Case Study

Enhance Billing Speed

Project Overview

Project Name:
Enhance Monthly Billing for Speed

Industry:
Records and Information Management (RIM)

Role:
Boomi Integration Developer / Architect

Project Duration:
3 Months Maximum

Business Objective

Billing and Accounts Receivable are dissatisfied with the existing End-of-Month (EOM) process, aggregating Invoices/Credit Memos in their SQL-based billing software and pushing those to NetSuite. The current process takes anywhere from 6-8 days, depending on errors that need to be manually fixed and reprocessed. The Team is looking to decrease this process to less than 5 days, if possible.

Systems Integrated

- SQL (Existing Billing System)
- NetSuite (ERP – Invoicing)

High-Level Architecture

Existing Architecture Summary:

- Reliant on the Data Services Team (DS) - processing invoices to a staging table and, when all Invoices for a given Branch were staged, the entire Branch was ready to process by Boomi, causing delays.
- No Error Handling - batches were left incomplete and unknown until the Account Team ran their reports to determine if the process was complete.

- Multiple Maps - Found, in succession, slowing down processing.
- Dynamic Process Properties (DPP) - Used to track document-specific Ids and require a Flow Control to run each document separately.

Proposed Architecture Summary:

- Staging Table – Create Integration to pull Invoices, ready for billing, throughout the month, allowing those Invoices to transfer to NetSuite as soon as it is "closed" (ready to bill). This eliminates any DS interaction or delay as
- Invoice Processing – Create Integration to read the staging table, create "locks" to allow variable-sized batches to transfer to NetSuite. The "locks" will allow multiple Integration Runs at the same time, increasing speed.
    - Limit batches to 1000, max. Allow smaller if less than 500 to push through quicker.
    - Limit Maps – try to map translations once.
    - Add Flow Control – multi-threaded.
- Error Handling – Incorporate Error Handling to log and notify failures so the Integration can be enhanced to handle them in the future or manually fix and reprocess.

Integration Flows

Briefly describe the **key processes**, not every mapping.

1. **Populate Staging Table**
    a. Schedule multiple times per day
    b. Scan Billing System (SQL) for Invoices created on or after the **LastSuccessfulRunDate** and push to Staging table
    c. Existing Invoices in Staging table that are not "closed", check Billing System to see if they have been closed since **LastSuccessfulRunDate.** They don't track a LastModifiedDate on the Invoice.
2. **Push Invoice to Netsuite**
    a. "Unlock" records that have been locked over xx-hours and Invoice has not been created (compensate for network/database/ERP issues) and reduce manual intervention.
    b. "Lock" a batch of Invoices in the Staging Table for this Execution based on **LastSuccessfulRunDate.**
    c. Generate NetSuite Invoice for each "Locked" row in the Staging Table.

d. Log NetSuite meta data for reporting purposes.

Boomi Capabilities Used

- Native connectors (Database, NetSuite)
- Process orchestration and sub-process reuse
- Mapping with validation and default handling
- Error handling with notifications and retries
- Flow Control with threading
- Environment promotion (DEV → QA → PROD)

Key Challenges & Solutions

Challenge:
We want to run multiple instances of the Integration at a time, several minutes apart, to speed up the entire process. With over 54,000 Invoices each month, breaking it up into smaller chunks, running as they are "closed" would be optimal.

Solution:
"Locking" Invoices in batches will allow us to run multiple instances of the Integrations, each processing their own "Locked" Invoices. If a batch fails, for any reason, build an "unlock" to reset the remaining Invoices for processing in the next batch.

Results & Business Impact

- Reduced processing from 6-8 days to 3, dependent upon Accounts Receivable Team "closing" Invoices
- Improved billing turnaround time
- Fewer errors requiring manual intervention to reprocess Invoices
- Scalable foundation for future integrations

Lessons Learned

- Built reusable Boomi components to accelerate future projects
- Standardized error framework improved support response time
- Early data validation significantly reduced downstream failures

Tools & Technologies

- Boomi AtomSphere
- NetSuite API
- SQL Server
- SFTP