados al nuevo canal web, ofreciéndoles mejores tasas y rapidez en financiamiento de sus títulos valores.
3. Este nuevo canal web debe ser capaz de integrarse con el aplicativo legacy que administra y procesa los títulos valores que se encuentra instalado sobre plataforma centralizada (mainframe z/OS). El sistema core empresarial, cuenta con programas de interfaz online para proveer las funcionalidades necesarias al front-end, estas son: • Registro de planillas de letras/facturas. • Consulta de letras/facturas vigentes, vencidas, canceladas, devueltas. • Exportación de dietario de letras/facturas. • Pago de letras/facturas. • Devolución de letras/facturas. • Financiamiento de letras/facturas. Pero uno de los principales retos técnicos, es el disponibilizar el acceso a los programas online de interfaz nativa en plataforma mainframe, hacia el front-end (ambiente distribuido). Por tal motivo, se debe analizar, diseñar e implementar un esquema de integración que cumpla con los estándares y patrones de integración de aplicaciones en plataformas heterogéneas.
4. Primer paso es clasificar el patrón de integración a implementar, según:
    - La función o tarea.
    - El foco de la integración.
    - El modo de conectividad desplegado.
    - La topología objetivo.

Finalmente, en función a la clasificación mencionada se debe seleccionar el tipo de patrón aplicativo de integración (orientado al proceso u orientado a los datos) idóneo para dar solución al proyecto y sobre el cual el front-end pueda conectarse y consumir los servicios del sistema core