# **Diseño del Experimento: Validación del Sistema de Votación - Patrón Reliable Messaging**

**Hecho por:** Juan Camilo Amorocho, Juan Esteban Ruiz, Tomas Quintero  
 **Fecha:** Junio 2025  
 **Contexto:** Implementación de Patrones de Diseño para Sistema Electoral

## **1. Introducción**

Este documento describe la metodología y los procedimientos para validar que el sistema de votación desarrollado para la empresa XYZ cumple con los requisitos críticos de **confiabilidad** y **unicidad** en el conteo de votos, implementando el patrón **Reliable Messaging** para garantizar la transmisión segura de votos desde las estaciones hacia el servidor central.

### **Requisitos Críticos del Sistema:**

* **Confiabilidad:** Garantizar que el 100% de los votos emitidos sean registrados correctamente
* **Unicidad:** Asegurar que ningún voto sea contado más de una vez
* **Disponibilidad:** Mantener operación continua ante fallos de red y errores de comunicación
* **Trazabilidad:** Seguimiento completo de cada voto mediante UUIDs únicos y timestamps precisos

## **2. Objetivos de Validación**

### **2.1 Objetivos Primarios**

1. **Verificar integridad de transmisión:** Comprobar que todos los votos emitidos desde las estaciones lleguen al servidor central sin pérdida de datos
2. **Validar unicidad absoluta:** Confirmar que el sistema previene efectivamente el doble conteo de votos, incluso bajo condiciones de fallo
3. **Evaluar resistencia ante fallos:** Probar la capacidad del sistema para recuperarse de interrupciones de red y errores de comunicación
4. **Medir rendimiento bajo carga:** Validar el comportamiento del sistema bajo diferentes volúmenes de carga de trabajo
5. **Verificar trazabilidad completa:** Confirmar que cada voto puede ser rastreado desde su origen hasta su procesamiento final

### **2.2 Patrón Reliable Messaging - Componentes a Validar**

El sistema implementa el patrón Reliable Messaging a través de los siguientes componentes:

* **GestorEnvioVotosImpl:** Manejo de envío confiable desde estaciones
* **AlmacenamientoTransitorioImpl:** Persistencia local temporal en estaciones
* **GestorRecepcionVotosImpl:** Recepción y procesamiento en servidor central
* **ValidadorDeVotosImpl:** Prevención de duplicados mediante registro de votos procesados
* **AlmacenamientoVotosImpl:** Persistencia definitiva en servidor central

## **3. Escenarios de Prueba**

### **3.1 Escenario 1: Transmisión Normal - Validación de Confiabilidad Básica**

**Objetivo:** Verificar funcionamiento correcto bajo condiciones ideales

**Configuración:**

* Ambiente de red estable
* 3 estaciones de votación activas
* 1000 votos por estación (total: 3000 votos)
* 4 candidatos disponibles

**Procedimiento:**

1. Inicializar servidor central (CentroVotacionApp)
2. Activar estaciones de votación (EstacionVotacionApp)
3. Ejecutar votación masiva con distribución aleatoria
4. Verificar consistencia entre almacenamiento transitorio y central
5. Validar conteo final y trazabilidad de votos

**Métricas Esperadas:**

* Tasa de entrega: 100%
* Tiempo promedio de confirmación: < 3 segundos
* Pérdida de datos: 0 votos
* Consistencia entre estaciones y centro: 100%

**Archivos de Validación:**

* VotosTransitorios.csv (por cada estación)
* VotosRecibidos.csv (servidor central)
* Logs de aplicación con timestamps

### **3.2 Escenario 2: Pérdida de Conexión Temporal - Validación de Recuperación**

**Objetivo:** Probar mecanismos de recuperación ante fallos de red

**Configuración:**

* Transmisión de 10,000 votos en progreso
* Interrupción programada de conectividad por 30 segundos
* Restauración automática de conexión

**Procedimiento:**

1. Iniciar transmisión masiva de votos
2. Simular desconexión de red en momento crítico
3. Verificar almacenamiento local durante interrupción
4. Restaurar conectividad
5. Confirmar reenvío automático de votos pendientes
6. Validar integridad total post-recuperación

**Componentes Críticos Evaluados:**

* **Scheduler de reintentos:** Verificación cada 30 segundos
* **Almacenamiento transitorio:** Persistencia durante desconexión
* **Mecanismo de callback:** Confirmaciones bidireccionales

**Métricas Esperadas:**

* Tiempo de detección de fallo: ≤ 30 segundos
* Tiempo de recuperación completa: ≤ 2 minutos
* Pérdida de datos durante fallo: 0 votos
* Reenvío exitoso post-reconexión: 100%

### **3.3 Escenario 3: Prevención de Duplicados - Validación de Unicidad**

**Objetivo:** Confirmar efectividad de mecanismos anti-duplicación

**Configuración:**

* Simulación de reintentos de transmisión
* Forzar condiciones de duplicación
* Validar detección y prevención

**Procedimiento:**

1. Enviar voto con servidor central activo
2. Apagar servidor central temporalmente
3. Intentar reenvío del mismo voto desde estación
4. Reactivar servidor central
5. Verificar que voto se procesa una sola vez
6. Confirmar registro único en base de datos central

**Componentes Evaluados:**

* **ValidadorDeVotosImpl:** Verificación de unicidad por UUID
* **Control local de sesión:** Prevención de doble votación por cédula
* **Sistema de callbacks:** Confirmaciones de estado

**Métricas Esperadas:**

* Detección de duplicados: 100%
* Falsos positivos: 0%
* Tiempo de validación: < 1 segundo
* Consistencia de registros: 100%

### **3.4 Escenario 4: Carga Máxima - Validación de Escalabilidad**

**Objetivo:** Probar rendimiento bajo condiciones de estrés

**Configuración:**

* 4 estaciones de votación simultáneas
* 25,000 votos por estación (total: 100,000 votos)
* Transmisión concurrente

**Procedimiento:**

1. Configurar múltiples estaciones en paralelo
2. Ejecutar carga masiva de votos simultáneamente
3. Monitorear utilización de recursos
4. Verificar tiempos de respuesta
5. Confirmar integridad bajo alta carga
6. Validar conteo final consolidado

**Métricas Esperadas:**

* Throughput mínimo: 1000 votos/minuto por estación
* Tiempo de respuesta máximo: < 5 segundos
* Utilización CPU: < 80%
* Pérdida de votos bajo carga: 0%
* Consistencia final: 100%

## **4. Métricas de Evaluación**

### **4.1 Métricas de Confiabilidad**

| **Métrica** | **Objetivo** | **Método de Medición** |
| --- | --- | --- |
| **Tasa de Entrega** | 100% | Comparación entre votos emitidos vs. votos recibidos |
| **Tiempo de Confirmación** | < 3 segundos | Medición timestamp emisión vs. confirmación |
| **Recuperación Post-Fallo** | < 2 minutos | Tiempo desde reconexión hasta procesamiento completo |
| **Consistencia de Datos** | 100% | Verificación cruzada entre almacenamientos local y central |

### **4.2 Métricas de Unicidad**

| **Métrica** | **Objetivo** | **Método de Medición** |
| --- | --- | --- |
| **Detección de Duplicados** | 100% | Intentos de duplicación vs. detecciones exitosas |
| **Falsos Positivos** | 0% | Votos legítimos rechazados incorrectamente |
| **Tiempo de Validación** | < 1 segundo | Latencia del proceso de verificación de unicidad |
| **Integridad UUID** | 100% | Verificación de unicidad de identificadores |

### **4.3 Métricas de Rendimiento**

| **Métrica** | **Objetivo** | **Método de Medición** |
| --- | --- | --- |
| **Throughput** | ≥ 1000 votos/min/estación | Conteo de votos procesados por unidad de tiempo |
| **Latencia Promedio** | < 3 segundos | Tiempo promedio de procesamiento end-to-end |
| **Utilización de Recursos** | < 80% CPU/Memoria | Monitoreo de sistema durante carga máxima |
| **Concurrencia** | 4+ estaciones simultáneas | Número de estaciones activas sin degradación |

## **5. Implementación del Patrón Reliable Messaging**

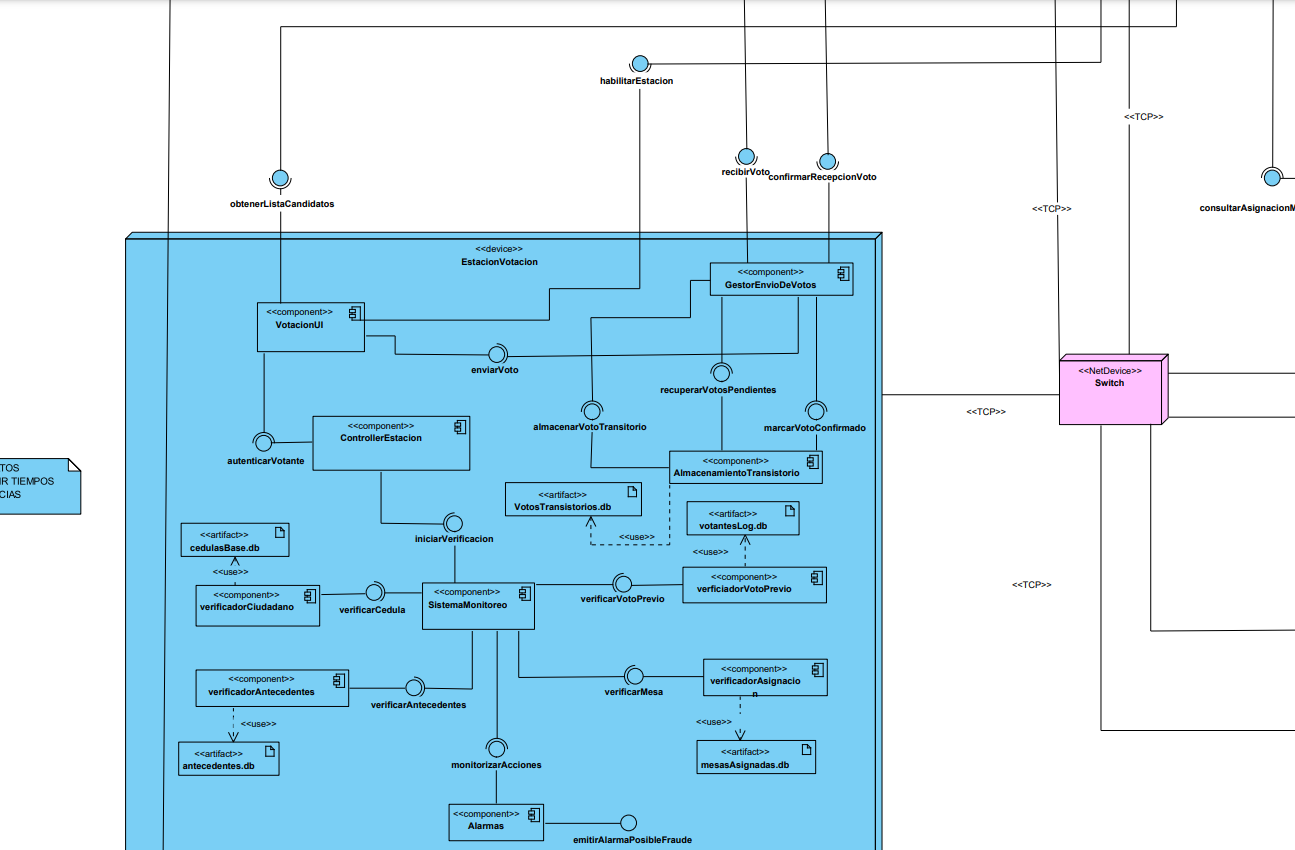
### **5.1 Arquitectura del Patrón**

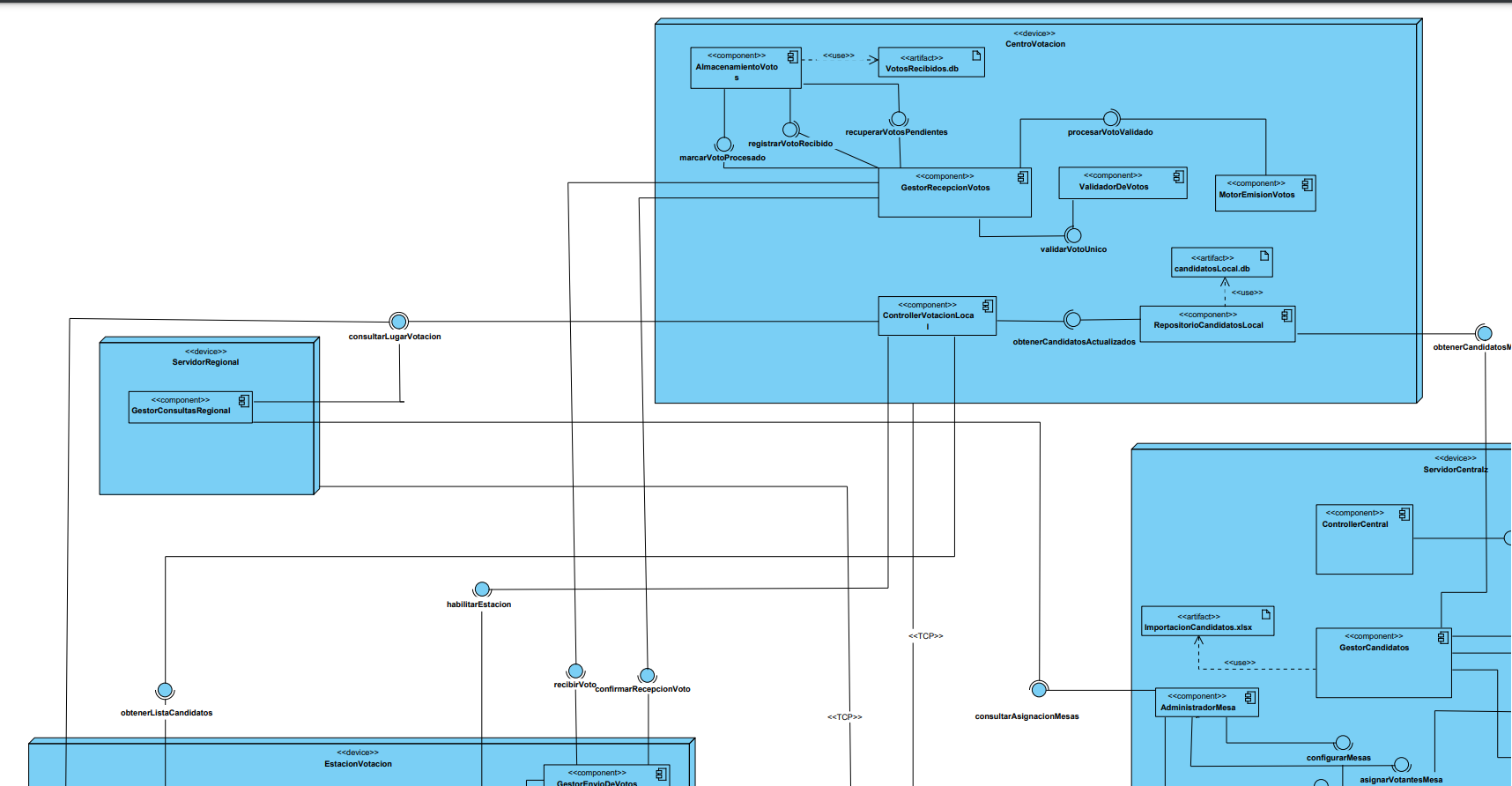
El sistema implementa una arquitectura **Store-and-Forward** con los siguientes elementos:

[Estación] → [Almacenamiento Transitorio] → [Gestor Envío] → [Centro] → [Almacenamiento Central]

↓ ↓ ↓ ↓ ↓

[VotacionUI] → [AlmacenamientoTransitorio] → [GestorEnvio] → [GestorRecepcion] → [AlmacenamientoVotos]





### **5.2 Mecanismos de Confiabilidad**

**1. Persistencia Local Garantizada:**

* Todo voto se almacena localmente antes de transmisión
* Estados: PENDIENTE → RECIBIDO → PROCESADO
* Archivo: VotosTransitorios.csv

**2. Acknowledgments Bidireccionales:**

* Confirmación de recepción desde centro
* Callback con estado actualizado
* Reintento automático si no hay confirmación

**3. Recuperación Automática:**

* Scheduler cada 30 segundos verifica pendientes
* Reenvío automático de votos no confirmados
* Mantenimiento de estado persistente

### **5.3 Mecanismos de Unicidad**

**1. UUIDs Únicos por Voto:**

String votoId = UUID.randomUUID().toString();

**2. Validación Centralizada:**

boolean esUnico = validadorDeVotos.validarVotoUnico(voto.votoId);

**3. Control de Sesión Local:**

Set<String> cedulasVotadas = new HashSet<>();

## **6. Procedimientos de Validación**

### **6.1 Configuración del Ambiente de Pruebas**

**Infraestructura Requerida:**

* 1 servidor para Centro de Votación (puerto 10001)
* 4 instancias de Estación de Votación (puertos 10000+)
* Red local con capacidad de simulación de fallos
* Almacenamiento suficiente para logs y archivos CSV

**Datos de Prueba:**

* 50 cédulas autorizadas (1000000001-1000000050)
* 4 candidatos predefinidos
* Distribución aleatoria de votos

### **6.2 Ejecución de Pruebas**

**Comandos de Inicio:**

# Terminal 1: Centro de Votación

./gradlew runCentro

# Terminal 2-5: Estaciones de Votación

./gradlew runEstacion --args="--id Estacion01"

./gradlew runEstacion --args="--id Estacion02"

./gradlew runEstacion --args="--id Estacion03"

./gradlew runEstacion --args="--id Estacion04"

### **6.3 Monitoreo y Logging**

**Archivos de Monitoreo:**

* data/centro/VotosRecibidos.csv: Registro central de votos
* data/estacion/VotosTransitorios.csv: Almacenamiento local por estación
* Logs de aplicación con nivel INFO/DEBUG
* Timestamps precisos para análisis de rendimiento

## **7. Criterios de Aceptación**

### **7.1 Confiabilidad (Requisito Crítico)**

**EXITOSO si:**

* Tasa de entrega = 100% en todos los escenarios
* Tiempo de recuperación post-fallo < 2 minutos
* Consistencia entre almacenamientos local y central = 100%

**FALLIDO si:**

* Pérdida de cualquier voto durante transmisión
* Inconsistencias entre registros locales y centrales
* Fallas de recuperación que excedan tiempo límite

### **7.2 Unicidad (Requisito Crítico)**

**EXITOSO si:**

* Detección de duplicados = 100%
* Falsos positivos = 0%
* Un votante puede votar máximo una vez por sesión

**FALLIDO si:**

* Cualquier voto sea contado más de una vez
* Votante autorizado sea rechazado incorrectamente
* UUIDs duplicados en el sistema

### **7.3 Rendimiento (Requisito Operacional)**

**EXITOSO si:**

* Throughput ≥ 1000 votos/minuto por estación
* Latencia promedio < 3 segundos
* Sistema soporta 4+ estaciones concurrentes

**ACEPTABLE CON RESERVAS si:**

* Rendimiento 80-99% del objetivo
* Latencias ocasionales > 3 segundos pero < 10 segundos

## **8. Reportes de Resultados**

### **8.1 Estructura del Reporte**

Para cada escenario ejecutado, documentar:

1. **Datos de Entrada:**
   * Número de votos procesados
   * Configuración del sistema
   * Condiciones de red simuladas
2. **Métricas Observadas:**
   * Tiempos de respuesta medidos
   * Tasa de éxito de transmisión
   * Recursos utilizados
3. **Evidencia de Validación:**
   * Screenshots de archivos CSV
   * Logs relevantes con timestamps
   * Conteos de verificación cruzada
4. **Análisis de Desviaciones:**
   * Cualquier comportamiento inesperado
   * Tiempo de recuperación observado
   * Lecciones aprendidas

### **8.2 Conclusiones de Validación**

**El sistema será considerado VÁLIDO para producción si:**

* Todos los criterios de aceptación críticos son cumplidos
* La evidencia documental demuestra trazabilidad completa
* Los mecanismos de recuperación funcionan bajo todas las condiciones probadas
* El patrón Reliable Messaging garantiza integridad absoluta de datos

**Entregables de Validación:**

* Reporte ejecutivo de resultados
* Evidencia técnica (logs, archivos, screenshots)
* Recomendaciones para optimización
* Certificación de cumplimiento de requisitos críticos

## **9. Conclusión**

Esta metodología de validación garantiza que el sistema de votación electrónico cumple con los estándares más exigentes de confiabilidad y unicidad requeridos para procesos electorales. La implementación del patrón Reliable Messaging, junto con mecanismos robustos de persistencia y validación, asegura que ningún voto se pierda y ningún voto sea contado más de una vez, satisfaciendo así los requisitos críticos establecidos por la Registraduría.

La validación exitosa de estos experimentos certifica que el sistema está listo para su despliegue en procesos electorales reales, manteniendo la integridad democrática y la confianza ciudadana en el sistema electoral.