# Sterling Integrator & Sterling File Gateway 6.1 Property Files - Detailed Study Material

## Introduction

IBM Sterling B2B Integrator (SI) and Sterling File Gateway (SFG) version 6.1 are enterprise-grade B2B integration platforms that rely heavily on property files for configuration and customization. Property files control every aspect of the system—from database connections and server ports to performance tuning and security settings. Understanding these property files is essential for administrators, implementers, and support personnel working with Sterling products.

This study material provides comprehensive coverage of property file types, structure, management strategies, and detailed examination of critical property files in Sterling Integrator and Sterling File Gateway 6.1 environments.

## Property File Architecture

### Property File Types

Sterling B2B Integrator uses several types of property files, each serving a specific purpose in the configuration lifecycle:

| File Type | Description |
|---|---|
| *.properties | Operational files used during runtime. These are the active configuration files referenced by the application during normal operations. |
| *.properties_ext | Extension files for application-specific customization. Provide additional configuration beyond base properties files. |
| *.properties.in | Initialization files used during installation. Set initial values for *.properties files. Serve as templates. |
| *.properties*_ext.in | Initialization files for extension files. Set initial values for *.properties*_ext files during installation. |
| customer_overrides.properties | Master override file that maintains changes across reinstallations, patches, and upgrades. Overrides both .in files and runtime properties. |
| sandbox.cfg | Parameter file containing name-value pairs merged with .in files during property file generation. |

Table 1: Property File Types in Sterling B2B Integrator

## File Location

All property files are located in a standard directory structure:

- **UNIX/Linux**: install_dir/properties
- **Windows**: install_dir\properties
- **Installation templates**: install_dir/install/properties (for .in files)

## Property File Hierarchy

Understanding the hierarchy and precedence of property files is crucial for effective configuration management:

1. **Installation Phase**: *.properties.in files → generate → *.properties files
2. **Runtime Phase**: Application reads *.properties files
3. **Override Phase**: customer_overrides.properties values supersede all other sources
4. **Parameter Substitution**: sandbox.cfg parameters are merged into .in files when setupfiles is executed

## Critical Configuration File: sandbox.cfg

## Purpose and Function

The sandbox.cfg file is a critical configuration file containing name-value parameters that are merged with initialization (.in) files to create final runtime properties files[8]. This file acts as a centralized parameter store for installation-specific values.

## Parameter Syntax

Parameters in .in files that pull values from sandbox.cfg are identified with the `&` and `;` delimiter characters:

Example in jdbc.properties.in:
oraclePool.user=&ORA_USER;

Corresponding entry in sandbox.cfg:
ORA_USER=oracle

Resulting value in jdbc.properties:
oraclePool.user=oracle

## Key sandbox.cfg Parameters

| Parameter | Description |
|---|---|
| INSTALL_DIR | Installation directory path for Sterling B2B Integrator |
| DB_VENDOR | Database vendor (Oracle, DB2, MSSQL, MySQL) |
| ORA_USER, DB2_USER, etc. | Database username for specific database type |
| ORA_PASSWORD, DB2_PASSWORD | Database password (encrypted or plain text) |
| ORA_HOST, DB2_HOST | Database server hostname or IP address |
| ORA_PORT, DB2_PORT | Database listener port |
| ORA_SID, DB2_DATABASE | Database instance name or database name |
| PORT1 | Base port for installation (starting port in port range) |
| PORT2 | SSL port |
| LIST_PORT | Base port for installation (same as PORT1) |
| JMS_PORT | JMS messaging port |
| JMX_PORT | JMX monitoring port for JConsole connectivity |
| RMI_PORT | RMI registry port |
| ADMIN_EMAIL_ADDRESS | Administrator email for system alerts |
| SI_ADMIN_SMTP_HOST | SMTP server for sending administrative alerts |
| JVM_LOC | Java Development Kit (JDK) installation location |
| IS_CLUSTER | Indicates cluster configuration (YES/NO) |
| PARTITION_NAME | Cluster partition identifier |
| LOCAL_JNDI_PORT | JNDI service port for clustering |

Table 2: Important sandbox.cfg Parameters

## Modifying sandbox.cfg

When changing sandbox.cfg parameters:

1. Stop Sterling B2B Integrator
2. Edit sandbox.cfg file in install_dir/properties directory
3. Run setupfiles script to regenerate property files:

- UNIX/Linux: ./setupfiles.sh
- Windows: setupfiles.cmd

4. Restart Sterling B2B Integrator

**Important Note**: Most sandbox.cfg parameters are not used at runtime. Changes require running setupfiles to propagate values into operational property files.

## Master Override File: customer_overrides.properties

## Purpose and Importance

The customer_overrides.properties file is the most important file for maintaining custom configurations across system lifecycle events. This file ensures that customizations survive:

- System upgrades and patches
- Reinstallations
- Execution of setupfiles script
- Version migrations

## Override Syntax

Each override entry in customer_overrides.properties follows this format:

PROPERTY_FILE_NAME_PREFIX.PROPERTY_NAME=PROPERTY_VALUE

Where:

- **PROPERTY_FILE_NAME_PREFIX**: Name used in servers.properties to reference the property file
- **PROPERTY_NAME**: Exact property name from the target property file
- **PROPERTY_VALUE**: New value to assign

## Finding Property File Prefixes

To determine the correct PROPERTY_FILE_NAME_PREFIX, examine the servers.properties file:

Example from servers.properties:
jdbcService=jdbc.properties

Therefore, the prefix for jdbc.properties is: jdbcService

## Examples of customer_overrides.properties Entries

**Database Connection Pool Configuration**:
jdbcService.oraclePool.max=100
jdbcService.oraclePool.min=10
jdbcService.oraclePool.timeout=300

**HTTP Server Port Configuration**:
httpServerAdapter.B2B_HTTP_PORT=8080

**Performance Tuning Parameters**:
noappService.persistence_level=PERSISTENCE_NONE
noappService.SchedulingPolicyName=FAST

**Security and Authentication**:
securityService.LDAP_ENABLED=true
securityService.LDAP_URL=ldap://ldapserver.company.com:389

# Files That Cannot Be Overridden

Certain property files do not support customer_overrides.properties:

- archivethread.properties
- security.properties (some properties)
- tuning.properties
- ui.properties

For these files, edit the corresponding .properties.in file and run setupfiles.

# Best Practices for customer_overrides.properties

1. **Document all changes**: Add comments explaining why each override exists
2. **Version control**: Keep customer_overrides.properties in version control system
3. **Backup before changes**: Always backup before modifying
4. **Test in lower environments**: Validate overrides in dev/test before production
5. **Avoid whitespace**: Be careful with leading/trailing spaces—they are respected by the application
6. **Use for SFG too**: Sterling File Gateway configurations should also use this file

## Critical Property Files in Sterling Integrator

### jdbc.properties

Controls database connectivity and connection pool management.

**Key Properties**:

| Property | Description |
|---|---|
| oraclePool.user | Database username for Oracle connections |
| oraclePool.password | Database password (encrypted) |
| oraclePool.url | JDBC connection URL |
| oraclePool.driver | JDBC driver class name |
| oraclePool.min | Minimum number of connections in pool |
| oraclePool.max | Maximum number of connections in pool |

| | |
|---|---|
| oraclePool.timeout | Connection timeout in seconds |
| oraclePool.increment | Number of connections to add when pool exhausted |
| oraclePool.validate | SQL query to validate connections |

Table 3: jdbc.properties Key Configuration

**Example Configuration**:
oraclePool.user=SIDB_USER
oraclePool.url=jdbc:oracle:thin:@dbserver:1521:SIPROD
oraclePool.driver=oracle.jdbc.driver.OracleDriver
oraclePool.min=10
oraclePool.max=100
oraclePool.timeout=300
oraclePool.increment=5
oraclePool.validate=SELECT 1 FROM DUAL

**Override Example**:

## In customer_overrides.properties

jdbcService.oraclePool.max=150
jdbcService.oraclePool.min=20

## noapp.properties and noapp.properties_*_ext

Controls core application server-independent (ASI) runtime behavior and performance tuning.

**Key Properties**:

| Property | Description |
|---|---|
| persistence_level | Controls database persistence for workflow processing. Values: PERSISTENCE_FULL, PERSISTENCE_NONE |
| SchedulingPolicyName | Workload scheduling algorithm. Values: FAST, THOROUGH, BALANCED |
| NumWorkflowExecutionThreads | Number of threads for workflow execution |
| MaxSystemResources | Maximum system resource objects in memory |
| MaxInMemoryRouteQueue | Queue size for routing operations |

Table 4: noapp.properties Performance Settings

**Performance Tuning Recommendations**:

## For high-throughput production environments

persistence_level=PERSISTENCE_NONE
SchedulingPolicyName=FAST

NumWorkflowExecutionThreads=50
MaxSystemResources=5000

**Important Note**: The persistence_level property is set in noapp.properties_gis_ext.in file.

## servers.properties

Defines property file prefixes and mappings used throughout the system.

**Format**:
prefix=filename.properties

**Examples**:
jdbcService=jdbc.properties
opsService=ops.properties
securityService=security.properties
httpServerAdapter=httpserveradapter.properties

This file is essential for understanding which prefix to use in customer_overrides.properties.

## ops.properties

Contains operational configuration settings for business processes and adapters.

**Key Properties**:

- **operational_server_flag**: Indicates if server is operational
- **scheduling_queue_size**: Size of internal scheduling queues
- **enable_transaction_logging**: Controls detailed transaction logging
- **max_retry_count**: Maximum retry attempts for failed operations

## security.properties

Manages authentication, authorization, and security policies.

**Key Properties**:

| Property | Description |
|---|---|
| LDAP_ENABLED | Enable/disable LDAP authentication (true/false) |
| LDAP_URL | LDAP server URL (ldap://server:port) |
| LDAP_BASE_DN | Base Distinguished Name for LDAP searches |
| LDAP_USER_DN | User DN pattern for authentication |
| LDAP_FACTORY | LDAP context factory class (com.sun.jndi.ldap.LdapCtxFactory) |
| PASSWORD_POLICY | Password complexity requirements |
| SESSION_TIMEOUT | User session timeout in minutes |

Table 5: security.properties Configuration

**Note**: Some security properties cannot be overridden via customer_overrides.properties. Check file comments for guidance.

## authentication_policy.properties

Configures authentication providers and policies.

**Configuration Pattern**:
authentication_<number>.display_name=GIS Authentication
authentication_<number>.jndi_factory=com.sun.jndi.ldap.LdapCtxFactory
authentication_<number>.provider_url=ldap://ldap.company.com:389
authentication_<number>.security_authentication=simple
authentication_<number>.base_dn=ou=users,dc=company,dc=com

**Multiple Authentication Providers**:

You can configure multiple authentication providers (internal database, LDAP, Active Directory) by incrementing the `<number>` suffix.

**Sterling File Gateway Requirement**: In SFG environments, override both authentication_policy.properties entries in customer_overrides.properties to ensure consistent authentication.

## cluster_control.properties

Manages clustering configuration for high-availability environments.

**Key Properties**:

| Property | Description |
|---|---|
| partition_name | Unique identifier for cluster partition |
| local_jndi_url | JNDI provider URL for cluster communication |
| jms_provider_url | JMS provider URL for messaging |
| cluster_mode | Enable/disable cluster mode (true/false) |
| node_id | Unique identifier for cluster node |
| heartbeat_interval | Interval for cluster health checks (seconds) |

Table 6: cluster_control.properties Settings

## archivethread.properties

Controls the Purge service functionality for database maintenance.

**Key Properties**:

- **archive_thread_count**: Number of threads for archiving operations
- **purge_batch_size**: Number of records to purge in single batch
- **retention_days**: Days to retain archived data

- **purge_interval**: Frequency of automatic purge operations

**Modification Method**: Cannot use customer_overrides.properties. Edit archivethread.properties.in file directly and run setupfiles.

## tuning.properties

Advanced JVM and application tuning parameters.

**Key Areas**:

- JVM heap size settings
- Garbage collection parameters
- Thread pool configurations
- Memory allocation settings

**Modification Method**: Edit tuning.properties.in file as it does not support customer_overrides.properties.

## httpserveradapter.properties

Configures HTTP/HTTPS server adapters for inbound communications.

**Key Properties**:

| Property | Description |
|---|---|
| B2B_HTTP_PORT | HTTP listener port (base port + offset) |
| B2B_HTTPS_PORT | HTTPS listener port |
| MAX_CONNECTIONS | Maximum concurrent HTTP connections |
| SOCKET_TIMEOUT | Socket read timeout (milliseconds) |
| SSL_ENABLED | Enable SSL/TLS (true/false) |
| KEYSTORE_PATH | Path to SSL keystore file |
| KEYSTORE_PASSWORD | Keystore password |

Table 7: httpserveradapter.properties Configuration

## jms11.properties

Configures JMS 1.1 messaging services and connection pools.

**Key Properties**:

DEFAULT_TIME_TO_LIVE=300000
MAX_POOL_SIZE=50
MIN_POOL_SIZE=5
CONNECTION_FACTORY_JNDI=ConnectionFactory
INITIAL_CONTEXT_FACTORY=org.apache.activemq.jndi.ActiveMQInitialContextFactory
PROVIDER_URL=tcp://localhost:61616

**JMX Monitoring**: Use JMX_PORT from sandbox.cfg to monitor JMS pools via JConsole.

# Sterling File Gateway Specific Property Files

## Overview

Sterling File Gateway (SFG) runs on top of Sterling B2B Integrator and shares the same property file infrastructure. However, SFG introduces additional configuration requirements specific to file transfer operations.

### customer_overrides.properties in SFG Context

Sterling File Gateway requires careful attention to customer_overrides.properties because it affects both SI and SFG layers:

**SFG-Specific Overrides**:

### Routing channel configuration

routingService.IGNORE_TEMP_FILES=true
routingService.TEMP_FILE_PATTERN=.*.tmp$

### File structure processing

fileStructureService.MAX_FILE_SIZE=2147483648
fileStructureService.TIMEOUT=3600

### Authentication for SFG users

authentication_1.display_name=SFG LDAP Authentication
authentication_1.jndi_factory=com.sun.jndi.ldap.LdapCtxFactory
authentication_1.provider_url=ldap://ldapserver:389

# Routing Configuration Properties

Sterling File Gateway uses routing channels extensively. Configuration is driven by properties controlling:

- Producer and consumer mailbox paths
- File name pattern matching using regular expressions
- File structure layers (ZIP, GZIP, PGP, Text, Unknown)
- Provisioning facts for dynamic routing
- System facts (ProducerName, ConsumerName, ProducerFilename)

**Example Regular Expression Configuration**:

### File name pattern with parenthetic grouping

producer.filename.pattern=^(.*?)_(\d{4})(\d{2})(\d{2}).dat$
producer.filename.facts=sanitizedFilename,fileYear,fileMonth,fileDay

### Routing based on extracted facts

consumer.mailbox.path=/*ConsumerName*/{fileYear}/${fileMonth}/Inbox

## File Structure Properties

File structures define the layers and formats of files being transferred:

| Layer Type | Description |
|---|---|
| Container Layers | ZIP, GZIP, PGP (can contain nested layers) |
| Primitive Layers | Text, Unknown (innermost layer, no nesting) |

Table 8: SFG File Structure Layer Types

**Configuration Principle**: Producer file structure must match consumer file structure. If consumer is Unknown, producer can be Text or Unknown.

## Ignoring Temporary Files

To prevent routing of temporary files during transfer:

### In customer_overrides.properties

routingService.IGNORE_TEMP_FILES=true
routingService.TEMP_FILE_EXTENSIONS=.tmp,.temp,.partial

# Property File Management Procedures

## Procedure 1: Modifying Properties Using .in Files

This is the recommended method for properties that support initialization files:

1. Stop Sterling B2B Integrator
2. Navigate to install_dir/install/properties directory
3. Edit the appropriate *.properties.in file
4. Save changes (trim whitespace carefully)
5. Run setupfiles script:
    - UNIX/Linux: cd install_dir/bin; ./setupfiles.sh
    - Windows: cd install_dir\bin; setupfiles.cmd
6. Verify changes in corresponding *.properties file
7. Start Sterling B2B Integrator
8. Test functionality

**Cluster Environments**: Perform this procedure on each cluster node.

## Procedure 2: Using customer_overrides.properties (Preferred)

This method ensures changes survive upgrades and reinstallations:

1. Identify the property to override:
    - Locate property in target *.properties file
    - Find PROPERTY_FILE_NAME_PREFIX from servers.properties
2. Stop Sterling B2B Integrator
3. Edit customer_overrides.properties in install_dir/properties
4. Add entry: PROPERTY_FILE_NAME_PREFIX.PROPERTY_NAME=VALUE
5. Save file (verify no whitespace issues)
6. Run setupfiles script
7. Start Sterling B2B Integrator
8. Verify override took effect:
    - Check target *.properties file for new value
    - Monitor logs for errors
    - Test affected functionality

**Example Workflow**:

### Step 1: Identify property

### In jdbc.properties: oraclePool.max=100

### In servers.properties: jdbcService=jdbc.properties

### Step 2: Add to customer_overrides.properties

jdbcService.oraclePool.max=150

### Step 3: Run setupfiles

cd /opt/IBM/SterlingB2BIntegrator/bin
./setupfiles.sh

### Step 4: Verify in jdbc.properties

grep "oraclePool.max" ../properties/jdbc.properties

### Should show: oraclePool.max=150

# Procedure 3: Direct Property File Editing (Last Resort)

Only use when the file does NOT support customer_overrides.properties and has no .in file[3]:

1. Stop Sterling B2B Integrator

2. Navigate to install_dir/properties
3. Backup the property file
4. Edit the *.properties file directly
5. Save changes carefully (trim whitespace)
6. Start Sterling B2B Integrator
7. Test changes

**Warning**: Direct edits may be lost during upgrades or when setupfiles runs. Document all direct edits externally.

# Procedure 4: Modifying sandbox.cfg Parameters

For changing installation-specific parameters:

1. Stop Sterling B2B Integrator
2. Backup sandbox.cfg file
3. Edit install_dir/properties/sandbox.cfg
4. Modify parameter values:
    - Example: PORT1=8080 changed to PORT1=9080
5. Save file
6. Run setupfiles script (propagates changes to dependent .properties files)
7. Verify changes in affected property files
8. Start Sterling B2B Integrator
9. Update any hardcoded references in business processes

# Whitespace Handling

**Critical Warning**: Property files respect leading and trailing whitespace, which can cause unexpected behavior.

**Best Practices**:

- Trim all whitespace before saving
- Use editors that show whitespace characters
- Avoid spaces around = signs: `property=value` not `property = value`
- Test after editing to catch whitespace issues early

# Common Property File Scenarios

# Scenario 1: Changing Database Connection Pool Size

**Requirement**: Increase Oracle connection pool from 100 to 200 connections.

**Solution**:

### In customer_overrides.properties

jdbcService.oraclePool.max=200
jdbcService.oraclePool.min=20
jdbcService.oraclePool.increment=10

Run setupfiles and restart.

# Scenario 2: Enabling LDAP Authentication

**Requirement**: Configure LDAP authentication for user management.

**Solution**:

### In customer_overrides.properties

securityService.LDAP_ENABLED=true
securityService.LDAP_URL=ldap://ldap.company.com:389
securityService.LDAP_BASE_DN=ou=users,dc=company,dc=com

### In authentication_policy override

authentication_1.display_name=Corporate LDAP
authentication_1.jndi_factory=com.sun.jndi.ldap.LdapCtxFactory
authentication_1.provider_url=ldap://ldap.company.com:389
authentication_1.security_authentication=simple

# Scenario 3: Changing HTTP Port

**Requirement**: Move HTTP adapter from port 8080 to 9080 due to port conflict.

**Solution Option 1 - Via sandbox.cfg**:

### Edit sandbox.cfg

PORT1=9080

### Run setupfiles - automatically updates all port-dependent properties

**Solution Option 2 - Via customer_overrides.properties**:
httpServerAdapter.B2B_HTTP_PORT=9080

# Scenario 4: Performance Tuning for High Volume

**Requirement**: Optimize system for high-volume transaction processing.

**Solution**:

### In customer_overrides.properties

### Set persistence to NONE for better performance

noappService.persistence_level=PERSISTENCE_NONE

### Use FAST scheduling

noappService.SchedulingPolicyName=FAST

### Increase workflow threads

noappService.NumWorkflowExecutionThreads=75

## Increase connection pool

jdbcService.oraclePool.max=200
jdbcService.oraclePool.min=50

# Increase HTTP connections

httpServerAdapter.MAX_CONNECTIONS=500

## Scenario 5: Configuring Cluster Node

**Requirement**: Add new node to existing Sterling cluster.

**Solution**:

### In sandbox.cfg for new node

IS_CLUSTER=YES
PARTITION_NAME=PROD_CLUSTER
LOCAL_JNDI_PORT=2809
NODE_ID=NODE3

### In customer_overrides.properties

clusterControlService.cluster_mode=true
clusterControlService.node_id=NODE3
clusterControlService.heartbeat_interval=30

## Scenario 6: SFG Routing Channel Optimization

**Requirement**: Configure routing to ignore temporary files and process only completed files.

**Solution**:

### In customer_overrides.properties

routingService.IGNORE_TEMP_FILES=true
routingService.TEMP_FILE_PATTERN=.*.(tmp|temp|partial)$
routingService.MIN_FILE_AGE=60
fileStructureService.MAX_FILE_SIZE=5368709120

# Troubleshooting Property File Issues

## Issue 1: Property Changes Not Taking Effect

**Symptoms**: Modified property in customer_overrides.properties but system still uses old value.

**Causes and Solutions**:

1. **Forgot to run setupfiles**
    - Solution: Run setupfiles.sh/setupfiles.cmd after editing customer_overrides.properties
2. **Incorrect property file prefix**
    - Solution: Verify prefix in servers.properties file
3. **Property does not support override**
    - Solution: Check if file is in exclusion list (archivethread, tuning, ui, security)
    - Alternative: Edit .properties.in file directly
4. **System not restarted**
    - Solution: Restart Sterling B2B Integrator after property changes
5. **Whitespace issues**
    - Solution: Check for leading/trailing spaces in property value

# Issue 2: Database Connection Failures After Property Change

**Symptoms**: Application fails to start or cannot connect to database after modifying jdbc.properties.

**Diagnosis**:

- Check install_dir/logs/noapp/system.log for JDBC errors
- Look for connection pool exceptions
- Verify database is accessible

**Solutions**:

1. Verify JDBC URL syntax is correct
2. Confirm database credentials are accurate
3. Check database listener is running
4. Ensure JDBC driver JAR is in install_dir/lib directory
5. Test connection independently using database client

# Issue 3: Performance Degradation After Tuning

**Symptoms**: System slower after modifying noapp.properties performance settings.

**Common Causes**:

- persistence_level set to PERSISTENCE_FULL (very slow)

- NumWorkflowExecutionThreads set too high (thread contention)
- Connection pool exhaustion (max too low for load)

**Remediation**:

# Optimal production settings

noappService.persistence_level=PERSISTENCE_NONE
noappService.NumWorkflowExecutionThreads=50
jdbcService.oraclePool.max=150

# Issue 4: Port Conflicts

**Symptoms**: Server fails to start with "Address already in use" error.

**Solution**:

1. Identify conflicting port in error message
2. Check which process is using the port:
    - UNIX/Linux: netstat -an | grep PORT_NUMBER
    - Windows: netstat -ano | findstr PORT_NUMBER
3. Modify PORT1 in sandbox.cfg or override specific port in customer_overrides.properties
4. Run setupfiles
5. Restart application

# Issue 5: Cluster Synchronization Problems

**Symptoms**: Cluster nodes out of sync, workload not distributing evenly.

**Check**:

- Verify cluster_control.properties on all nodes
- Confirm LOCAL_JNDI_PORT is unique per node
- Validate JMS connectivity between nodes
- Check firewall rules for cluster ports

**Resolution**:

# Verify on each node

grep "node_id" properties/cluster_control.properties
grep "partition_name" properties/cluster_control.properties

**Ensure consistency**

**All nodes should have same PARTITION_NAME**

**Each node should have unique NODE_ID**

# Advanced Topics

## Custom Property Files

You can create custom property files for application-specific configurations[10].

**Steps**:

1. Create property file in custom location:
    - Example: install_dir/custom/myapp.properties
2. Define mapping in servers.properties:
    - myAppService=custom/myapp.properties
3. Access properties in business processes using sci-get-property() function
4. Override properties via customer_overrides.properties:
    - myAppService.myProperty=myValue

## Property Encryption

Sensitive properties (passwords, keys) should be encrypted.

**Encryption Methods**:

1. Use Sterling encryption utilities:
    - encrypt_string.sh/encrypt_string.cmd
2. Store encrypted value in property file:
    - oraclePool.password=ENCRYPTED:aes256:base64encodedvalue
3. System automatically decrypts at runtime

## Property File Validation

**Validation Script** (custom development):

#!/bin/bash

# validate_properties.sh

PROPS_DIR=/opt/IBM/SterlingB2BIntegrator/properties

echo "Validating property files..."

# Check critical files exist

for file in jdbc.properties noapp.properties ops.properties; do
if [ ! -f "$PROPS_DIR/file" ]; then

```
echo "ERROR: $file not found"
exit 1
fi
done
```

# Check for common issues

```
grep -n "[1].=" $PROPS_DIR/.properties
if [ $? -eq 0 ]; then
echo "WARNING: Properties with leading whitespace found"
fi

echo "Validation complete"
```

# Property File Backup Strategy

**Best Practices**:

1. Backup before ANY changes:
    - cp customer_overrides.properties customer_overrides.properties.bak_YYYYMMDD
2. Version control integration:
    - Maintain customer_overrides.properties in Git/SVN
    - Track all changes with commit messages
3. Environment-specific versions:
    - customer_overrides.properties.DEV
    - customer_overrides.properties.QA
    - customer_overrides.properties.PROD
4. Automated backup in cron job:
    - Daily backup of properties directory
    - Retain 30 days of backups

# Migration and Upgrade Considerations

**Pre-Upgrade**:

1. Backup entire properties directory
2. Document all customizations in customer_overrides.properties
3. Export sandbox.cfg parameters
4. Test customer_overrides.properties in upgrade test environment

**Post-Upgrade**:

1. Review new .properties.in files for new parameters
2. Merge new parameters into customer_overrides.properties if customization needed
3. Run setupfiles to apply all overrides

4. Validate all functionality
5. Update documentation with new properties

# Property File Documentation Template

For enterprise environments, maintain documentation for each customized property:

| Field | Example |
|---|---|
| Property Name | jdbcService.oraclePool.max |
| Property File | jdbc.properties |
| Current Value | 200 |
| Default Value | 100 |
| Change Date | 2025-01-15 |
| Changed By | John Smith |
| Reason | Increase capacity for high volume processing |
| Related Ticket | JIRA-12345 |
| Environment | PROD |
| Tested In | DEV, QA |

Table 9: Property Change Documentation Template

# Summary and Best Practices

## Key Takeaways

1. **Always use customer_overrides.properties** for customizations to survive upgrades
2. **Run setupfiles** after modifying .in files or customer_overrides.properties
3. **Backup before changes** - property file errors can prevent system startup
4. **Test in lower environments** before production changes
5. **Document all customizations** with reasons and dates
6. **Be careful with whitespace** - it matters in property files
7. **Version control** customer_overrides.properties and sandbox.cfg
8. **Understand file hierarchy** - know which file overrides which

## Production Environment Checklist

Before making property changes in production:

- ☐ Change tested successfully in QA environment
- ☐ Change window scheduled and approved
- ☐ Backup of properties directory completed
- ☐ Rollback plan documented and tested
- ☐ Monitoring alerts configured for affected functionality
- ☐ Support team notified of change
- ☐ Post-change validation steps documented
- ☐ Configuration management database (CMDB) updated

## Quick Reference Command Summary

| Task | Command |
|---|---|
| Run setupfiles (Unix) | cd /opt/IBM/SterlingB2BIntegrator/bin; ./setupfiles.sh |
| Run setupfiles (Windows) | cd C:\IBM\SI\bin; setupfiles.cmd |
| Backup properties | tar -czf props_backup.tar.gz properties/ |
| View property file | cat properties/jdbc.properties |
| Search for property | grep "property_name" properties/*.properties |
| Check for whitespace | cat -A properties/file.properties |
| Validate syntax | ./validate_properties.sh |

Table 10: Common Property File Management Commands

# Conclusion

Property files are the foundation of Sterling B2B Integrator and Sterling File Gateway configuration management. Mastering property file concepts—understanding file types, hierarchy, override mechanisms, and management procedures—is essential for successful administration of Sterling platforms. The customer_overrides.properties file is your most powerful tool for maintaining sustainable, upgrade-safe configurations. Combined with proper backup, documentation, and testing procedures, property file management becomes a reliable and efficient process.

By following the best practices and procedures outlined in this study material, administrators can confidently manage Sterling Integrator and Sterling File Gateway configurations across development, testing, and production environments while maintaining system stability and performance.