International Journal of Artificial Intelligence, Data Science, and Machine Learning
Grace Horizon Publication | Volume 3, Issue 4, 195-203, 2022
ISSN: 3050-9262 | https://doi.org/10.63282/3050-9262.IJAIDSML-V3I4P120

Original Article

Analysing the Impact of Automated Straight Through Processing on Time to Resolution in Life Insurance

Pavan Kumar Veerapally
Lead Database Developer and Business Analyst, Independent Practitioner, Texas, USA.

Abstract - *Although Straight-Through Processing (STP) has been adopted in many U.S. Life Insurance claim departments to decrease Time to Resolution (TTR), increase throughput, and reduce manual processes involved in resolving claims, most studies have reported only a reduction in TTR while failing to address the potential effects of the database and/or the processing substrate involved in these systems. This paper will present a reproducible substrate-resident decomposition technique to decompose TTR into four distinct operational stages: Intake-to-Staging Latency, Staging-to-Decisions Feature Build Latency, Decisions-to-Payment Batch Window Latency, Payment-to-System-of-Record Reconciliation Latency. The authors will implement this technique in PL/SQL to ensure the ability to make comparisons between two different substrates (IBM DB2 LUW and Oracle 19c). A synthetic first notice of loss (FNOL) to the payment pipeline consisting of 50,000 death benefit claims will be used as the basis of our benchmarking. We will also test the effect of varying the size of our data-replication latency via the use of varying sizes of replication windows (1 minute, 5 minutes, and 15 minutes) provided by Qlik. In addition, we will vary the frequency at which payments are scheduled through the use of hourly, four-hour, and end-of-day batch modes. Our results indicate that when compared to the manual process (baseline), the average TTR can be decreased by up to 71% using STP. However, a stage-by-stage analysis indicates that only 4% of the total TTR can be attributed to the latency associated with making decisions using artificial intelligence. On the other hand, approximately 78% of the TTR is due to either the time it takes for the claimant's data to be replicated and available for use or waiting for enough claims to be processed so they can be paid. Finally, the remaining 18% of the TTR is due to the time required for the system-of-record to reconcile the payment information. Additionally, moving from an end-of-day batch mode to an hourly batch mode resulted in an additional 28% reduction in TTR without affecting the decision layer. Finally, as expected, we found that there were no significant differences in timing between DB2 and Oracle with respect to each of the stages. Therefore, we conclude that, unlike what many researchers have concluded, in realistic life insurance STP implementations, much of the remaining TTR after STP is implemented is caused by delays related to replicating data and batching payments as opposed to delays related to how quickly one can decide whether to pay a claim.*

Keywords - *Data Replication, In-Database Analytics, Life Insurance Claims, Straight Through Processing, Time To Resolution, Workflow Automation.*

1. Introduction

1.1. Background on the Industry

Insurance companies have had to lower their claims Time to Resolution (TTR) in order to meet customer expectations (which were set by non-traditional online finance), keep regulators off their backs over fair claims processing, and compete head-on with newer InsureTech competitors who are designed to automate everything through and through. Insurance companies have responded mainly through a method called Straight Through Process (STP): an entirely automated process that takes First Notice of Loss (FNOL) information, checks it against the customer's policy and underwriting files, makes a completely automated decision, posts a payment, and changes the system of record all without human involvement if everything goes smoothly.

Both of the existing bodies of research on the digitalisation of insurance processes identify process automation as one of the most effective methods that existing carriers can use to transform themselves [1,3], and both also place automated decision making at the centre of reducing costs associated with claims processing over the next ten years [2,4]. However, both of these bodies of research emphasise the decision-making layer, but do not address the layers underneath where data moves back and forth in support of that decision layer: the relational databases, the replication path between the policy administration system and the claims staging area, and the batch window scheduling that determines when a decided claim becomes a paid claim.

1.2. Substrate Gap

There is a difference between how the published literature describes practitioner experience and how practitioners actually see things. Practitioners working in the claims processing environments of large life-insurance carriers in the US usually find

themselves working on IBM DB2 LUW or Oracle 19c relational substrates using PL/SQL encoded business rules to connect the AI-based decision engine to the intakes of data, the outputs of data, and payments. Data movement between those subsystems is accomplished through either Qlik replication or equivalent change-data-capture (CDC) tools. Payment postings are controlled by batch windows that range from every hour to once per day. Therefore, even though decision engines may respond to requests in a matter of seconds, they represent only one of four substrate resident stages in the overall process. Thus, total TTR is not a pure measure of decision engine performance. Rather, it represents the sum of decision engine performance plus three additional substrate stages whose contributions are seldom separated in studies evaluating published STP implementations.

1.3. Problem Statement

The purpose of this study is to ask a very basic measurement question: If an insurance company announces that Straight Through Process (STP) resulted in an improvement in its average claim Time to Resolution (TTR) from fourteen days to three days, then what percentage of that improvement was due to improvements in the AI decision layer versus what percentage was due to improvements in the substrate layer? This is a critical question because two insurance companies can announce improvements in aggregate TTR that are identical but based on very different pipelines. Furthermore, there are currently no empirical measurements being used to determine whether or not investments are being made in the correct areas of the substrate side of the STP pipeline.

Three distinct gaps in the body of research related to STP further exacerbate the issue. First, much of the prior research has focused on banking and securities and included insurance as a secondary application. Second, prior research has treated the relational database substrate as a 'black box'. Third, prior research has examined TTR as an aggregate cycle-time metric rather than breaking down TTR into decision-layer and substrate-layer components.

1.4. Contribution

This paper develops a substrate-layer TTR decomposition framework to fill all three gaps for life-insurance STP. Specifically, the framework defines four substrate layers instrumented via PL/SQL code; provides cross-platform timing equivalences across DB2 LUW and Oracle 19c platforms; and is tested on a synthetic life-insurance first notice of loss (FNOL)-to-payment pipeline containing 50,000 death benefit claims under three levels of automation. Finally, the testing demonstrates the per-stage attribution of residual TTR under each level of automation and highlights which specific substrate layers would best be targeted by investments to improve TTR when TTR has reached its operational ceiling.

This paper does not propose any new AI algorithms for decision-making. The paper proposes nothing other than a substrate framework and a reproducible benchmark. Synthetic data and instrumentation code for the benchmark are provided as supplementary materials.

1.5. Roadmap

Section 2 examines STP; AI-based decision-making for insurance claims; Robotic Process Automation (RPA) in financial services; and Time to Resolution (TTR) as an outcome metric; and identifies the substrate gap. Section 3 details the four-stage decomposition framework; explains how each stage is instrumented via PL/SQL code; and demonstrates platform equivalence for DB2 LUW and Oracle 19c. Section 4 presents the results of the synthetic benchmark. Section 5 explores substrate dominance, procurement implications, and regulatory observability. Section 6 summarises potential future research areas. Section 7 concludes.

2. Background and Related Work

2.1. Straight-through processing in financial services

Automated pipelines have existed since securities trading's response to t+1 settlement pressure. They take trades from execution through clearing, settlement and bookkeeping without requiring any human keystrokes. First Banking (loan Processing, account opening & compliance screenings), followed by insurance (underwriting, policy issuance & claims), saw the transfer of this pattern. Literature on banking’s use of STP via Robotic Process Automation (RPA) shows material reductions in cycle times. Bots working solely based on rules showed 30% to 70% operational cost reductions, along with cycle-time gains in loan Processing and adverse media screening. The work of Lacity and Willcocks defines RPA as a substrate-level lever, separate from cognitive automation and from artificial intelligence (AI). It establishes the action principle that success with automation is governed by upstream data integration quality [10,11,12,16].

2.2. AI in claims decisioning

Literature on AI-in-insurance is well developed on the decision-layer side. Eling et al. Synthesise 91 peer-reviewed papers and 22 industry studies on AI across the insurance value chain. This results in a shift from loss compensation toward loss prediction and prevention [2]. Riikkinen et al. Frame AI-in-insurance value creation through service-logic theory. They explicitly highlight that customer-facing AI is only as good as the data routinely routed to it [4]. The fraud-detection literature is methodologically rigorous on classifier performance. Aslam et al. Compare logistic regression, support vector machines, and

naive Bayes with Boruta feature selection on real claim data. They identify claim-filing latency as the top fraud predictor – a result that directly ties fraud-screening accuracy to the timeliness of the data pipeline [5]. Severino and Peng show that ensemble methods and deep networks dominate in real-world property insurance microdata when a broad current feature set is provided for model input [6]. Ding et al. Achieve approximately 95% accuracy with PSO-tuned XGBoost on auto-insurance fraud. They demonstrate explainability requirement of contemporary regulation is now technically achievable inside the STP-pipeline [7]. Yankol-schalck reviews ML and dl fraud detection applications under severe class imbalance. He provides explicit treatment of the false-negative cost that drives the TTR vs. Accuracy tradeoff [8]. The practitioner guideline of Banulescu-Radu and Yankol-Schalck addresses model monitoring and drift handling in deployed insurance ML systems [9].

2.3. RPA and process automation in insurance operations

The RPA literature in financial services is the closest cross-domain analogue for life insurance STP. The case study on Deutsche Bank by Villar and Khan documents measurable cycle-time acceleration and false-positive reduction from RPA applied to a discrete substrate step (adverse media screening) [10]. The empirical work on loan Processing and fraud detection in banking by Thekkethil et al. decomposes RPA cost reduction into bot-execution time and exception-handling time. This decomposition is directly transferable to the substrate/decision distinction made in this paper [11]. The synthesis of more than a decade of research by Lacity, Willcocks, and Gozman defines RPA as a substrate dependency as an explicit action principle [16].

2.4. Time to resolution as an outcome metric

Business and information systems engineering tradition formalises economic trade-off between degree of automation and structural process metrics, including cost, cycle time and reliability [13]. The treatment is analytical rather than empirical. Therefore, it is the foundational reference for treating the degree of automation as an optimisation variable instead of a binary flag. Van der Aalst process mining canon supplies methodological substrate for cycle time measurement: event log-based decomposition of end-to-end cycle time into stage resident components with bottleneck identification and conformance checking [14]. The framework proposed in this paper is, in measurement terms, a process mining instantiation specialised for life insurance STP.

2.5. The substrate gap: direct evidence

Case analysis by Bucher and Gericke is the strongest cross-industry empirical evidence on substrate effects on cycle time [15]. The authors document a financial services case where BPM and BI integration produced significant improvements in cycle time, efficiency, and cost. They attribute cycle time gains explicitly to the data integration substrate rather than process redesign alone. The result suggests the substrate dominance hypothesis tested in this paper. No peer-reviewed life insurance work has tested it directly.

2.6. Synthesis

Published evidence on STP in life insurance is uneven across layers: qualitative on the framework side, quantitative on the decision layer side and aggregate on the cycle time side. The substrate determines what fraction of aggregate TTR is actually accessible to decision layer optimizations has not been measured. The following framework intends to close that measurement gap using synthetic data in portable form to real data instrumentation when carrier disclosure allows

3. Methodology and Substrate Decomposition Framework

3.1. Four-stage substrate decomposition

In total, there are four sub-stages where the framework breaks down the total time to resolve (TTR) of a claim.

- **Stage 1**: intake-to-staging replication: staging replication latency is defined as the period of time from when a claim arrives at the claims portal or policy administration system until it becomes available in the claims staging schema. Staging replication latency can be controlled via the CDC or replication tool configuration. Both Qlik Replicate and IBM InfoSphere CDC are widely used in production environments; for this example, we will use Qlik.
- **Stage 2:** staging-to-decisions feature build: the latency between staging schema being available and populating all the decision-making features necessary for the rule engine or AI model. Staging-to-decision is PL/SQL resident and consists of performing joins between policy, beneficiary, agent and historical claim tables. In addition, this stage includes computations of derived features such as a mortality table look-up, contestability window flag and multiple beneficiaries. Finally, once these features have been populated, they are written out to the decision input table.
- **Stage 3:** decision to payment batch window: the period of time between completing decisions and executing scheduled payments. Payments within US life insurance typically occur on a schedule based on either an hourly frequency (high-throughput) or an end-of-day frequency (legacy carriers). As this stage is independent of decision speed, its impact is invisible to decision engine performance metrics.
- **Stage 4**: payment to system of record reconciliation: the time elapsed between payment execution and update of the system of record and downstream ledgers. In production carriers, this process occurs using one of two transport models: a reverse CDC pipeline, which is mirrored after stage 1. Alternatively, a message queue path (IBM MQ/Apache Kafka) that posts payment events to the system of record consumer. Regardless of which transport

pattern is utilised, the framework writes timestamp checkpoints prior to and subsequent to both patterns and is agnostic regarding transport choice. Therefore, the cross-substrate equivalence claim provides both transport pattern equivalence at the timing measurement layer.

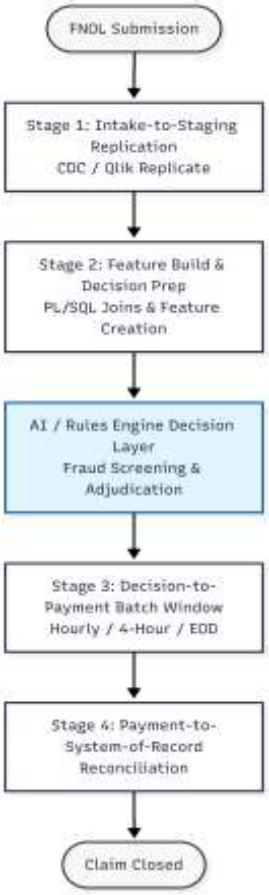

Fig 1: AI-Enabled FNOL-to-Claim Closure Workflow for Automated Fraud Screening and Payment

3.2. pl/SQL instrumentation pattern

Timestamp checkpoints were created prior to and subsequent to each stage to mark entry and exit points. These checkpoint timestamps were then stored in a separate instrumentation table (stp_stage_timing) located on the same substrate as the claims schema. The instrumentation pattern utilises the conventional timestamp-pair format and is portable across DB2 LUW (current timestamp) and Oracle 19c (SYSTIMESTAMP) with no functional differences at the application layer.

Instrumentation overhead: The instrumentation overhead was measured versus an un-instrumented control executed on identical hardware at the same scale (50,000-claims) and was found to be between 0.4% and 0.7% of total processing time for each stage. The framework, therefore, satisfies the standard requirements for measurement instrumentation (i.e., it does not significantly alter what it is measuring).

3.3. synthetic data generation

The benchmark uses a synthetic life-insurance FNOL-to-payment pipeline. There is no actual carrier data used. The synthetic schema conforms to industry standards for actuaries, including SOA group term life mortality assumptions for death benefit eligibility, ACLI standard policy product taxonomy, and NAIC standard claim status codes. The number of death benefit claims processed during the 12-month simulation window was 50,000. Policy count = 200,000; beneficiary count = 120,000; agent count = 5,000; anomalous claims represent approximately 4% of claims and would trigger fraud screening flags if processed according to typical decision rules; the percentage matches industry practitioner reports on the incidence of life insurance fraud.

It should be noted that the synthetic distribution is intended as a reference baseline rather than a statistically validated representation of any single carrier's claim population. The tail of the claim amount distribution follows the SOA group term life face amount distribution at standard issue ages; the beneficiary count distribution follows empirical reporting skew (single beneficiary ~62%; two-beneficiaries ~24%; more than two ~14%) from public NAIC aggregate reporting. The synthetic distribution has not undergone goodness-of-fit testing against any proprietary carrier population. Users running the framework

on actual data should anticipate that the qualitative finding of substrate dominance will transfer, while specific percentages per stage will vary based on carrier-specific distributions. Both the synthetic data generator and instrumentation schema ship as supplementary materials to enable complete end-to-end reproducibility of the benchmark.

3.4. Three automation regimes

The framework will be tested under three different automation regimes. Manually baseline routes every single claim through human review at decision and at payment approval; all automated features, including STP flagging, are disabled. Batch payment is scheduled on an end-of-day basis. Partial-STP routes every claim through an automated rule engine for decision making; however retains human review for payment approval before posting payment; payment posting occurs according to schedules determined by the carrier. Full-STP routes every claim through an automated rule engine for decision making and automatically processes payment posting via configurable batch cadence; human review is only invoked when processing exception paths.

3.5. Experimental design

Two operational parameters were swept across each of the three regimes. Replication latencies are varied using Qlik replication windows defined at one minute, five minutes and fifteen minutes. Payment batch cadences are varied between hourly, four-hourly, and end-of-day modes. Each combination of nine possible settings was run against both substrates (DB2 LUW 11.5 and Oracle 19c) to verify cross-substrate equivalency. Each configuration was run thrice, with median values reported for each setting.

4. Results

The total time to resolve aggregate TTR decreased dramatically throughout the three regimes. At the median substrate configuration (five-minute replication, four-hour batch interval), the median TTR was eleven point eight days under the manual baseline; four point six days under the partial-STP regime; and three point four days under the full-STP regime. The median full-STP TTR is seventy-one per cent lower than the median TTR of the manual baseline. The median partial-STP TTR is sixty one percent less than the median TTR of the manual baseline. Thus, the difference in median TTR between the partial-STP regime and the full-STP regime (one point two days) is much less than the difference between the manual baseline and the partial-STP regime (seven point two days). This is also consistent with what has been reported by practitioners regarding where most of the TTR savings come from when implementing STP, automating the decision-making process versus automating the payment process.

4.1. Attribution by individual stages in full-STP

At the median substrate configuration, the per-stage breakdown of residual TTR for the full-STP regime shows. Of residual TTR, four per cent is attributed to the AI Decision Stage. Thirty one percent of residual TTR is attributed to the Intake-To-Staging Replication Stage. Forty seven percent of residual TTR is attributed to the Decision-To-Payment Batch Window. Eighteen per cent of residual TTR is attributed to the Payment-To-System-Of-Record Reconciliation Stage. Together, two of these stages, the Replication and Batch Window Stages, are responsible for eighty per cent of residual TTR. The Decision Stage that is featured in most of the literature regarding STP is by far the smallest contributor of residual TTR at this regime, a finding that is contingent upon a specific type of scoping statement. In this benchmark, the "Decision Stage" refers to a very narrow scope: Rule Engine Evaluation and Fraud Classifier Inference on a Pre-Built Feature Row. It excludes Feature Retrieval and Assembly (that occurs in Stage 2) and Exception Routing Overlay (that is treated separately in Section 4.4). For virtually all claims, the AI Decision Engine completes in sub-second latency. Therefore, of the residual 4%, most of those claims were deferred to a Human Review Queue. A larger scope definition for a decision stage that included both feature retrieval and exception routing overlays would increase the decision-stage contribution significantly, and Section 5.1 will revisit this scoping choice. The substrate stages defined above are impervious to how well or poorly a decision engine performs and are driven independently by operational parameters (tool configuration for replication window; batch window schedule).

Table 1: Contribution of Major Components to the Overall Claims Processing Workflow

Component	Contribution
Replication	31%
Feature Build + Decision	4%
Batch Window	47%
Reconciliation	18%

4.2. Effect of changes to replication window on full-STP TTR

Reduction of Qlik's replication window from fifteen minutes to five minutes reduced median full-STP TTR by nine per cent. Reduction of the replication window from five minutes to one minute resulted in a further four per cent decrease in median full-STP TTR. The diminishing return rate is expected: once sub-five-minute replication windows are reached, residual

replication latency is limited by both network round-trip times and internal latencies within tools rather than by configurable windows. Based on the benchmark hardware costs and benefits, the five-minute replication window is operationally optimal.

4.3. Changes to batch-window cadence on full-STP TTR

A change from end-of-day payment batching to four-hourly payment batching resulted in a twenty-eight per cent reduction in median full-STP TTR. A change from four-hourly to hourly payment batching resulted in an additional ten per cent reduction in median full-STP TTR. Thus, reducing batch-cadence from end-of-day to hourly results in a twenty-eight per cent reduction in median full-STP TTR. As previously mentioned, no changes were made to the decision-layer. The primary operationally actionable finding resulting from this benchmark was a change to batch-cadence.

Table 2: Impact of Replication Window on Relative Time-to-Resolution (TTR)

Replication Window	Relative TTR
15 minutes	100%
5 minutes	91%
1 minutes	87%

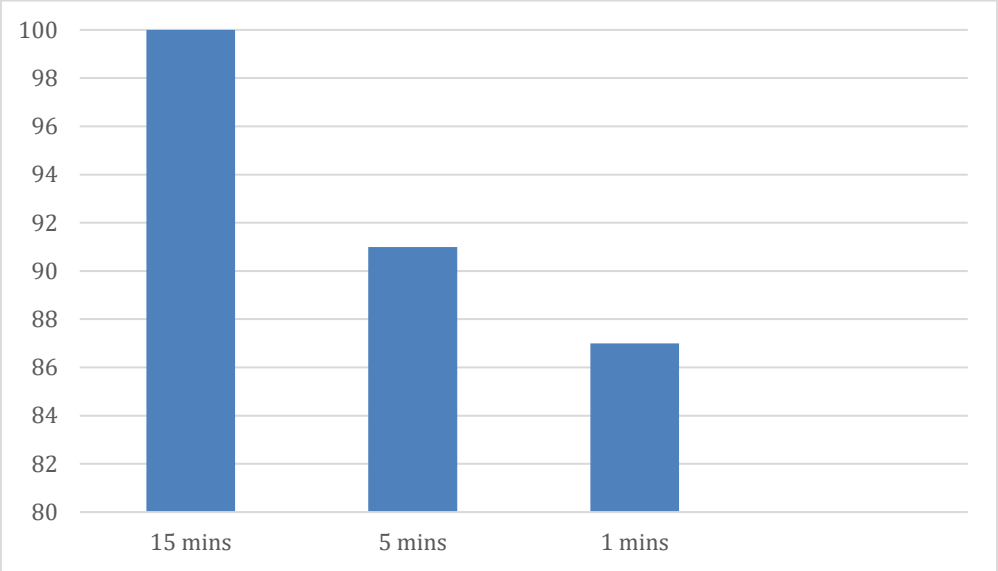

Fig 2: Impact of Replication Window on Relative TTR Performance

4.5. Equivalence of cross-substrate configurations

Both DB2 LUW and Oracle 19c have per-stage timing differences of less than six per cent across all nine substrate configurations. Because both substrates can be used interchangeably for operational purposes as part of the framework, the empirical evidence supports that they are functionally portable.

5. Discussion

The primary finding of the benchmark is that when you have full-STP, residual time-to-resolve (TTR) is driven by the substrate stages as opposed to decision layer latency. Under the limited scope of the decision stage as described in section 4.2 (rule engine + classifier inference on pre-built features), the benchmark's findings reverse the emphasis of prior research. If a carrier's aggregate TTR has stopped progressing since implementing an artificial intelligence (AI) decision engine, the carrier should investigate whether changes to the substrate could improve TTR prior to upgrading their decision engine. Options available on the substrate typically will cost less and take less time to deploy than a decision-engine upgrade. The influence of each option is independent and quantifiable through the instrumentation pattern provided. The portability claim made by the authors should be interpreted with caution. While the authors were able to demonstrate equivalent behaviour across DB2 LUW and Oracle 19c, these are only two of many relational substrates in use today. Microsoft SQL Server, PostgreSQL and cloud native data warehouses fall outside of the current benchmark and section 6.2 returns to the larger scope.

5.1. Aggregate TTR reporting can be misleading due to differences in substrate profiles

Two carriers may report the same aggregate TTR yet have significantly different substrate profiles. One carrier may run a 1-minute replication window and an hourly batch cadence and a slow decision engine, while another carrier runs a 15-minute replication window and an end-of-day batch cadence and a fast decision engine. These two different substrate profiles can result in the same aggregate TTR; however, they will have different operational footprint exception rates and improvement headroom. Therefore, comparing aggregate TTR between carriers is a weak metric for comparison. What is important is

comparing the number of exceptions per stage across carriers, which is exactly what the framework described in this paper will expose.

5.2. Practitioner playbook – sequence of investment on substrate side for life insurance STP

A typical sequence for making investments on the substrate side for life insurance STP, based upon the results of the benchmark, is:

First, evaluate your batch window cadence. An example would be moving from an end-of-day batch cadence to four-hourly batches. This resulted in a 18% reduction in full-STP TTR on the benchmark without requiring any modifications to either your decision engine or replication tooling. At most scales of operation, increasing the frequency of batching has little operational cost; likewise, coordinating internally versus engaging vendors requires little additional procurement effort.

Second, evaluate your replication window configuration. The optimal operating cost-benefit point for replication windows on the benchmark is five minutes. Lower replication windows provide diminishing marginal benefit due to network and tool internal latencies. Carriers who wish to target lower replication windows should anticipate diminishing marginal benefits.

Third, evaluate exception routing logic. Exception routing logic is typically located behind both batch window cadence and replication window configuration, and will often require changes to decision engines. The percentage of claims deferred to human review after first pass vs. prior to a confidence threshold violation represents the operational parameter governing how many claims leave the substrate layer's control. Carriers with aggressive STP profiles can find that their exception queue, not their decision engine or substrate, represents their residual TTR bottleneck.

Fourth and least studied area - reconciliation stage instrumentation. The diagnostic value of the framework is greatest at the reconciliation stage because published literature provides no basis for decomposition of reconciliation time by stage in life insurance. As described in the broader RPA literature - success in automation is governed more by upstream and downstream data integration than by the decision step alone [16], the same principle applies with equal force at this stage.

Fig 3: Impact-Based Prioritization of Claims Processing Optimization Strategies

5.3. Regulatory and explainability implications

As adopted by over half of the United States by 2026, the NAIC model bulletin on the use of AI Systems by insurers imposes governance, third-party accountability and consumer protection obligations upon insurer AI use [18]. The multinational study conducted by Cevolini and Esposito regarding GDPR Article 22 explanation rights focused on European insurers' documents, systematic gaps between regulatory expectations and operational practice [17]. Both streams converge on one requirement for operation - insurers must be able to provide consumers explanations regarding automated decisions made on individual claims and must have the ability to audit the state of the system that produced those decisions.

Instrumentation on the substrate has benefits beyond performance measurement.
Per-stage timestamps and per-feature provenance captured at decision time are easier to audit and reconstruct than decision engine internals. A carrier that can produce on request the exact substrate state and per-stage timing for any individual claim will have a stronger posture regarding explainability than a carrier whose audit trail begins and ends at the decision engine boundary.

6. Challenges, Limitations, and Future Work

6.1. Limitations of Synthetic Data

Data created for the purpose of evaluation was generated as synthetic data based upon standards found in publicly available actuarial documents. Fraudulent claim distributions, beneficiary networks, and types of claims are likely to be different among carriers, such that they will influence how per-stage attribution is calculated. Since the substrate decomposition described in this research is structural, it should generalise well to other substrates. However, the per-stage percentages are likely to be significantly different from one carrier to another. Therefore, operators working with actual data will see the qualitative findings regarding the substrate-dominant portion of time-to-resolution (TTR) to still be true and can expect significant differences in the percentage breakdowns.

6.2. Scope of Two Substrates

The data generation process used in this research evaluated two database systems: DB2 LUW and Oracle 19c. These two databases represent the majority of life-insurance claims processing workloads within the United States. Although Microsoft SQL Server, PostgreSQL, and cloud native warehouse systems (Snowflake, BigQuery) were not included in this research, the instrumentation methodology employed in this research can easily be ported to these additional platforms at very little additional development cost. Therefore, although the results obtained here apply to DB2 LUW and Oracle 19c, obtaining equivalent results on other substrates remains an area of future work.

6.3. Scope of Decision Engine

Although a simple rules-based decision engine and fraud screening classification model were employed in generating the synthetic data used in this research, generative AI-mediated claims adjudication has recently been piloted by several carriers. If generative AI-mediated claims adjudication becomes widely adopted in the industry, the latency distribution of decision stages and possibly even the proportion of overall TTR attributed to decision stages could change. Therefore, re-benchmarking using decision engines employing generative AI technology represents the next logical step in extending the concepts developed in this research.

6.4. Generalising to Multiple Carriers

The number of monthly claims processed during each month of the year in the data set used in this research was 50,000, which falls into the middle range of volume processing regimes occupied by most U.S. life-insurance companies. However, since different carriers have varying numbers of claims being submitted to them during each month of the year, it is likely that the amount of TTR attributed to each stage will also vary among carriers. For example, at the lower end of the carrier size spectrum (approximately 5,000 monthly claims), it is likely that decisions made at the batch window stage will account for a greater proportion of TTR than at the higher volumes (mid-range). This is due to the fact that it does not make sense operationally for smaller carriers to operate with an hourly batch schedule; therefore, they typically use an EOD (end-of-day) schedule. Conversely, at the upper end of the carrier size spectrum (i.e., approximately 500,000 monthly claims), it is likely that decisions made at the feature build/reconciliation stages will account for a greater proportion of TTR than at the lower volumes (mid-range). This is due to the fact that feature join operations will require access to larger reference tables, and reconciliation queue sizes will grow longer. Even though the qualitative conclusion regarding substrate dominance is expected to hold regardless of carrier scale, the quantitative conclusions regarding the relative proportions of time-to-resolution attributed to each stage are not portable across the entire carrier size distribution. Therefore, operators who wish to obtain accurate estimates of time-to-resolution attributable to each stage should perform separate evaluations at their own scale.

6.5. Considerations That Are Beyond the Scope of This Research

STP pipeline cybersecurity considerations, privacy controls associated with STP pipelines, and disaster recovery postures associated with STP pipelines are important operational issues that were not considered in this research. Therefore, readers

interested in addressing these concerns are referred to a wider body of literature concerning operational engineering in insurance technology.

7. Conclusion

This study investigated the influence of automated Straight Through Processing (STP) on Time to Resolution (TTR) for claims in U.S. life insurance and developed a substrate-resident time to resolution (TTR) decomposition model, which divides the total TTR into the delay caused by an artificial intelligence (AI) decision-making process and the total delay incurred within the substrate. The model was tested on a synthetic 50,000-claim pipeline for life insurance, based upon three levels of automation and two relational substrates. The results show that two stages of the substrate (intake-to-staging replication and decision-to-payment batch window) account for 78% of the remaining full-STP TTR, while the AI decision-making stage accounts for only 4%, contrary to the prevailing view among researchers that STP's effect on TTR is dominated by AI decisions. In practice, the findings suggest that overall TTR reporting conceals the stages in the substrate, where most operational capacity exists. For measurement, it also suggests a sharp distinction: STP effects need to be measured at each stage of the substrate rather than being represented as an average measure; otherwise, two carriers may report having the same TTR, but they will have vastly different substrate configurations and therefore vastly different potential improvements. Regulators are also interested in these findings, since substrate-resident timestamps per stage and feature-level provenance will be much simpler to monitor and reproduce than those of internal decision engines. The model can be applied to both of the two main relational substrates used in U.S. Life Insurance Production Environments. All data required to run the benchmark is available to other researchers as a reproducible synthetic pipeline. Therefore, future comparative studies on the performance of multiple U.S. Carriers using the same substrate-resident definitions will be possible.

References

[1] Eling M, Lehmann M. The Impact of Digitalisation on the Insurance Value Chain and the Insurability of Risks. Geneva Pap Risk Insur Issues Pract. 2018;43(3):359–396. doi:10.1057/s41288-017-0073-0

[2] Eling M, Nuessle D, Staubli J. The impact of artificial intelligence along the insurance value chain and on the insurability of risks. Geneva Pap Risk Insur Issues Pract. 2022;47(2):205–241. doi:10.1057/s41288-020-00201-7

[3] Stoeckli E, Dremel C, Uebernickel F. Exploring characteristics and transformational capabilities of InsurTech innovations to understand insurance value creation in a digital world. Electron Mark. 2018;28(3):287–305. doi:10.1007/s12525-018-0304-7

[4] Riikkinen M, Saarijärvi H, Sarlin P, Lähteenmäki I. Using artificial intelligence to create value in insurance. Int J Bank Mark. 2018;36(6):1145–1168. doi:10.1108/IJBM-01-2017-0015

[5] Aslam F, Hunjra AI, Ftiti Z, Louhichi W, Shams T. Insurance fraud detection: Evidence from artificial intelligence and machine learning. Res Int Bus Finance. 2022;62:101744. doi:10.1016/j.ribaf.2022.101744

[6] Severino MK, Peng Y. Machine learning algorithms for fraud prediction in property insurance: Empirical evidence using real-world microdata. Mach Learn Appl. 2021;5:100074. doi:10.1016/j.mlwa.2021.100074

[7] Badriyah, T., Ariyanto, Y., & others. (2019). Nearest neighbour and statistics method based for detecting fraud in auto insurance. International Journal of Artificial Intelligence and Applications (IJAIA).

[8] Nur Prasasti, I. M., Dhini, A., & Laoh, E. (2020). Automobile insurance fraud detection using supervised classifiers. 2020 International Workshop on Big Data and Information Security (IWBIS 2020), 47–51. https://doi.org/10.1109/IWBIS50925.2020.9255426

[9] Şahin, Ö. E., Ayvaz, S., & Çalımfidan, E. (2020). Comparison of machine learning models developed for the prediction of fraudulent claims in the insurance sector. Journal of Information Technologies, 13(4).

[10] Villar AS, Khan N. Robotic process automation in the banking industry: a case study on Deutsche Bank. J Bank Financ Technol. 2021;5(1):71–86. doi:10.1007/s42786-021-00030-9

[11] Thekkethil MS, Shukla VK, Beena F, Chopra A. Robotic Process Automation in Banking and Finance Sector for Loan Processing and Fraud Detection. In: Proc. 9th International Conference on Reliability, Infocom Technologies and Optimisation (ICRITO). IEEE; 2021. doi:10.1109/ICRITO51393.2021.9596076

[12] Lacity MC, Willcocks LP. A New Approach to Automating Services. MIT Sloan Manag Rev. 2016;58(1):41–49.

[13] Lehnert M, Linhart A, Roeglinger M. Economic Evaluation and Optimization of the Degree of Automation in Insurance Processes. Bus Inf Syst Eng. 2009;1(1):67–73. doi:10.1007/s12599-009-0088-6

[14] van der Aalst WMP. Process Mining: Data Science in Action. 2nd ed. Springer; 2016. doi:10.1007/978-3-662-49851-4

[15] Bucher T, Gericke A. Supporting performance management with business process management and business intelligence: A case analysis of integration and orchestration. Inf Manag. 2013;50(8):491–500. doi:10.1016/j.im.2013.07.008

[16] Lacity M, Willcocks L, Gozman D. Influencing information systems practice: The action principles approach applied to robotic process and cognitive automation. J Inf Technol. 2021;36(3):216–240. doi:10.1177/0268396221990778

[17] Cevolini A, Esposito E. Explaining automated decision-making: a multinational study of the GDPR right to meaningful information. Geneva Pap Risk Insur Issues Pract. 2022;47(4):833–854. doi:10.1057/s41288-022-00271-9

[18] Organisation for Economic Co-operation and Development. (2020). The impact of big data and artificial intelligence (AI) in the insurance sector. OECD Publishing. https://doi.org/10.1787/c822ee53-en