Sistema de Integración Nubox

Documentación Técnica Completa

Versión 1.0 | Septiembre 2025

Tabla de Contenidos

16. Roadmap

17. Conclusión

1. Contexto del Problema

1.1 Desafío de Negocio

El sistema debe resolver la integración entre dos sistemas críticos para el procesamiento de nóminas y control de asistencia:

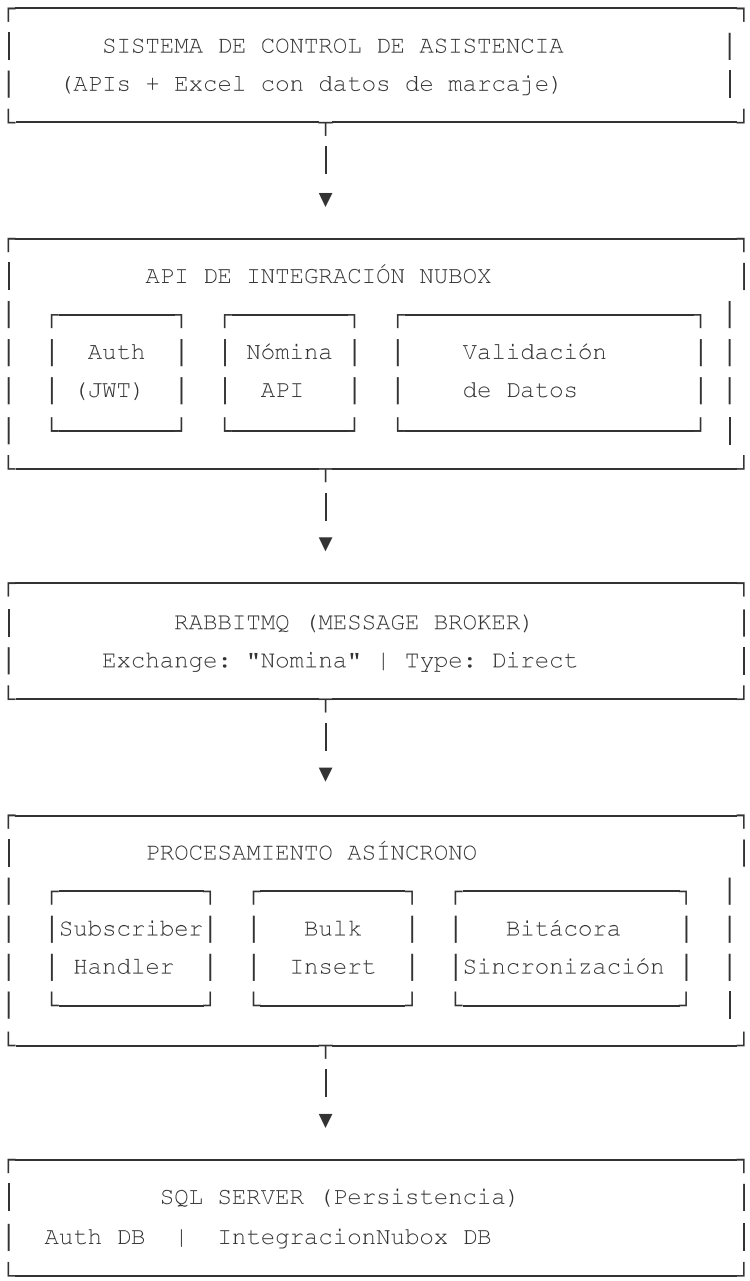
- **Sistema de Cálculo de Liquidaciones:** Procesa nóminas con diferentes periodicidades (semanal, quincenal, mensual) para **100,000 empresas**
- **Sistema de Control de Asistencia:** Proveedor externo que proporciona dispositivos físicos/tecnológicos para el registro de marcajes de asistencia

1.2 Requerimientos Clave

> **Objetivos Principales:**
> - Intercambio seguro de información entre ambos sistemas
> - Procesamiento de datos de asistencia (horas trabajadas, horas extras, licencias médicas)
> - Soporte para múltiples formatos: APIs REST y archivos Excel
> - Arquitectura escalable, observable, trazable y resiliente

2. Arquitectura de la Solución

2.1 Diseño de Alto Nivel



2.2 Características de la Arquitectura

Seguridad Escalabilidad Observabilidad Trazabilidad Resiliencia

3. Componentes Principales

3.1 Capa de API (ASP.NET Core 8.0)

3.1.1 Autenticación y Seguridad

- **JWT Bearer Authentication:** Tokens seguros con expiración de 24 horas
- **Endpoint de Login:** `/api/auth/login`
- **Claims personalizados:** IdUsuario, NombreUsuario
- **Algoritmo:** HS256

3.1.2 Endpoints de Nómina

> **POST** `/api/nomina/sincronizar`
> - Requiere: Authorization Bearer Token
> - Valida existencia de compañía
> - Publica evento a RabbitMQ
> - Retorna respuesta inmediata (asíncrono)

3.2 Mensajería Asíncrona (RabbitMQ)

3.2.1 Configuración del Exchange

Propiedad	Valor
Nombre	Nomina
Tipo	Direct
Durable	true

Auto-delete	false

3.3 Persistencia de Datos

Base de Datos IntegracionNubox

```
 CREATE TABLE Companias (
    Id uniqueidentifier PRIMARY KEY,
    Nombre nvarchar NOT NULL
);

CREATE TABLE BitacoraSincronizacion (
    Id uniqueidentifier PRIMARY KEY,
    CompaniaId uniqueidentifier FOREIGN KEY,
    FechaSincronizacion datetime2 NOT NULL,
    Detalles nvarchar NOT NULL
);
```

3.4 Servicios de Carga Masiva

> **Estrategia Inteligente:** El sistema selecciona automáticamente la mejor estrategia según el volumen de datos.

Estrategias de Bulk Insert

Volumen	Estrategia	BatchSize
> 10,000 registros	SqlBulkCopy	10,000
1,000 - 10,000	EF Core Optimizado	1,000
< 1,000	EF Core Normal	100

4. Flujo de Datos

4.1 Flujo de Sincronización Completo

Paso 1: Autenticación

```
Cliente → POST /api/auth/login → JWT Token (válido 24 horas)
```

Paso 2: Solicitud de Sincronización

```
1. Cliente envía POST /api/nomina/sincronizar (con Token)
2. SincronizarNominaHandler valida Compañía
3. Publica SincronizarNominaEvent a RabbitMQ
4. Retorna respuesta inmediata (202 Accepted)
5. Indica número de trabajadores procesados
```

Paso 3: Procesamiento Asíncrono

```
1. RabbitMQ encola mensaje
2. SincronizarNominaSubscriber consume
3. Selecciona estrategia de Bulk Insert:
   - >10,000: SqlBulkCopy
   - 1,000-10,000: EF Core Optimizado
   - <1,000: EF Core Normal
4. Inserta en BitacoraSincronizacion
5. ACK a RabbitMQ
```

5. Tecnologías Utilizadas

5.1 Stack Principal

Tecnología	Versión	Propósito
.NET	8.0	Framework base
ASP.NET Core	8.0	Web API
Entity Framework Core	8.0	ORM
SQL Server	2022	Base de datos
RabbitMQ	3.x	Message broker
Redis	7	Cache

5.2 Librerías NuGet

Paquete	Versión	Uso
EPPlus	8.1.1	Procesamiento Excel
MediatR	13.0.0	CQRS pattern
JWT Bearer	8.0.20	Autenticación
RabbitMQ.Client	7.1.2	Cliente RabbitMQ
Serilog	4.3.0	Logging

6. Guía de Instalación

6.1 Prerrequisitos

- Docker Desktop instalado
- Docker Compose
- .NET 8.0 SDK
- Git

6.2 Pasos de Instalación

Paso 1: Clonar el Repositorio

```
git clone <repository-url>
cd Integracion.Nubox
```

Paso 2: Levantar Infraestructura

```
cd Integracion.Nubox.Api
docker-compose up -d
```

> **Servicios levantados:**
> - SQL Server: localhost:1433
> - RabbitMQ: localhost:5672 (Management: 15672)
> - Redis: localhost:6379
> - Prometheus: localhost:9090
> - Grafana: localhost:3000

Paso 3: Ejecutar Migraciones

```
dotnet ef database update --context AuthContext
dotnet ef database update --context IntegracionNuboxContext
```

Paso 4: Ejecutar la API

```
dotnet run --project Integracion.Nubox.Api
```

> **Instalación Completa!**
> Acceder a Swagger: `https://localhost:7224/swagger`

7. API Endpoints

7.1 Autenticación

Login

POST `/api/auth/login`

Request Body:

```
{
  "username": "admin",
  "password": "hashed_password"
}
```

Response 200 OK:

```
{
  "status": true,
  "message": "OK",
  "data": {
    "token": "eyJhbGciOiJIUzI1NiIs...",
    "expires": "2025-09-26T12:00:00Z"
  }
}
```

7.2 Sincronizar Nómina

POST `/api/nomina/sincronizar`

Headers: Authorization: Bearer {token}

Request Body:

```
{
  "idCompania": "guid",
  "origen": 2,
  "trabajadores": [{
    "idTrabajador": "guid",
    "nombresTrabajador": "Juan",
    "apellidosTrabajador": "Pérez",
    "dniTrabajador": "12345678"
  }]
}
```

7.3 Códigos de Respuesta

Código	Descripción
200 OK	Solicitud exitosa
400 Bad Request	Error en datos
401 Unauthorized	Token inválido
404 Not Found	Recurso no existe
500 Internal Error	Error del servidor

8. Configuración

8.1 Configuración JWT

Parámetro	Valor
Algoritmo	HS256
Expiración	24 horas
Secret	Mínimo 32 caracteres

8.2 Optimizaciones de Rendimiento

Entity Framework Core

```
 // Deshabilitar tracking
_context.ChangeTracker.AutoDetectChangesEnabled = false;

// Limpiar después de batch
_context.ChangeTracker.Clear();
```

RabbitMQ

```
 RequestedConnectionTimeout = TimeSpan.FromSeconds(30)
NetworkRecoveryInterval = TimeSpan.FromSeconds(10)
AutomaticRecoveryEnabled = true
```

9. Monitoreo y Observabilidad

9.1 Prometheus

- **URL:** http://localhost:9090
- Métricas de API, RabbitMQ y SQL Server

9.2 Grafana

- **URL:** http://localhost:3000
- **Credenciales:** admin / admin123
- Dashboards preconfigurados

9.3 Logging con Serilog

Nivel	Uso
Information	Eventos normales
Warning	Situaciones anómalas
Error	Errores operacionales
Critical	Errores fatales

10. Principios de Diseño Implementados

10.1 Seguridad

- Autenticación JWT obligatoria
- Validación de compañías
- Passwords hasheadas
- Usuario Docker no-root
- CORS configurado

10.2 Escalabilidad

- Procesamiento asíncrono con RabbitMQ
- Estrategias adaptativas de bulk insert
- Batching de operaciones
- API Stateless
- Cache distribuido con Redis

10.3 Observabilidad

- Logging estructurado con Serilog
- Métricas con Prometheus
- Dashboards en Grafana

- Health checks

10.4 Trazabilidad

- BitacoraSincronizacion por operación
- Timestamps UTC
- Relación con Compañía
- Correlation IDs

10.5 Resiliencia

- Reintentos automáticos en RabbitMQ
- Manual ACK
- Transacciones en bulk operations
- Timeouts configurados

11. Patrones de Diseño

11.1 CQRS con MediatR

Separación de comandos (modifican estado) de consultas (solo lectura).

11.2 Repository Pattern

Abstracción del acceso a datos con repositorios genéricos y específicos.

11.3 Unit of Work

Gestión de transacciones y coordinación entre repositorios.

11.4 Publisher/Subscriber

Comunicación asíncrona mediante RabbitMQ.

11.5 Strategy Pattern

Selección dinámica de estrategia de bulk insert según volumen.

11.6 Dependency Injection

Inversión de control para todas las dependencias del sistema.

12. Estructura del Proyecto

```
Integracion.Nubox/
├── Integracion.Nubox.Api/
│   ├── Common/
│   │   ├── Entities/
│   │   ├── Persistence/
│   │   └── Services/
│   ├── Features/
│   │   ├── Auth/
│   │   │   ├── Endpoints/
│   │   │   ├── Handlers/
│   │   │   └── Requests/
│   │   └── Nomina/
│   │       ├── Endpoints/
│   │       ├── Events/
│   │       ├── Handlers/
│   │       ├── Publishers/
│   │       ├── Subscribers/
│   │       └── Requests/
│   ├── Infrastructure/
│   │   ├── Persistence/
│   │   │   ├── Contexts/
│   │   │   │   ├── Auth/
│   │   │   │   └── IntegracionNubox/
│   │   │   └── UnitOfWork.cs
│   │   └── Services/
│   ├── Extensions/
│   ├── docker-compose.yml
│   ├── Dockerfile
│   ├── Program.cs
│   └── appsettings.json
└── README.md
```

13. Comandos Útiles

13.1 Docker

```
 # Levantar servicios
docker-compose up -d

# Ver logs
docker-compose logs -f

# Detener servicios
docker-compose down

# Limpiar volúmenes
docker-compose down -v
```

13.2 Entity Framework

```
 # Crear migración
dotnet ef migrations add NombreMigracion --context AuthContext

# Aplicar migraciones
dotnet ef database update --context IntegracionNuboxContext

# Eliminar última migración
dotnet ef migrations remove --context AuthContext
```

13.3 RabbitMQ Management

```
 # Ver estado de queues (API REST)
curl -u admin:admin123 http://localhost:15672/api/queues
```

```
# Purgar queue
curl -u admin:admin123 -X DELETE \
  http://localhost:15672/api/queues/%2F/SincronizarNominaEvent/contents
```

14. Testing

14.1 Testing con cURL

Login

```
curl -X POST https://localhost:7224/api/auth/login \
  -H "Content-Type: application/json" \
  -d '{"username":"admin","password":"hashed_password"}'
```

Sincronizar Nómina

```
curl -X POST https://localhost:7224/api/nomina/sincronizar \
  -H "Authorization: Bearer {token}" \
  -H "Content-Type: application/json" \
  -d '{
    "idCompania":"3fa85f64-5717-4562-b3fc-2c963f66afa6",
    "origen":2,
    "trabajadores":[{
      "idTrabajador":"3fa85f64-5717-4562-b3fc-2c963f66afa7",
      "nombresTrabajador":"Juan",
      "apellidosTrabajador":"Pérez",
      "dniTrabajador":"12345678"
    }]
  }'
```

14.2 Testing con Swagger

1. Navegar a `https://localhost:7224/swagger`
2. Ejecutar `/api/auth/login` para obtener token
3. Hacer clic en "Authorize" y pegar el token
4. Probar endpoints protegidos

15. Solución de Problemas

15.1 Problemas Comunes

Error: "Cannot connect to RabbitMQ"

> **Causa:** RabbitMQ no está ejecutándose
> **Solución:**
>
> ```
> # Verificar estado
> docker ps | grep rabbitmq
>
> # Reiniciar
> docker-compose restart rabbitmq
> ```

Error: "JWT Token expired"

> **Causa:** Token expirado (>24 horas)
> **Solución:** Generar nuevo token mediante `/api/auth/login`

Error: "Database migration pending"

> **Causa:** Migraciones no aplicadas
> **Solución:**
>
> ```
> dotnet ef database update --context AuthContext
> dotnet ef database update --context IntegracionNuboxContext
> ```

Performance: Bulk insert lento

Causa: Estrategia incorrecta o índices faltantes

Soluciones:

- Verificar SqlBulkCopy para >10,000 registros
- Agregar índices en columnas frecuentes
- Aumentar BatchSize

16. Roadmap y Mejoras Futuras

16.1 Mejoras Planificadas

- **Autenticación OAuth 2.0:** Integración con proveedores externos
- **Dashboard de administración:** UI para gestión
- **Sistema de notificaciones:** Alertas por email/webhook
- **Métricas de negocio:** KPIs y reportes
- **Tests automatizados:** Cobertura >80%
- **Internacionalización:** Soporte multi-idioma

16.2 Optimizaciones Técnicas

- **Caché de segundo nivel:** Redis para consultas frecuentes
- **gRPC:** Comunicación interna de baja latencia
- **Event sourcing completo:** Historial inmutable
- **CQRS con bases separadas:** Optimización lectura/escritura
- **Kubernetes:** Orquestación avanzada

17. Conclusión

El **Sistema de Integración Nubox** proporciona una solución robusta, escalable y segura para sincronizar datos de asistencia entre el Sistema de Control de Asistencia y el Sistema de Cálculo de Liquidaciones.

17.1 Logros Principales

> ### Cumplimiento de Requerimientos
>
> - **Seguridad:** JWT, validaciones, usuario no-root
> - **Escalabilidad:** Mensajería asíncrona, bulk insert adaptativo
> - **Observabilidad:** Prometheus, Grafana, Serilog
> - **Trazabilidad:** BitacoraSincronizacion completa
> - **Resiliencia:** Reintentos, transacciones, recovery automático

17.2 Capacidades del Sistema

- Procesa **100,000 empresas** con diferentes periodicidades
- Soporta múltiples formatos: **APIs REST y Excel**
- Procesamiento asíncrono de alto rendimiento
- Monitoreo en tiempo real
- Arquitectura preparada para escalado horizontal

17.3 Métricas de Performance

Métrica	Objetivo

Latencia API	< 100ms
Throughput mensajería	> 1000 msg/min
Bulk insert < 1,000	< 1 segundo
Bulk insert 1,000-10,000	< 5 segundos
Bulk insert > 10,000	< 30 segundos
Uptime	99.9%

17.4 Contacto y Soporte

Para consultas técnicas, problemas o sugerencias sobre el sistema, contactar al equipo de desarrollo de Nubox.

> **Recursos Adicionales:**
>
> - Swagger UI: `https://localhost:7224/swagger`
> - RabbitMQ Management: `http://localhost:15672`
> - Grafana Dashboards: `http://localhost:3000`
> - Prometheus Metrics: `http://localhost:9090`

Sistema de Integración Nubox

Documentación Técnica - Versión 1.0

Septiembre 2025