

# Guía de Integración - SIESA ERP ↔ Kong WMS/RFID

## Arquitectura de las APIs

### Información de Conexión SIESA

#### Entorno de Pruebas (QA)

- **Base URL Consultas:** `https://integradorqa.siesacloud.com/api/consultas/`
- **Base URL Conectores:** `https://integradorqa.siesacloud.com/api/conectores/`
- **Protocolo:** REST API
- **Formato:** JSON
- **Método Consultas:** GET
- **Método Conectores:** POST

#### Autenticación SIESA

http

Headers Requeridos:

ConniKey: [tu\_conni\_key]

ConniToken: [tu\_conni\_token]

Content-Type: application/json

**Nota Importante:** Los tokens ConniKey y ConniToken son específicos de cada implementación y se obtienen al configurar la integración en el Módulo de Conectividad de SIESA.

#### Parámetros Comunes

- **IdCompañía:** Identificador de la compañía (ej: "1" para prospecto)

- **IdDocumento:** ID del documento/entidad a consultar o insertar
- **IdInterface:** Requerido SOLO para conectores (inserciones)

## Información de Conexión Kong RFID

### Entorno de Staging

- **Base URL:** <https://api-apes-stock-rfid-staging.technoapes.io/>
- **Protocolo:** REST API (HTTPS)
- **Formato:** JSON
- **Documentación:** OpenAPI/Swagger disponible en </api/docs/>

### Autenticación Kong

http

POST /auth/token/login/

**Content-Type:** application/json

```
{  
  "username": "usuario",  
  "password": "contraseña"  
}
```

**Response:**

```
{  
  "auth_token": "your_token_here"  
}
```

Headers para requests:

**Authorization:** Token {auth\_token}

**Content-Type:** application/json

---

## Patrones de Integración

### Patrón 1: Sincronización de Maestros (SIESA → Kong)

#### Consulta en SIESA

**Endpoint:** `GET /api/consultas/{nombre_consulta}`

#### Parámetros de Consulta:

Parámetro	Tipo	Obligatorio	Descripción	Ejemplo
IdCompañía	string	Sí	Código de compañía	"1"
descripcion	string	No	Filtro por descripción	"Laptop"
pagina	integer	No	Número de página	1
registrosPorPagina	integer	No	Tamaño de página	100

#### Ejemplo de Solicitud:

```
bash

curl -X GET "https://integradorqa.siesacloud.com/api/consultas/items?IdCompañía=1&pagina=1&registrosPorPagina=100" \
-H "ConniKey: {tu_key}" \
-H "ConniToken: {tu_token}"
```

#### Respuesta Exitosa (200):

```
json
```

```
{
  "success": true,
  "data": [
    {
      "f120_id_item": 12345,
      "f120_referencia": "PROD-001",
      "f120_descripcion": "Laptop Dell XPS 15",
      "f120_id_grupo": 10,
      "f120_codigo_barras": "7501234567890",
      "f120_unidad_medida": "UN",
      "f120_ind_estado": 1
    }
  ],
  "pagination": {
    "total": 250,
    "pagina_actual": 1,
    "total_paginas": 3
  }
}
```

## Inserción en Kong

**Endpoint:** POST /inventory/skus/

**Payload:**

json

```
{
  "external_id": "PROD-001",
  "group_external_id": "10",
  "name": "Laptop Dell XPS 15",
  "display_name": "Laptop Dell XPS 15",
  "ean": "7501234567890",
  "is_active": true,
  "properties": {
    "unidad_medida": "UN",
    "siesa_id": "12345"
  }
}
```

### Ejemplo de Solicitud:

bash

```
curl -X POST "https://api-apes-stock-rfid-staging.technoapes.io/inventory/skus/" \
-H "Authorization: Token {your_token}" \
-H "Content-Type: application/json" \
-d '{
  "external_id": "PROD-001",
  "name": "Laptop Dell XPS 15",
  "group_external_id": "10",
  "ean": "7501234567890",
  "is_active": true
}'
```

### Respuesta Exitosa (201):

json

```
{
  "id": 456,
  "external_id": "PROD-001",
  "name": "Laptop Dell XPS 15",
  "display_name": "Laptop Dell XPS 15",
  "group_external_id": "10",
  "ean": "7501234567890",
  "is_active": true,
  "created_at": "2025-10-01T10:30:00Z",
  "updated_at": "2025-10-01T10:30:00Z"
}
```

### Errores Comunes:

- **400**: Validación fallida (external\_id duplicado, campos obligatorios faltantes)
- **401**: Token inválido o expirado
- **404**: Grupo no encontrado

---

## Patrón 2: Envío de Movimientos (Kong → SIESA)

### Preparar Documento en Kong

**Evento Trigger:** Move cerrado en Kong

### Consultar Detalle del Move:

```
bash

curl -X GET "https://api-apes-stock-rfid-staging.technoapes.io/inventory/moves/789/" \
-H "Authorization: Token {your_token}"
```

### Respuesta:

```
json
{
  "id": 789,
  "move_type": "RECEIVING",
  "destination_location": {
    "id": 10,
    "external_id": "BOD01",
    "name": "Bodega Principal"
  },
  "status": "CLOSED",
  "closed_at": "2025-10-01T11:45:00Z",
  "reference": "SIESA-OC-12345"
}
```

### Consultar Líneas del Move:

```
bash

curl -X GET "https://api-apes-stock-rfid-staging.technoapes.io/inventory/moves/789/lines/" \
-H "Authorization: Token {your_token}"
```

### Respuesta:

```
json
```

```
[
  {
    "id": 1001,
    "sku": {
      "external_id": "PROD-001",
      "name": "Laptop Dell XPS 15"
    },
    "quantity_expected": 50,
    "quantity_received": 50,
    "quantity_difference": 0
  },
  {
    "id": 1002,
    "sku": {
      "external_id": "PROD-002",
      "name": "Mouse Logitech"
    },
    "quantity_expected": 100,
    "quantity_received": 98,
    "quantity_difference": -2
  }
]
```

## Transformar a Formato SIESA

javascript



*// Transformación Kong Move → SIESA Document*

```
const transformarMoveASIESA = (move, lines) => {
  return {
    Inicial: [{ F_CIA: "1" }],
    Documentos: [{
      F_CIA: "1",
      f350_id_co: "1",
      f350_id_tipo_docto: move.move_type === "RECEIVING" ? "ENT" : "SAL",
      f350_consec_docto: "AUTO",
      f350_fecha: move.closed_at.split("T")[0],
      f350_ind_estado: "2",
      f350_notas: `Movimiento Kong #${move.id} - ${move.reference}`,
      f450_id_concepto: "1",
      f450_id_bodega_entrada: move.destination_location.external_id,
      f450_docto_alterno: `KONG-MOVE-${move.id}`
    }],
    Movimientos: lines.map((line, index) => ({
      F_CIA: "1",
      f470_id_item: line.sku.external_id,
      f470_id_bodega: move.destination_location.external_id,
      f470_id_unidad_medida: "UN",
      f470_cant_base: String(line.quantity_received),
      f470_nro_registro: String(index + 1),
      f470_notas: `Línea ${index + 1} - Esperado: ${line.quantity_expected}, Recibido: ${line.quantity_received}`
    })),
    Final: [{ F_CIA: "1" }]
  };
};
```

**Enviar a SIESA**

**Endpoint:** `POST /api/conectores/documentoinv`

## Headers:

http

ConniKey: {tu\_key}

ConniToken: {tu\_token}

Content-Type: application/json

## Parámetros URL:

- IdCompañia=1
- IdDocumento=142951
- IdInterface={id\_interface} ← Obtenido al iniciar integración

## Ejemplo de Solicitud Completa:

bash

```
curl -X POST "https://integradorqa.siesacloud.com/api/conectores/documentoinv?IdCompañia=1&IdDocumento=1429" \
-H "ConniKey: {tu_key}" \
-H "ConniToken: {tu_token}" \
-H "Content-Type: application/json" \
-d '{
  "Inicial": [{"F_CIA": "1"}],
  "Documentos": [{
    "F_CIA": "1",
    "f350_id_co": "1",
    "f350_id_tipo_docto": "ENT",
    "f350_consec_docto": "AUTO",
    "f350_fecha": "2025-10-01",
    "f350_ind_estado": "2",
    "f350_notas": "Recepción desde Kong",
    "f450_id_concepto": "1",
    "f450_id_bodega_entrada": "BOD01",
    "f450_docto_alterno": "KONG-MOVE-789"
  }],
  "Movimientos": [{
    "F_CIA": "1",
    "f470_id_item": "PROD-001",
    "f470_id_bodega": "BOD01",
    "f470_id_unidad_medida": "UN",
    "f470_cant_base": "50",
    "f470_nro_registro": "1"
  }],
  "Final": [{"F_CIA": "1"}]
}'
```

**Respuesta Exitosa (200):**

```
json
{
  "success": true,
  "message": "Documento creado exitosamente",
  "data": {
    "consecutivo": "ENT-00012345",
    "id_documento": 98765
  }
}
```

### Respuesta de Error (400):

```
json
{
  "success": false,
  "message": "Error de validación",
  "errors": [
    {
      "campo": "f470_id_item",
      "linea": 1,
      "error": "El item PROD-999 no existe en la compañía"
    }
  ]
}
```

---

## Casos de Uso Detallados

### Caso 1: Sincronización de Catálogo de Productos

#### Objetivo

Mantener el catálogo de productos sincronizado entre SIESA y Kong

### **Frecuencia Recomendada**

Cada 4 horas o ante cambios

### **Flujo Completo**

```
javascript
```

*// 1. Consultar productos modificados desde última sincronización*

```
const ultimaSync = getUltimaFechaSincronizacion();

const productos = await fetch(
  `https://integradorqa.siesacloud.com/api/consultas/items?IdCompañia=1&fechaModificacion=${ultimaSync}`,
  {
    headers: {
      'ConniKey': CONNI_KEY,
      'ConniToken': CONNI_TOKEN
    }
  }
).then(r => r.json());
```

*// 2. Transformar cada producto*

```
for (const prod of productos.data) {
  const skuKong = {
    external_id: prod.fl20_referencia,
    group_external_id: String(prod.fl20_id_grupo),
    name: prod.fl20_descripcion,
    display_name: prod.fl20_descripcion_comercial || prod.fl20_descripcion,
    ean: prod.fl20_codigo_barras || '',
    is_active: prod.fl20_ind_estado === 1,
    properties: {
      unidad_medida: prod.fl20_unidad_medida,
      siesa_id: prod.fl20_id_item
    }
  };
}
```

*// 3. Verificar si existe en Kong*

```
const existeEnKong = await fetch(
  `https://api-apes-stock-rfid-staging.technoapes.io/inventory/skus/?external_id=${prod.fl20_referencia}`,
  {
    headers: { 'Authorization': `Token ${KONG_TOKEN}` }
  }
)
```

```
}  
)}.then(r => r.json());  
  
// 4. Crear o actualizar  
if(existeEnKong.results && existeEnKong.results.length > 0) {  
  // Actualizar  
  await fetch(  
    `https://api-apes-stock-rfid-staging.technoapes.io/inventory/skus/${existeEnKong.results[0].id}`,  
    {  
      method: 'PATCH',  
      headers: {  
        'Authorization': `Token ${KONG_TOKEN}`,  
        'Content-Type': 'application/json'  
      },  
      body: JSON.stringify(skuKong)  
    }  
  );  
  console.log(`SKU actualizado: ${prod.f120_referencia}`);  
} else {  
  // Crear  
  await fetch(  
    'https://api-apes-stock-rfid-staging.technoapes.io/inventory/skus/',  
    {  
      method: 'POST',  
      headers: {  
        'Authorization': `Token ${KONG_TOKEN}`,  
        'Content-Type': 'application/json'  
      },  
      body: JSON.stringify(skuKong)  
    }  
  );  
  console.log(`SKU creado: ${prod.f120_referencia}`);  
}
```

```
}
```

```
// 5. Registrar sincronización
```

```
setUltimaFechaSincronizacion(new Date());
```

## Manejo de Errores

```
javascript
```



```

// Wrapper con reintentos
const sincronizarProducto = async (producto, reintentos = 3) => {
  for (let i = 0; i < reintentos; i++) {
    try {
      // Lógica de sincronización
      return await crearOActualizarSKU(producto);
    } catch (error) {
      if (i === reintentos - 1) {
        // Log del error final
        console.error(`Error sincronizando ${producto.fl20_referencia}`, error);
        // Registrar en tabla de errores para reproceso
        await registrarErrorSincronizacion({
          tipo: 'producto',
          external_id: producto.fl20_referencia,
          error: error.message,
          timestamp: new Date()
        });
        throw error;
      }
      // Esperar antes de reintentar
      await sleep(Math.pow(2, i) * 1000);
    }
  }
};

```

## Caso 2: Recepción de Mercancía

### Objetivo

Registrar en SIESA las recepciones físicas confirmadas en Kong WMS

## Trigger

Evento: Move cerrado con tipo "RECEIVING" en Kong

## Flujo Paso a Paso

### Paso 1: Webhook o Polling en Kong

```
javascript

// Opción A: Webhook (recomendado)
// Configurar en Kong para enviar notificación cuando un move se cierra

// Opción B: Polling
const verificarMovimientosPendientes = async () => {
  const ultimaConsulta = getUltimaConsultaMovimientos();

  const moves = await fetch(
    `https://api-apes-stock-rfid-staging.technoapes.io/inventory/moves/?status=CLOSED&closed_at__gte=${ultimaCor
    {
      headers: { 'Authorization': `Token ${KONG_TOKEN}` }
    }
  ).then(r => r.json());

  return moves.results.filter(m => m.move_type === 'RECEIVING');
};
```

### Paso 2: Obtener Detalle Completo

```
javascript
```

```
const obtenerDetalleRecepcion = async (moveId) => {  
  // Obtener cabecera del move  
  const move = await fetch(  
    `https://api-apes-stock-rfid-staging.technoapes.io/inventory/moves/${moveId}/`,  
    {  
      headers: { 'Authorization': `Token ${KONG_TOKEN}` }  
    }  
  ).then(r => r.json());  
  
  // Obtener líneas  
  const lines = await fetch(  
    `https://api-apes-stock-rfid-staging.technoapes.io/inventory/moves/${moveId}/lines/`,  
    {  
      headers: { 'Authorization': `Token ${KONG_TOKEN}` }  
    }  
  ).then(r => r.json());  
  
  return { move, lines };  
};
```

### Paso 3: Transformar a Documento SIESA

javascript

```
const generarEntradaSIESA = (move, lines) => {  
  // Validar que haya líneas  
  if (!lines || lines.length === 0) {  
    throw new Error('No hay líneas para procesar');  
  }  
  
  // Validar que todas las cantidades sean positivas  
  const lineasInvalidas = lines.filter(l => l.quantity_received <= 0);  
  if (lineasInvalidas.length > 0) {  
    throw new Error(`Líneas con cantidad 0 o negativa: ${lineasInvalidas.map(l => l.id).join(', ')}`);  
  }  
  
  return {  
    Inicial: [{ F_CIA: "1" }],  
    Documentos: [{  
      F_CIA: "1",  
      f350_id_co: "1",  
      f350_id_tipo_docto: "ENT",  
      f350_consec_docto: "AUTO",  
      f350_fecha: new Date(move.closed_at).toISOString().split('T')[0],  
      f350_ind_estado: "2", // Confirmado  
      f350_notas: `Recepción Kong Move #${move.id} - ${move.reference || ""}`,  
      f450_id_concepto: "1",  
      f450_id_bodega_entrada: move.destination_location.external_id,  
      f450_docto_alterno: `KONG-RCV-${move.id}`  
    }],  
    Movimientos: lines.map((line, index) => ({  
      F_CIA: "1",  
      f470_id_item: line.sku.external_id,  
      f470_id_bodega: move.destination_location.external_id,  
      f470_id_unidad_medida: "UN",  
      f470_cant_base: String(line.quantity_received),  
      f470_nro_registro: String(index + 1),
```

```
f470_notas: `SKU: ${line.sku.name} - Esperado: ${line.quantity_expected}, Recibido: ${line.quantity_received}``),
Final: [{ F_CIA: "1" }]
};
};
```

#### Paso 4: Enviar a SIESA

```
javascript

const enviarEntradaASIESA = async (documento) => {
  const response = await fetch(
    `https://integradorqa.siesacloud.com/api/conectores/documentoinv?IdCompañia=1&IdDocumento=142951&IdInter
    {
      method: 'POST',
      headers: {
        'ConniKey': CONNI_KEY,
        'ConniToken': CONNI_TOKEN,
        'Content-Type': 'application/json'
      },
      body: JSON.stringify(documento)
    }
  );

  if (!response.ok) {
    const error = await response.json();
    throw new Error(`Error SIESA: ${error.message || JSON.stringify(error)}`);
  }

  return await response.json();
};
```

## Paso 5: Actualizar Estado en Kong

javascript

*// Opcional: Marcar el move como procesado en SIESA*

```
const actualizarMoveConDocumentoSIESA = async (moveId, documentoSIESA) => {  
  await fetch(  
    `https://api-apes-stock-rfid-staging.technoapes.io/inventory/moves/${moveId}/`,  
    {  
      method: 'PATCH',  
      headers: {  
        'Authorization': `Token ${KONG_TOKEN}`,  
        'Content-Type': 'application/json'  
      },  
      body: JSON.stringify({  
        reference: `SIESA-${documentoSIESA.consecutivo}`  
      })  
    }  
  );  
};
```

## Flujo Completo Integrado

javascript

```
const procesarRecepcionKongASIESA = async (moveId) => {
  try {
    console.log(`Procesando Move #${moveId}...`);

    // 1. Obtener datos
    const { move, lines } = await obtenerDetalleRecepcion(moveId);

    // 2. Validaciones
    if (move.status !== 'CLOSED') {
      throw new Error('El move no está cerrado');
    }

    // 3. Generar documento
    const documento = generarEntradaSIESA(move, lines);

    // 4. Enviar a SIESA
    const resultado = await enviarEntradaASIESA(documento);

    // 5. Actualizar Kong
    await actualizarMoveConDocumentoSIESA(moveId, resultado.data);

    // 6. Log exitoso
    console.log(`✓ Move #${moveId} procesado. Documento SIESA: ${resultado.data.consecutivo}`);

    return {
      success: true,
      kong_move_id: moveId,
      siesa_documento: resultado.data.consecutivo
    };
  } catch (error) {
    console.error(`✗ Error procesando Move #${moveId}:`, error.message);
  }
}
```

```
// Registrar error para reproceso  
await registrarErrorIntegracion({  
  tipo: 'recepcion',  
  kong_move_id: moveId,  
  error: error.message,  
  timestamp: new Date()  
});  
  
throw error;  
}  
};
```

---

### Caso 3: Despacho de Pedidos

#### Objetivo

Registrar en SIESA los despachos confirmados desde Kong WMS

#### Trigger

Evento: Packing completado o Shipment cerrado en Kong

#### Flujo

##### Paso 1: Detectar Despacho Completado

```
javascript
```



```
const verificarDespachosCompletados = async () => {  
  // Opción 1: Desde Packings  
  const packings = await fetch(  
    'https://api-apes-stock-rfid-staging.technoapes.io/operations/packings/?status=COMPLETED',  
    {  
      headers: { 'Authorization': `Token ${KONG_TOKEN}` }  
    }  
  ).then(r => r.json());  
  
  // Opción 2: Desde Shipments  
  const shipments = await fetch(  
    'https://api-apes-stock-rfid-staging.technoapes.io/stock/shipments/?status=SHIPPED',  
    {  
      headers: { 'Authorization': `Token ${KONG_TOKEN}` }  
    }  
  ).then(r => r.json());  
  
  return packings.results; // o shipments.results según el flujo  
};
```

## Paso 2: Obtener Detalle del Packing

javascript

```
const obtenerDetallePacking = async (packingId) => {  
  const packing = await fetch(  
    `https://api-apes-stock-rfid-staging.technoapes.io/operations/packings/${packingId}/`,  
    {  
      headers: { 'Authorization': `Token ${KONG_TOKEN}` }  
    }  
  ).then(r => r.json());  
  
  const lines = await fetch(  
    `https://api-apes-stock-rfid-staging.technoapes.io/operations/packings/${packingId}/lines/`,  
    {  
      headers: { 'Authorization': `Token ${KONG_TOKEN}` }  
    }  
  ).then(r => r.json());  
  
  return { packing, lines };  
};
```

### Paso 3: Generar Salida en SIESA

javascript

```
const generarSalidaSIESA = (packing, lines) => {
  return {
    Inicial: [{ F_CIA: "1" }],
    Documentos: [{
      F_CIA: "1",
      f350_id_co: "1",
      f350_id_tipo_docto: "SAL",
      f350_consec_docto: "AUTO",
      f350_fecha: new Date().toISOString().split('T')[0],
      f350_id_tercero: packing.store_order?.requester_external_id || "",
      f350_ind_estado: "2",
      f350_notas: `Despacho Kong Packing #${packing.id}`,
      f450_id_concepto: "2",
      f450_id_bodega_salida: packing.location?.external_id || "BOD01",
      f450_docto_alterno: `KONG-SHIP-${packing.id}`
    }],
    Movimientos: lines.map((line, index) => ({
      F_CIA: "1",
      f470_id_item: line.store_order_line.sku.external_id,
      f470_id_bodega: packing.location?.external_id || "BOD01",
      f470_id_unidad_medida: "UN",
      f470_cant_base: String(line.quantity),
      f470_nro_registro: String(index + 1),
      f470_notas: `Packing confirmado RFID - SKU: ${line.store_order_line.sku.name}`
    })),
    Final: [{ F_CIA: "1" }]
  };
};
```

## **Caso 4: Ajuste de Inventario por Conteo RFID**

### **Objetivo**

Cuadrar diferencias entre saldo físico (Kong) y saldo contable (SIESA)

### **Frecuencia**

Después de cada auditoría/conteo cíclico

### **Flujo Completo**

#### **Paso 1: Completar Auditoría en Kong**

```
javascript
```

*// 1. Crear auditoría*

```
const audit = await fetch(
  'https://api-apes-stock-rfid-staging.technoapes.io/inventory/audits/',
  {
    method: 'POST',
    headers: {
      'Authorization': `Token ${KONG_TOKEN}`,
      'Content-Type': 'application/json'
    },
    body: JSON.stringify({
      location: 10, // ID de la ubicación
      audit_type: 'CYCLIC'
    })
  }
).then(r => r.json());
```

*// 2. Realizar conteo físico con RFID*

*// (Este paso se hace desde el dispositivo RFID)*

*// 3. Calcular diferencias*

```
await fetch(
  `https://api-apes-stock-rfid-staging.technoapes.io/inventory/audits/${audit.id}/audit_calculate/`,
  {
    method: 'POST',
    headers: {
      'Authorization': `Token ${KONG_TOKEN}`,
      'Content-Type': 'application/json'
    }
  }
);
```

*// 4. Obtener líneas con diferencias*

```
const auditLines = await fetch(
```

```
`https://api-apes-stock-rfid-staging.technoapes.io/inventory/audit-lines/?audit=${audit.id}&difference__ne=0`,  
{  
  headers: { 'Authorization': `Token ${KONG_TOKEN}` }  
}  
).then(r => r.json());
```

## Paso 2: Para cada SKU con diferencia

javascript

```

const procesarDiferencias = async (auditLines) => {
  for (const line of auditLines.results) {
    // Consultar saldo actual en SIESA
    const saldoSIESA = await consultarSaldoSIESA({
      item: line.sku.external_id,
      bodega: line.location.external_id
    });

    const saldoFisico = line.physical_quantity;
    const saldoContable = saldoSIESA.saldo_cantidad;
    const diferencia = saldoFisico - saldoContable;

    if (diferencia !== 0) {
      // Generar ajuste en SIESA
      await generarAjusteSIESA({
        item: line.sku.external_id,
        bodega: line.location.external_id,
        cantidad: Math.abs(diferencia),
        tipo: diferencia > 0 ? 'positivo' : 'negativo',
        notas: `Auditoría #${audit.id} - Físico: ${saldoFisico}, Contable: ${saldoContable}`
      });
    }
  }
};

```

### Paso 3: Generar Documento de Ajuste

javascript

```
const generarAjusteSIESA = async ({ item, bodega, cantidad, tipo, notas }) => {  
  const esPositivo = tipo === 'positivo';  
  
  const documento = {  
    Inicial: [{ F_CIA: "1" }],  
    Documentos: [{  
      F_CIA: "1",  
      f350_id_co: "1",  
      f350_id_tipo_docto: "AJU",  
      f350_consec_docto: "AUTO",  
      f350_fecha: new Date().toISOString().split("T")[0],  
      f350_ind_estado: "2",  
      f350_notas: `Ajuste por conteo RFID - ${notas}`,  
      f450_id_concepto: esPositivo ? "3" : "4",  
      f450_id_bodega_entrada: esPositivo ? bodega : "",  
      f450_id_bodega_salida: esPositivo ? "" : bodega,  
      f450_docto_alterno: `KONG-ADJ-${Date.now()}`  
    }],  
    Movimientos: [{  
      F_CIA: "1",  
      f470_id_item: item,  
      f470_id_bodega: bodega,  
      f470_id_unidad_medida: "UN",  
      f470_cant_base: String(cantidad),  
      f470_nro_registro: "1",  
      f470_notas: notas  
    }],  
    Final: [{ F_CIA: "1" }]  
  };  
  
  return await enviarASIESA(documento);  
};
```



---

## Validaciones Pre-envío

### Validador de Documentos

javascript

```
class ValidadorDocumentoSIESA {
  static validar(documento) {
    const errores = [];

    // Validar Inicial
    if (!documento.Inicial || documento.Inicial.length === 0) {
      errores.push('Sección Inicial es obligatoria');
    }

    // Validar Documentos
    if (!documento.Documentos || documento.Documentos.length === 0) {
      errores.push('Sección Documentos es obligatoria');
    } else {
      const doc = documento.Documentos[0];

      // Campos obligatorios
      const camposObligatorios = [
        'F_CIA', 'f350_id_co', 'f350_id_tipo_docto',
        'f350_consec_docto', 'f350_fecha', 'f350_ind_estado',
        'f450_id_concepto'
      ];

      camposObligatorios.forEach(campo => {
        if (!doc[campo]) {
          errores.push(`Campo obligatorio faltante: ${campo}`);
        }
      });

      // Validar bodegas según tipo
      if (doc.f350_id_tipo_docto === 'ENT' && !doc.f450_id_bodega_entrada) {
        errores.push('Entrada requiere f450_id_bodega_entrada');
      }

      if (doc.f350_id_tipo_docto === 'SAL' && !doc.f450_id_bodega_salida) {
```

```

    errores.push('Salida requiere f450_id_bodega_salida');
  }
  if (doc.f350_id_tipo_docto === 'TRA' &&
    (!doc.f450_id_bodega_entrada || !doc.f450_id_bodega_salida)) {
    errores.push('Transferencia requiere ambas bodegas');
  }
}

// Validar Movimientos
if (!documento.Movimientos || documento.Movimientos.length === 0) {
  errores.push('Debe haber al menos un movimiento');
} else {
  documento.Movimientos.forEach((mov, idx) => {
    if (!mov.f470_id_item) {
      errores.push(`Línea ${idx + 1}: f470_id_item es obligatorio`);
    }
    if (!mov.f470_cant_base || parseFloat(mov.f470_cant_base) <= 0) {
      errores.push(`Línea ${idx + 1}: Cantidad debe ser mayor a 0`);
    }
    if (!mov.f470_id_unidad_medida) {
      errores.push(`Línea ${idx + 1}: Unidad de medida obligatoria`);
    }
  });
}

// Validar Final
if (!documento.Final || documento.Final.length === 0) {
  errores.push('Sección Final es obligatoria');
}

return {
  valido: errores.length === 0,
  errores

```

```
    };\n  }\n}\n\n// Uso\nconst validacion = ValidadorDocumentoSIESA.validar(documento);\nif (!validacion.valido) {\n  console.error('Errores de validación:', validacion.errores);\n  throw new Error(`Documento inválido: ${validacion.errores.join(', ')}`);\n}
```

---

## Manejo de Rate Limits

javascript

```
class RateLimiter {
  constructor(requestsPerMinute = 60) {
    this.requestsPerMinute = requestsPerMinute;
    this.requests = [];
  }

  async esperar() {
    const ahora = Date.now();
    const unMinutoAtras = ahora - 60000;

    // Limpiar requests antiguos
    this.requests = this.requests.filter(t => t > unMinutoAtras);

    if (this.requests.length >= this.requestsPerMinute) {
      const tiempoEspera = this.requests[0] - unMinutoAtras;
      await sleep(tiempoEspera);
    }

    this.requests.push(ahora);
  }
}

const limiter = new RateLimiter(60); // 60 requests por minuto

// Uso
await limiter.esperar();
const response = await fetch(url, options);
```

---

## Logs y Monitoreo

```
javascript
```

```
const logIntegracion = {  
  timestamp: new Date().toISOString(),  
  flujo: 'Kong → SIESA',  
  tipo: 'Entrada de Inventario',  
  kong_move_id: 789,  
  siesa_documento: 'ENT-12345',  
  items_procesados: 5,  
  status: 'SUCCESS',  
  duracion_ms: 1250,  
  detalles: {  
    bodega: 'BOD01',  
    total_unidades: 150  
  }  
};
```

*// Guardar en base de datos o servicio de logs*

```
await guardarLog(logIntegracion);
```