Luther Case Study

Long-term Commercial Real Estate Lease Contract Renewal, end-to-end Process Operations

Luther Systems

1. Executive Summary

A leading investment company (the Investor), with over 1000 employees in several countries, the Investor is a large multinational enterprise. One of the Investors' main operations is managing "real assets", and the Investor is one of the leading real asset managers in Europe. Of their real assets, a significant portion is real estate, and the Investor employs a large team to manage these holdings. The real estate management industry is worth more than $3 trillion globally[1].

The Investor owns commercial properties suitable for use as large retail outlets. Retailers who wish to utilize these spaces for long periods of time sign long-term lease contracts with the Investor. These contracts include periodic updates of the rent values in line with inflation, property values, and other factors. To implement the new rent amount, it is necessary to carry out the rent-renewal process in which the lease contract is updated to reflect the new rent amount. It is necessary to carry out this process efficiently to maximize operational revenue as well as comply with rent regulations. The estimated commercial impact of implementing the Luther Platform are:

ROI	Cost reduction	Avg. Speed increase
8X	70%	9X

A key component of real estate asset management is renewing long-term lease contracts.
This process has the following participants: Aviva Investors, the contract managing agent, the tenant, and the property manager.

The long-term Lease Rent Renewal process has these steps: i) Aviva Investors identify contract managing agent (who manages the contract on behalf of AI), ii) Aviva Investors notify the managing agent via email, iii) Aviva Investors source the updated index value from the Office of National Statistics at the correct date, iv) Aviva Investors internally calculate the new rent value, v) Aviva Investors internally approve and sign off on the updated rent, vi) Aviva Investors' lawyers generate the rent renewal memo and send it to the tenant and property manager (who manages the property itself), vii) tenant accepts and confirms the updated rent, viii) property manager accepts and confirms the updated rent, ix) Aviva Investors finalize the updated rent amount and contract, x) tenant accepts and confirms the updated contract, xi) property manager accepts and confirms the updated contract, xii) payment details between tenant and property manager update with the correct amount.

This process involves 6 teams; i) Investor managing agent team, who handles communications with the managing agent, ii) Investor lease management team, who handle calculating new rent values and updating leases, iii) Investor contracts and memos team, who approve the updated leases and notify the property manager and tenant, iv) property management team, who manages the property itself, v) the tenant, the enterprise that is leasing the property as a commercial outlet, and vi) payments between the tenant and property manager through which rent is paid.

[1] https://www.mckinsey.com/industries/private-capital/our-insights/a-turning-point-for-real-estate-investment-management#/

The Investor manages 18000 leases annually, with an average lease value of £750,000. Luther engaged in the operations of a portion of the portfolio.

Annual process runs	Average lease
18,000	£750000

The Investor operates the Long-term Lease Rent Renewal process as part of the commercial real estate value chain. This process operates across 6 teams & 10 software systems and it includes 56 tasks end-to-end.

Teams	Software systems	Tasks
6	10	56

To operate the process end-to-end, each function performs the same series of steps (Operations Cycle); i) send data & info to the system, ii) receive response from system, iii) compute & validate response, iv) share & store execution of step, v) evaluate & initiate next step.

> The Investor operates the Process across
> 6 teams & 10 Software Systems
>
> each team & system performs a function for the Process (generating memos, review contracts, ...)
>
> these teams & systems are siloed,
> they have separate ops & tech & governance
>
> but the end-to-end Process operates across them

For reliable operations, all teams & systems involved should operate the same end-to-end process. However, they often don't! This leads to operational & technical challenges, which make process operations unreliable. The opportunity is providing a platform to reliably operate the end-to-end process, across all teams & systems involved. Traditional solutions to end-to-end process operations are unreliable & expensive.

Enterprise Operations are generally function-first, which means they continue to focus on improving functions & systems, but processes are considered secondary. The thinking is that if we have great functions & systems, the business can operate any process! Traditionally enterprises use bespoke connectors & local operations scripts for process operations, which are fragmented, siloed, and change separately, and so are ineffective for reliable process operations.

> Enterprises primarily focus on the operations of individual teams & systems,
> and continuously improve them
>
> operations of the end-to-end process
> across 6 teams & 10 systems
> is of secondary focus and often neglected,
> especially as the process evolves
>
> This costs the enterprise
> millions in operational costs, and weeks in delays

To remedy this, enterprises use automation tools. However, they are ineffective at end-to-end process operations, due to their limited scope and scale, and stitching them together also doesn't solve the problem.

> Luther's platform takes a process first approach to process operations
> focusing on the reliable operations of the end-to-end process across all teams & systems,
> instead of cobbling & stitching together the separate & siloed functions of 6 teams & 10 software systems

Luther's platform is designed process-first, & primarily focuses on end-to-end processes. Reliable end-to-end process operations include consistent operations, and great functions & systems.

The automated Long-term Commercial Real Estate Contract Rent Renewal Process, built on the Luther Platform, uses Deep Process Automation Technology to automate the rent renewal of long-term lease contracts. The Luther Platform provides standard connectivity and a Common Operations Script shared by all participants. The platform reliably operates the end-to-end process across all teams and software systems from the common operations script.

Luther's unique value for reliable end-to-end Process Operations is providing i) standard connectivity & ii) a common operations script, across all teams & software systems.

Luther's unique value for reliable end-to-end Process Operations is providing

standard connectivity
a common operations script

across all teams and software systems.

Luther's platform vertically integrates i) distributed system technology ii) optimal resource allocation & management, iii) real time event ordering & streaming, iv) deterministic event processing & execution, for reliable end-to-end process operations.

Luther's platform vertically integrates

distributed system technology
optimal resource management
real time event streaming
deterministic event execution

To make reliable end-to-end process operations possible.

Luther's platform does this by i) connecting systems to standard platform nodes, rather than to each other, and ii) teams & systems can change the common operations script but all teams & systems have to know & agree to the change, so all teams & systems involved operate the same end-to-end Process all the time!

Finally the Luther Platform reliably operates the end-to-end Long Term Lease Rent Renewal process across 6 teams, 10 software systems & 56 tasks.

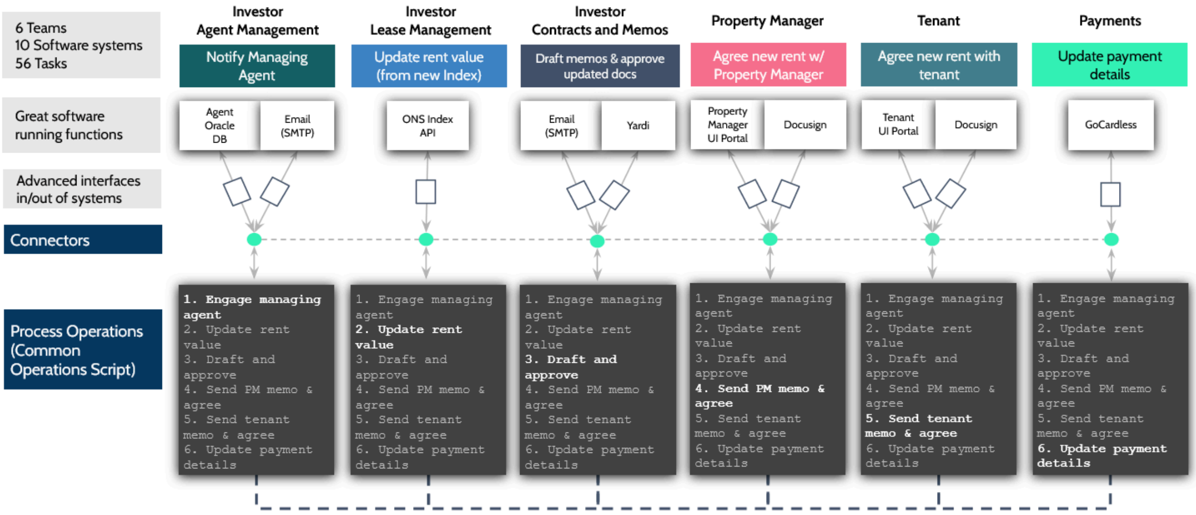
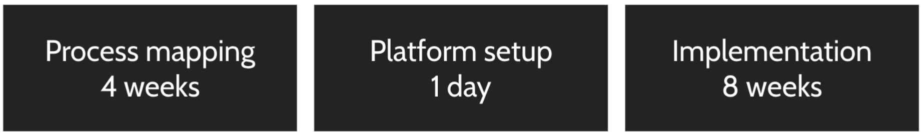

6 Teams
10 Software systems
56 Tasks
Great software running functions
Advanced interfaces in/out of systems
Connectors
Process Operations (Common Operations Script)
Investor Agent Management
Notify Managing Agent
Agent Oracle DB
Email (SMTP)
Investor Lease Management
Update rent value (from new Index)
ONS Index API
Investor Contracts and Memos
Draft memos & approve updated docs
Email (SMTP)
Yardi
Property Manager
Agree new rent w/ Property Manager
Property Manager UI Portal
Docusign
Tenant
Agree new rent with tenant
Tenant UI Portal
Docusign
Payments
Update payment details
GoCardless
1. Engage managing agent
2. Update rent value
3. Draft and approve
4. Send PM memo & agree
5. Send tenant memo & agree
6. Update payment details

Fig 1. Long-term Commercial Real Estate Lease Rent Renewal process operations built on the Luther Platform

To implement the platform, i) Luther's team mapped the Process, ii) Identified teams & software systems in the process, iii) allocated nodes (servers) to teams, iv) connected nodes to systems, v) set up the Platform on the nodes. vi) the Investors' team along with Luther's team developed the Common Operations Script (code) for Process Operations, vii) the process went Live.


Process mapping
4 weeks

Platform setup
1 day

Implementation
8 weeks

Fig 2. Implementation timeline of the automated Long-term Commercial Real Estate Contract Rent Renewal Process.

The results have been highly impactful. Thanks to increases in efficiency and operations reliability, a process that traditionally took nearly a week can now be completed in less than one day and operational costs have been reduced by 70%. Beyond the commercial results, this led to operational benefits in production; i) reliable operations across the end-to-end process & over time, ii) 5X smaller Ops teams, iii) real-time monitoring, iv) enforced compliance checks, v) real-time and consistent updates across all teams, vi) reduction of reconciliation. Also, technical benefits during development; i) standard dev. process so developers can focus on operations, ii) 5X smaller Dev teams, iii) standard process connectors, iv) automated infrastructure and connectors setup, v) real-time and consistent updates with the rest of operations.

Operational Advantages in production	Technical Advantages during development
Consistent Ops: across the process & over time	Standard development process
5X smaller Ops teams	Automated infra., connectors, dev. env. setup
Real-time monitoring	Focus only on development of process Ops
Enforced compliance checks	Real-time & consistent updates across operations
Real-time & consistent updates across teams	Eliminate DevOps teams, 5X smaller Dev teams
Reduction of reconciliation	Standard process connectors

The automated Long-term Commercial Real Estate Contract Rent Renewal Process demonstrates a reliable, standardized, and effective system built on the Luther platform to standardize and automate the long-term lease contract rent renewal process. The platform could be further expanded to include the entire property portfolio of the Investor, and the Luther platform could also be utilized to further streamline other areas of the Investors' operations, for example using automation to more effectively resolve rent disputes.

ROI	Cost reduction	Avg. Speed increase
8X	70%	9X

Fig 3. Estimated results of implementing the Platform for the Long-term Lease Contract Renewal Process

2. The Process

2.1. Process Operations

Different teams have different operations, rules and governance and they also utilize and operate a variety of software systems in different ways. Each system operates a specific function for the process. To operate the process end-to-end, each function performs the same cycle of steps: i) send data & information to the System, ii) receive response from the System, iii) compute & validate response, iv) share & store execution of step, v) evaluate & initiate next steps.

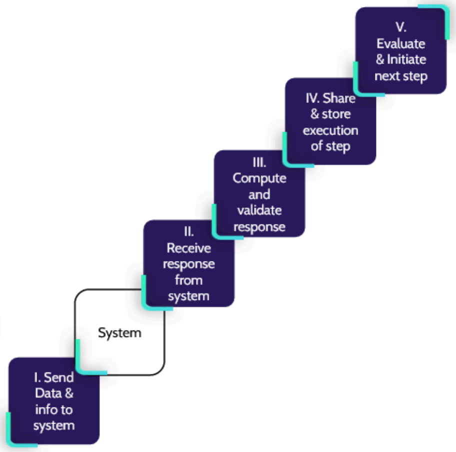

I. Send Data & info to system
System
II. Receive response from system
III. Compute and validate response
IV. Share & store execution of step
V. Evaluate & Initiate next step

Fig 4. These are the requirements that repeat for all functions across the end-to-end Process Operations.

Enterprises operate a set of specific functions based on their objective. For example, an investment management company's functions help it to manage investment portfolios. While the functions and systems may change, the process remains the same. However, expecting processes to be efficient because of efficient individual tools simply does not work for enterprises. Luther empowers enterprises with a process-first approach.

Tasks are simple events that are localized to one team involving one or two software systems, for example retrieving data from a database. Workflows are more complex, involving 10-20 tasks between one to two teams, and two to three software systems. An example of a workflow is onboarding a new employee. Processes are complex, involving 50+ tasks, 3 or more teams and multiple software systems. Renewing the rent of a long-term lease contract end-to-end is a process.

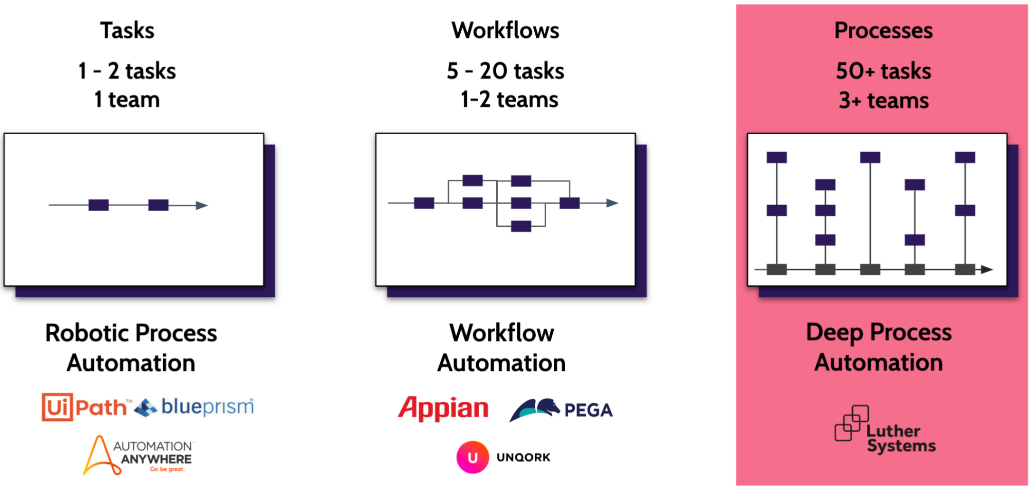


Fig 5. Different tools are used to automate different levels of complexity.

2.2. Function First Operations and its limitations

Generally, enterprise operations are function-driven. They have a large collection of software systems each operating a specific task. Tasks often have dedicated software systems and are operated by specific teams. By developing, purchasing and maintaining efficient systems, most enterprise tasks operate highly reliably.

Enterprise processes, however, operate across multiple teams and software systems, and involve many tasks. This means reliable end-to-end process operations require efficient teams and systems, as well as efficient connectivity and operations across these teams and systems.

Enterprises generally take a “function-first” approach to process operations. Great individual teams and systems provide the required ingredients for great process operations, so they focus on enhancing and improving the performance and efficiency of individual teams and software systems. A good analogy of this approach is “if we have great ingredients, anyone can cook anything they want and it’ll be of great quality!” Processes are considered secondary to functions and systems, as they are considered ever changing, and efficient functions and systems can enable any process that the business may envision. The problem is, efficient functions do not necessarily create an efficient process.

Efficient software systems and functions are not enough to reliably operate a process end-to-end

In practice, most enterprises have a defined charter and mission, particularly if they are in a regulated industry. They provide specific products and services which are generally enumerated and these rarely change. These form the basis of the value streams provided by an enterprise. For example, every real estate investment company researches opportunities, invests in real estate property, manages the property, manages the proceeds and divests from the property. The majority of "enterprise operations" are in operating these value streams. Each value stream has a set of processes, which are generally enumerated and these rarely change. The details might vary over time but the process functions remain the same. For example the property management value stream includes these processes: property acquisition, property management, risk management, tenant onboarding, tenant management, rent collection, contracts, and rent review. These are well known processes with well known functions, the details and data in these processes might change over time, however the functions of these processes remain the same.

The majority of processes and their functions (what each process does end-to-end) are enumerable for an enterprise. In fact a large deviation from these processes and venturing into new areas that are drastically different from the enumerated processes within an enterprise is a major event at an enterprise and is a multi-year plan. The vast majority of enterprise processes (what the process does) are enumerable and remain largely the same.

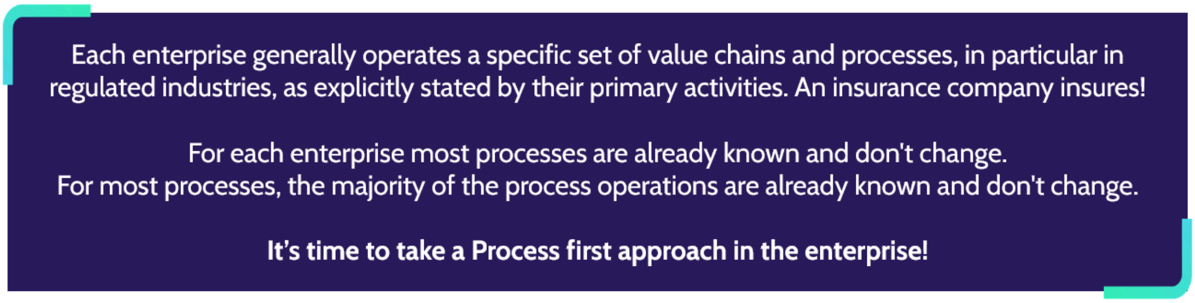
The prevailing view is if we build or purchase efficient teams and systems, then any process can be built on top of these great teams and systems. Processes are secondary to these functions and systems, as they are considered ever changing, and functions and systems are there to enable any process that the business may envision or desire to build!

> "If we have great functions, services and systems, we're enabling the business to build and operate any process they want!"

Enterprises continue to optimize and improve, and incorporate better functions and systems. Example functions include contract signing, rent calculations, customer onboarding, finance, payments, settlement, fraud, compliance, reconciliation. Example software systems include databases, CRMs, RPA, Workflow tools, cloud services, microservices, data lakes, and others.

The problem is i) enumerated processes are not flexible enough for this function-first approach, ii) efficient teams and systems are not enough to build efficient end-to-end processes.

For enterprise operations the process and its function (end-to-end operations) are equally as important as the individual teams and systems and their individual technology and functions (what they each do).



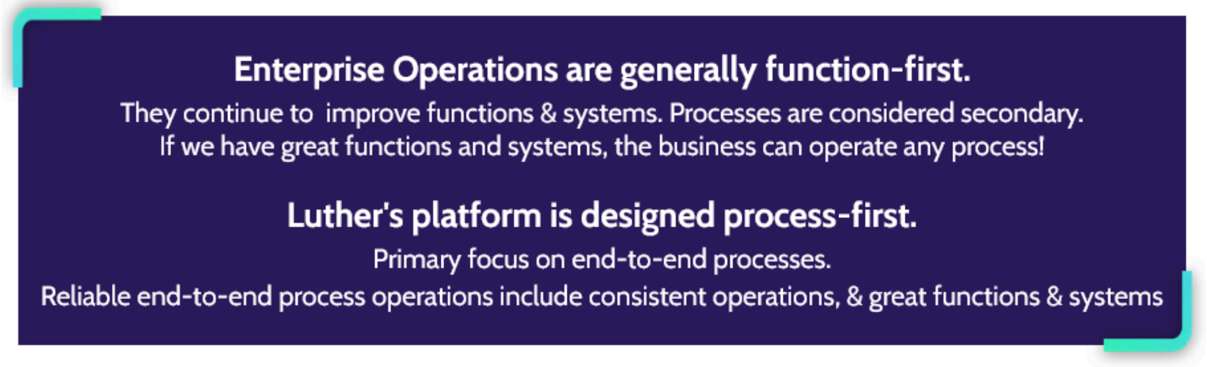
Each enterprise generally operates a specific set of value chains and processes, in particular in regulated industries, as explicitly stated by their primary activities. An insurance company insures!

For each enterprise most processes are already known and don't change.
For most processes, the majority of the process operations are already known and don't change.

It's time to take a Process first approach in the enterprise!

2.3. Process First Operations

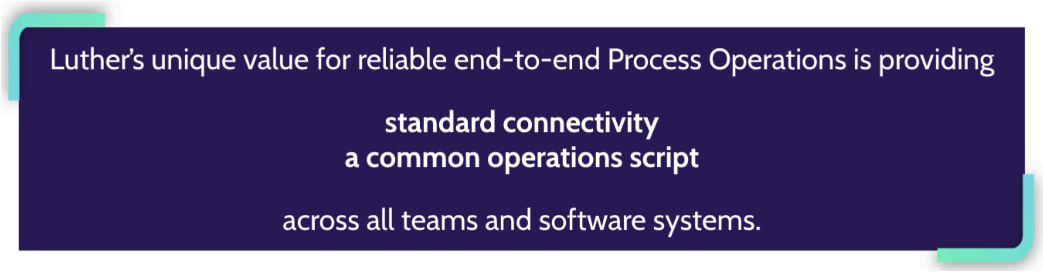
Luther's platform is designed process-first. For efficient enterprise operations, effective end-to-end operations are as important as effective individual services and teams and systems, primarily since the enterprise's core value is delivering a specific set of processes and value streams, particularly in regulated industries, where most value streams & processes are explicitly enumerated!



Enterprise Operations are generally function-first.
They continue to improve functions & systems. Processes are considered secondary.
If we have great functions and systems, the business can operate any process!

Luther's platform is designed process-first.
Primary focus on end-to-end processes.
Reliable end-to-end process operations include consistent operations, & great functions & systems

The most important attributes of process first operations are i) standardized connectivity between all systems involved in the process, ii) Common Operations Script operating the end-to-end process.



Luther's unique value for reliable end-to-end Process Operations is providing

standard connectivity
a common operations script

across all teams and software systems.

2.4. Long-term Lease Contract Rent Renewal process in context

The Investor has a number of general Value Streams involved in commercial real estate investment. One such value stream is “commercial real estate management”[2]. This value stream includes multiple processes. Periodically renewing the rent of long-term leases is a key process for enterprises involved in real estate asset management. Its primary purpose is to ensure rents reflect current index values, and that rent payments are timely and efficient, maximizing operational revenue and following rent regulation compliance guidelines.

[2] https://www.lev.co/blog/value-chain

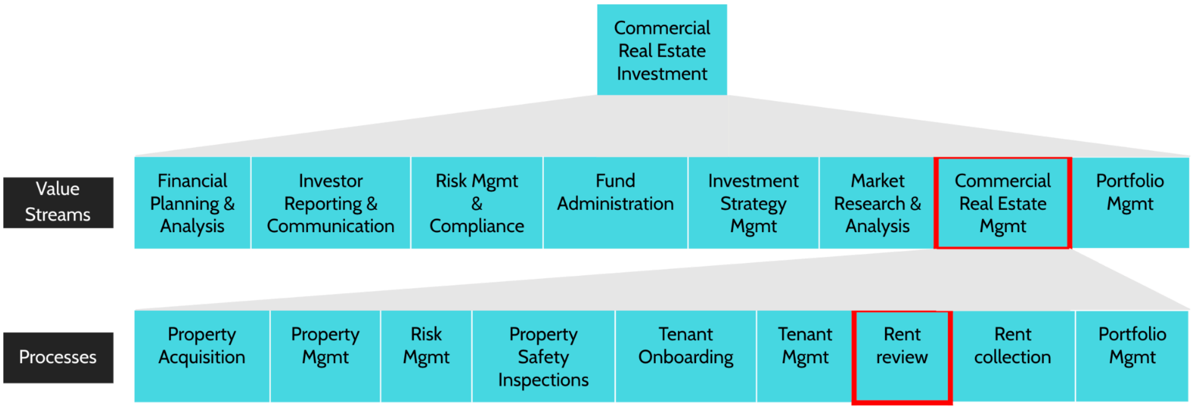


Fig 6. The Investor operates many value streams as part of commercial real estate investment. Each value stream contains many processes. Commercial property management is a value stream. It contains many processes, including rent renewal.

Rent renewal for long-term leases typically occurs every 3-5 years[3]. The commercial real estate investment management in the UK is around 18 billion, contributing around 1% of the UK's gross value added[4]. The real estate investment industry is worth more than $3 trillion globally[5], and this value is due to the utility provided to property managers and tenants. Consequently, it is critical to operate an efficient and accurate rent-renewal process to effectively manage commercial real estate investments. It results in fast turnaround times for contract updates, which leads to better relations with tenants, which is crucial as shifting work patterns and high interest rates threaten enterprises managing commercial real estate[6].

Fast turnaround times for contract updates

Better relations with tenants

Stability for property managers

The Investor manages many rental properties that require periodic rent renewals. As the world becomes increasingly digitized, companies look to streamline their operations with efficient digital processes to automate administrative tasks. Currently, much of the Investors' rent renewal process does operate digitally.

They employ large teams to operate this process and carry out the many instances of manual intervention required when operated with traditional methods. However, owing to the large number of participants involved and the need for consistent operations across the participants involved, current automation solutions have been ineffective at reliably automating the operations of this process.

[3] https://www.solegal.co.uk/insights/commercial-leases-short-term-versus-long-term-options

[4] https://www.rics.org/content/dam/ricsglobal/documents/market-surveys/UK_commercial_real_Estate_Impact_Report.pdf

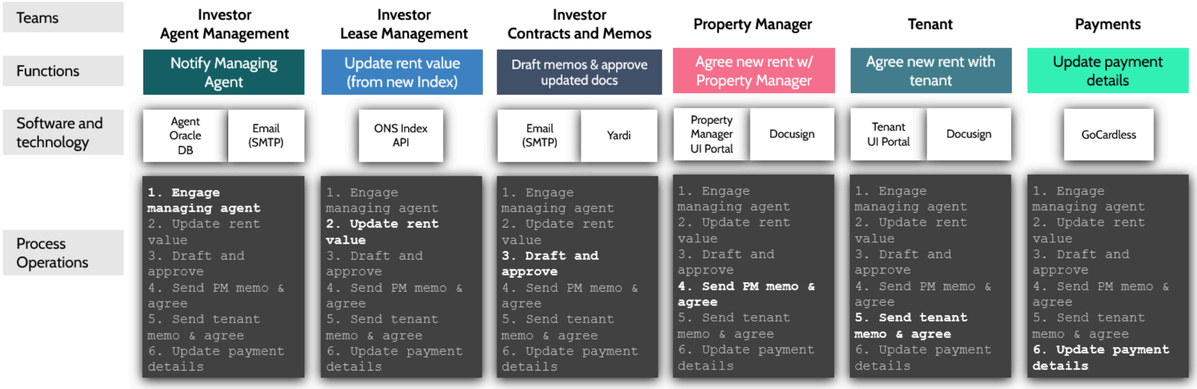
[5] https://www.mckinsey.com/industries/private-capital/our-insights/a-turning-point-for-real-estate-investment-management

[6] https://www2.deloitte.com/us/en/insights/industry/financial-services/commercial-real-estate-outlook.html

2.5. Long-term Lease Contract Rent renewal Process before

The Investor owns a property that a major retailer seeks to lease for retail purposes over a long-term period, potentially spanning up to 30 years. Initially, the retailer consents to a set rent amount payable to the Investor. However, this rent amount is subject to periodic adjustments, often annually, based on a predetermined economic index. Examples of such indices include the Consumer Price Index (CPI) or the House Price Index (HPI), among others. The agreed-upon index serves as a key input for recalculating the updated rent, which is then communicated to both the tenant and the property manager. Following this, the contract is updated through an addendum, which must be reviewed and signed by all involved parties. The updated rent amount is subsequently collected via the payment systems. This process of recalculation, communication, and contract amendment is repeated periodically throughout the entire term of the lease agreement. Below is the process of renewing a lease contract and the participants involved. The process is based on the rent renewal date, the date on which a rent renewal is initiated.

1. The managing agent associated with the contract is identified
2. The relevant managing agent is notified via email
3. The updated index is sourced from the Office of National Statistics at the correct date
4. The Investor internally calculates the new rent value
5. The Investor internally signs off and approve on the updated rent
6. The Investors' lawyers auto-populate a rent renewal memo, generating a document to send to the tenant and property manager
7. The Investor notifies the tenant and property manager via Yardi
8. The tenant accepts and confirms the updated rent
9. The property manager accepts and confirms the updated rent
10. The Investor finalizes the updated rent amount and contract
11. The tenant accepts and confirms the updated contract
12. The property manager accepts and confirms the updated contract
13. The tenant payment details updates with the correct amount



Teams	Investor Agent Management	Investor Lease Management	Investor Contracts and Memos	Property Manager	Tenant	Payments
Functions	Notify Managing Agent	Update rent value (from new Index)	Draft memos & approve updated docs	Agree new rent w/ Property Manager	Agree new rent with tenant	Update payment details
Software and technology	Agent Oracle DB Email (SMTP)	ONS Index API	Email (SMTP) Yardi	Property Manager UI Portal Docusign	Tenant UI Portal Docusign	GoCardless
Process Operations	**1. Engage managing agent** 2. Update rent value 3. Draft and approve 4. Send PM memo & agree 5. Send tenant memo & agree 6. Update payment details	1. Engage managing agent **2. Update rent value** 3. Draft and approve 4. Send PM memo & agree 5. Send tenant memo & agree 6. Update payment details	1. Engage managing agent 2. Update rent value **3. Draft and approve** 4. Send PM memo & agree 5. Send tenant memo & agree 6. Update payment details	1. Engage managing agent 2. Update rent value 3. Draft and approve **4. Send PM memo & agree** 5. Send tenant memo & agree 6. Update payment details	1. Engage managing agent 2. Update rent value 3. Draft and approve 4. Send PM memo & agree **5. Send tenant memo & agree** 6. Update payment details	1. Engage managing agent 2. Update rent value 3. Draft and approve 4. Send PM memo & agree 5. Send tenant memo & agree **6. Update payment details**

Fig 7. Illustrates the process of renewing long-term leases and the participants and systems involved.

3. Problem

3.1. Problem Overview

For reliable process operations, all teams and systems involved should operate the same end-to-end Process.

They often don't!

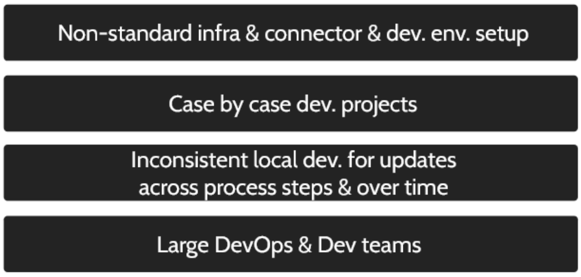
Enterprises are complex organizations operating many processes. Enterprises operate processes across fragmented and siloed teams and software systems resulting in disjointed, inconsistent, inefficient end-to-end operations which lead to high costs, delays and errors. The Investors' long-term commercial real estate lease contract rent renewal process runs across multiple teams and software systems, it requires approval by multiple participants internally, as well as the tenant and the landlord. The requirement for manual intervention by multiple parties at multiple stages increases the timescale of the process, and the manual generation of new contracts makes the process lengthy and error-prone, and ultimately not reliable.

3.2. Enterprise Process Operations Problems

Specifically, operating processes across fragmented and siloed teams and software systems affect process operations both i) technically during the development phase and ii) operationally once they go live in production.

On the technical side, for process changes, enterprises set up case-by-case projects, which includes large development and DevOps teams, and setup of nonstandard case-by-case infrastructure and development environments, as well as bespoke connectors between different systems. Further, as the teams and systems change over time they deploy local updates which usually impact the end-to-end operations, requiring further updates and patching.

Technical hurdles in development



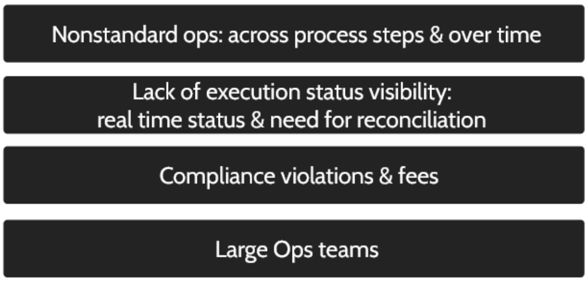
Non-standard infra & connector & dev. env. setup

Case by case dev. projects

Inconsistent local dev. for updates across process steps & over time

Large DevOps & Dev teams

Operational problems in production (live)



Nonstandard ops: across process steps & over time

Lack of execution status visibility: real time status & need for reconciliation

Compliance violations & fees

Large Ops teams

Once the process is live, the fragmented and separated teams and systems result in non-standard operations across the process and over time as the teams, operations, and systems change. The fragmentation also results in a lack of execution visibility and operations monitoring. This further results in the execution requiring reconciliation, which is often lengthy and expensive. This could also result in compliance issues and violations. All of this requires large operations teams to run the processes and fix their recurring issues.

3.3. Long-term Lease Contract Rent renewal Process Operations Problems

The current rent renewal process is fragmented. Each lease contract has its own individual rent renewal dates and indexations, and individual tenants and property managers that need to be informed of the changes during the process. New indexations are manually updated using a variety of data sources and require manual intervention from multiple participants including the Investor. New contracts need to be manually generated for finalization. These manual checks and disjointed systems result not only in errors, but also extended time periods over which these changes are made, and the reduced ability to efficiently scale this product offering to more customers. Duplicative processes increase the costs to perform this rent renewal function as well as increasing the tendency for errors and delays.

Ultimately, the need for large amounts of manual intervention and contract generation result in higher costs, increases the chance errors occur, and results in longer timeframes for the process.

4. Traditional approaches to process operations and automation solutions don't work

4.1. Approach to Process Operations today

Enterprises typically establish dedicated projects and project teams to set up process operations. This involves mobilizing large development and DevOps teams, as well as large operations and support teams. They create custom, often non-standard project infrastructure, connectors, and development environments, which require dedicated ongoing maintenance once the process is live. The project team writes bespoke operations code to manage the end-to-end process, including code that links the operations of various software systems.

As the process moves into production, developers must continuously write custom local code to adapt to the evolving landscape of team operations, process rules, and software systems. Additionally, the project team or other development teams need to develop and integrate separate execution monitoring software and reconciliation software. These tools are essential for detecting errors and inconsistencies, determining root causes, and correcting the issues. Furthermore, they deploy multiple distinct application systems, such as compliance software systems, to support the overall operation.

This demonstrates the bespoke, fragmented nature of process operations development, in addition to multiple auxiliary systems required to keep the operations going. Most importantly, this approach cannot keep pace with the ever-changing process operations.

Enterprise process operations are unreliable!

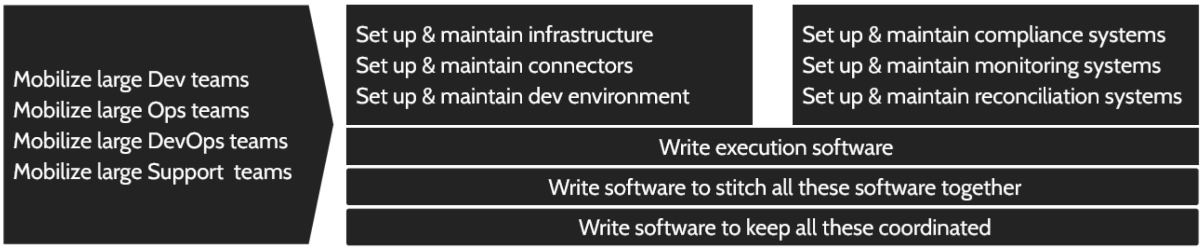

Mobilize large Dev teams
Mobilize large Ops teams
Mobilize large DevOps teams
Mobilize large Support teams
Set up & maintain infrastructure
Set up & maintain connectors
Set up & maintain dev environment
Set up & maintain compliance systems
Set up & maintain monitoring systems
Set up & maintain reconciliation systems
Write execution software
Write software to stitch all these software together
Write software to keep all these coordinated

Fig 8. Enterprises generally carry out all of the above to run a process.

4.2. Bespoke Connectors & Operations Scripts & why they don't work

To manage the long-term lease contract rent renewal process, the Investor typically; i) sets up local connectors between directly linked systems involved in the process, ii) Develops and updates local operations scripts to manage the process end-to-end. Both the bespoke connectors and operations scripts require regular updates and modifications as teams, process operations, and software systems evolve. These updates are reactive and localized, addressing immediate changes without fully considering the entire process.

The problem arises because these connectors and scripts are integral to the end-to-end process, where each step depends on others and assumes specific functions from other parts. Local changes alter the immediate local operations, but the rest of the process continues to rely on outdated assumptions about those functions. This results in a gradual drift and fragmentation between different parts of the process.

This drift and fragmentation requires further patches and updates, which will require further patches and updates in other parts of the process, and the cycle continues!

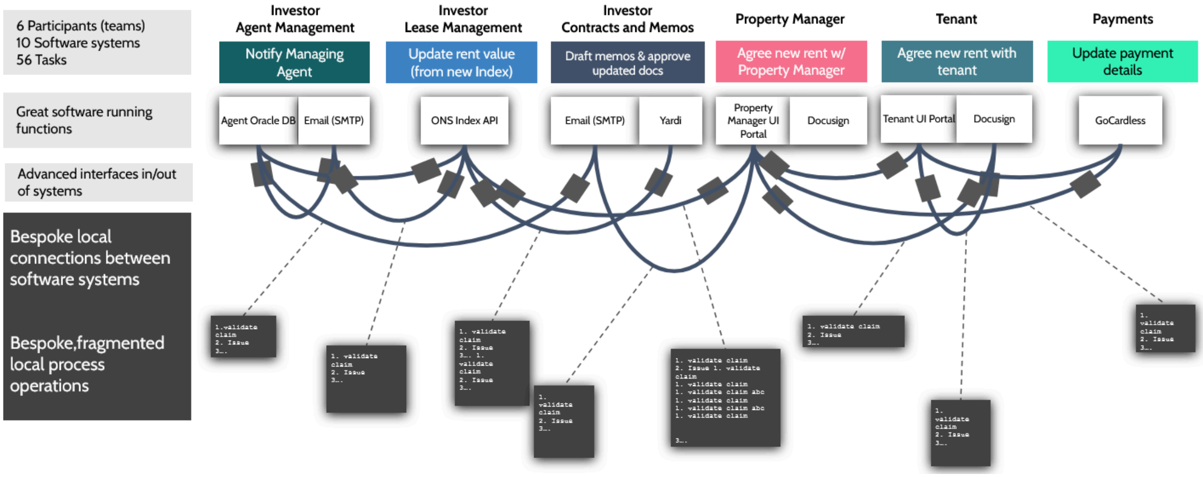

6 Participants (teams)
10 Software systems
56 Tasks
Great software running functions
Advanced interfaces in/out of systems
Bespoke local connections between software systems
Bespoke,fragmented local process operations
Investor Agent Management
Notify Managing Agent
Investor Lease Management
Update rent value (from new Index)
Investor Contracts and Memos
Draft memos & approve updated docs
Property Manager
Agree new rent w/ Property Manager
Tenant
Agree new rent with tenant
Payments
Update payment details
Agent Oracle DB
Email (SMTP)
ONS Index API
Email (SMTP)
Yardi
Property Manager UI Portal
Docusign
Tenant UI Portal
Docusign
GoCardless

Fig 9. Bespoke local connections across the end-to-end process that are internally developed by the enterprise.

4.3. Local Automation (RPA, Workflow) tools & why stitching them together doesn't work

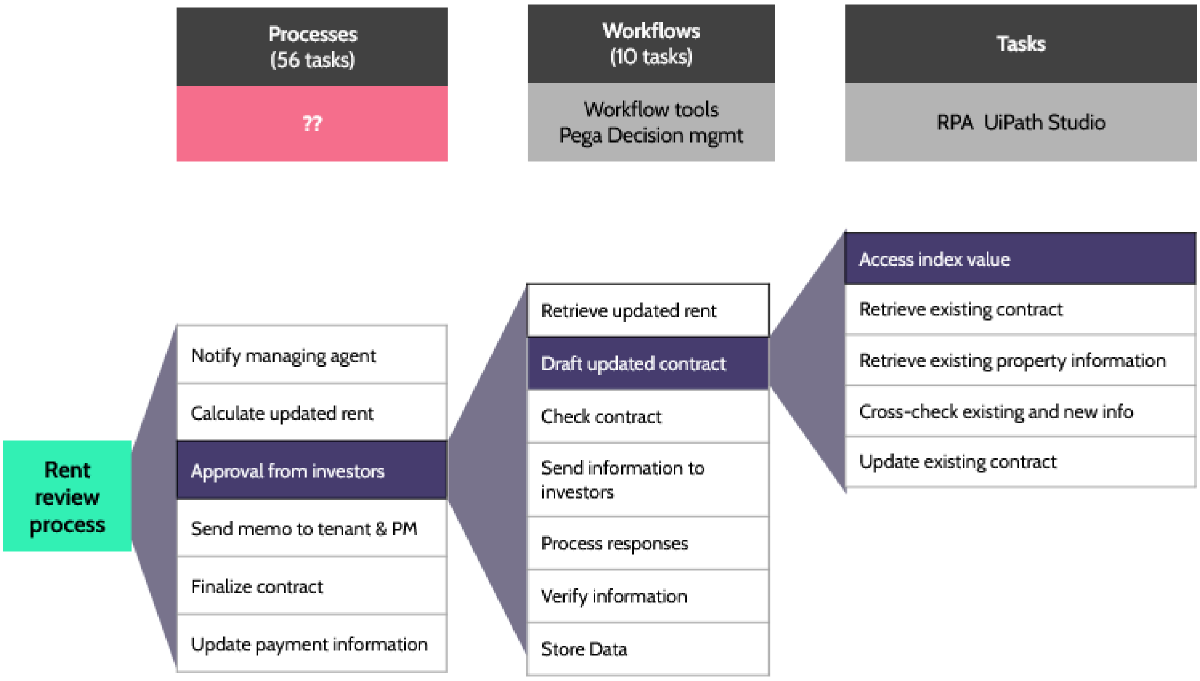


Fig 10. Today, there are no traditional tools which effectively automate processes.

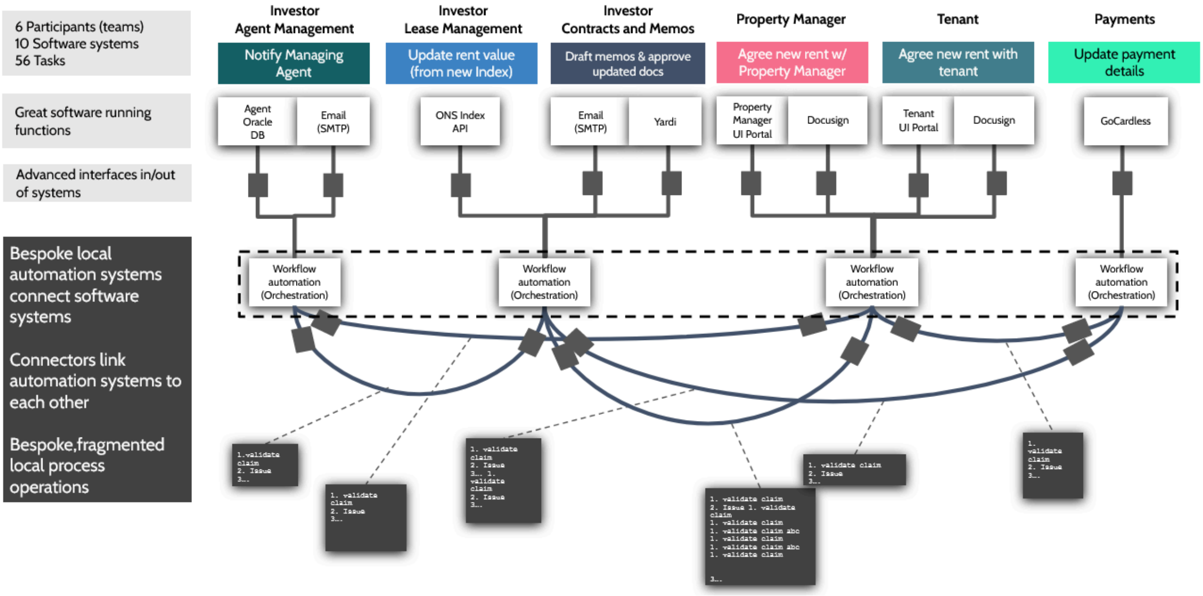
Enterprise processes consist of numerous operations (tasks). Each process includes a collection of workflows, and each workflow is a collection of multiple tasks. Tasks are simple, localized events involving one team and one or two software systems. For example, retrieving an index value is a task. Workflows are more complex, comprising 10-20 tasks that span one to two teams and involve two to three software systems. For instance, drafting a fully updated contract is a workflow consisting of 12 tasks. Processes are complex, involving over 50 tasks, three or more teams, and multiple software systems. An example of a process is the end-to-end renewal of a long-term lease contract, which includes 56 tasks.

Enterprises utilize Robotic Process Automation (RPA) tools to automate individual tasks. RPA tools have evolved into highly effective solutions for this purpose. However, for automating workflows (comprising 10-20 tasks), enterprises turn to Workflow Automation tools, as individual RPA bots are not scalable to handle such complexity. Workflow Automation tools have similarly advanced, becoming highly effective at automating entire workflows. These tools leverage a diverse array of technologies, including traditional ones like Workflow tools, ERPs, and BPMs, as well as modern innovations such as Hyper Automation, Intelligent Automation, and various developer tools.

RPA tools and Workflow tools don't scale to end-to-end Processes!

To overcome the limitations of the traditional approach, enterprises deploy numerous RPA and Workflow tools across the end-to-end process, and then connect and orchestrate these tools to function reliably. This integration and coordination are typically developed internally by the enterprise.

Process orchestration approaches integrate combinations of RPA and workflow systems using point-to-point message passing techniques. These services often employ a batch scheduler or workflow system, which effectively coordinates tasks within a single team. However, this method falls short for processes involving multiple teams. Each team tends to create bespoke code for their tasks, leading to “script bloat” — the proliferation of numerous, often redundant, and poorly documented scripts. This complicates maintenance and scalability. Furthermore, there is a lack of transparency between participants in the process. This lack of coordination and integration results in inefficiencies and errors, causing delays and operational friction. For a full explanation of traditional process operations and Luther’s solution, request access to the Deep Process Automation Primer.



6 Participants (teams)
10 Software systems
56 Tasks
Great software running functions
Advanced interfaces in/out of systems
Bespoke local automation systems connect software systems
Connectors link automation systems to each other
Bespoke,fragmented local process operations
Investor Agent Management
Notify Managing Agent
Investor Lease Management
Update rent value (from new Index)
Investor Contracts and Memos
Draft memos & approve updated docs
Property Manager
Agree new rent w/ Property Manager
Tenant
Agree new rent with tenant
Payments
Update payment details
Agent Oracle DB
Email (SMTP)
ONS Index API
Email (SMTP)
Yardi
Property Manager UI Portal
Docusign
Tenant UI Portal
Docusign
GoCardless
Workflow automation (Orchestration)
Workflow automation (Orchestration)
Workflow automation (Orchestration)
Workflow automation (Orchestration)
1.validate claim 2. Issue 3...
1. validate claim 2. Issue 3...
1. validate claim 2. Issue 3... 1. validate claim 2. Issue 3...
1. validate claim 2. Issue 1. validate claim 1. validate claim 1. validate claim abc 1. validate claim 1. validate claim abc 1. validate claim 3...
1. validate claim 2. Issue 3...
1. validate claim 2. Issue 3...

Fig 11. Stitching together local automation tools through local RPA and workflow tools is messy, localized and ultimately unreliable.

5. Solution

5.1. Luther Platform

Luther's Platform was used by the Investors' development team to build an exemplary long-term lease contract rent renewal system. This required a system that could effectively handle the operations of multiple teams, software systems, numerous tasks and validations end-to-end.

> Luther's unique value for reliable end-to-end Process Operations is providing
>
> **standard connectivity**
> **a common operations script**
>
> across all teams and software systems.

This is very difficult and costly with traditional automation tools and workflows. Automation of the rent-renewal process requires Luther's Deep Automation Platform. The automated Long-term Commercial Real Estate Contract Rent Renewal Process is the result of this work and is an end-to-end rent renewal system that standardizes the process with minimal manual intervention while reducing rent renewal times and eliminating errors from manual input, thereby lowering operational costs.

5.2. How it works on the Luther Platform

The Long-term Lease Contract Rent Renewal process:

- The managing agent associated with the contract is identified
- The relevant managing agent is notified via email
- The new index value is sourced from the Office of National Statistics at the correct date
- The platform internally calculates the new rent value
- The Investor internally signs off and approve on the updated rent
- A rent renewal memo is auto-populated, generating a PDF to send to the tenant and property manager
- The PDF is sent to the tenant and property manager via Yardi
- The tenant accepts and confirms the updated rent
- The property manager accepts and confirms the updated rent
- The final contract is generated and sent to the tenant and property manager
- The tenant accepts and confirms the updated contract
- The property manager accepts and confirms the updated contract
- The payment details between tenant & property manager updates to correct amount

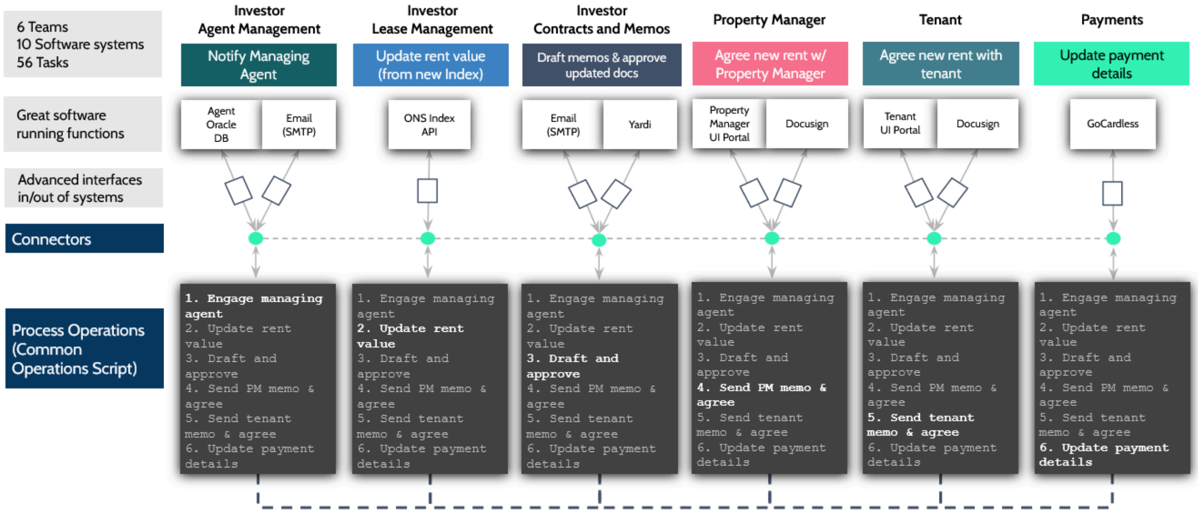

6 Teams
10 Software systems
56 Tasks
Great software running functions
Advanced interfaces in/out of systems
Connectors
Process Operations (Common Operations Script)
Investor Agent Management
Notify Managing Agent
Agent Oracle DB
Email (SMTP)
Investor Lease Management
Update rent value (from new Index)
ONS Index API
Investor Contracts and Memos
Draft memos & approve updated docs
Email (SMTP)
Yardi
Property Manager
Agree new rent w/ Property Manager
Property Manager UI Portal
Docusign
Tenant
Agree new rent with tenant
Tenant UI Portal
Docusign
Payments
Update payment details
GoCardless
1. Engage managing agent
2. Update rent value
3. Draft and approve
4. Send PM memo & agree
5. Send tenant memo & agree
6. Update payment details

Fig 12. Overview of the Luther Platform automating the rent renewal process.

For a more detailed view of the steps operating the Luther Platform, please view the appendix.

6. Implementation

Luther's team worked with the Investor's team to implement The automated Long-term Commercial Real Estate Contract Rent Renewal Process on the platform.

First, Luther's team worked with the teams at the Investor to map the process. View a process map here. Luther then identified all teams and all software systems involved in the operations of the process. Luther then allocated a node to each team, deployed the platform on all nodes, and connected the nodes to each of the software systems, through Luther's standard connectors. Then Luther's team worked with the Investor's developers to develop a robust common operations script for process operations. Then the application went live.

For more information please visit these links, for implementation steps, implementation in general, and sandbox.

Customer Team		*Business Owner, Application Owner, Technical Lead*	*Day 1*
Discover	*Phase 1*	*Describe process operations*	*4 weeks*
	Phase 2	*Describe systems & technical requirements*	
Process mapping		*Map the process*	*1 week*
Platform set-up		*One-time platform set-up*	*1 day*
Build application		*Develop (code) application operations*	*8 weeks*

Fig 13. Implementation timeline for the automated Commercial Real Estate Contract Rent Renewal Process.

To implement the rent renewal process, Luther and the Investor followed these steps:

6.1. Process mapping

Luther's team worked with multiple Investor teams to map the process operations. The process map includes i) functions, ii) data inputs and outputs at each step, and iii) rules and decisions at each step. Teams are operationally separate entities involved in the process. As part of process mapping, Luther identified the exact set of software systems and teams involved in operating the end-to-end process.

6.2. Identify teams and software systems

Luther's team identified the teams and participants involved in end-to-end process operations. These teams are: Investor Teams: Agent Management, Lease Management, Contracts and Memos. The other participants are the Property Manager, the Tenant, and Payments.

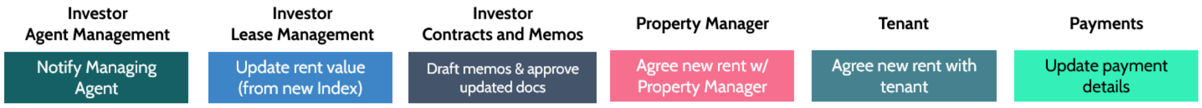

Investor Agent Management
Notify Managing Agent
Investor Lease Management
Update rent value (from new Index)
Investor Contracts and Memos
Draft memos & approve updated docs
Property Manager
Agree new rent w/ Property Manager
Tenant
Agree new rent with tenant
Payments
Update payment details

Fig 14. Luther's team worked with the Investor to map the process including 6 teams involved in end-to-end operations.

Luther's team identified the software systems involved in end-to-end process operations. These systems are: Agent Oracle Database, Email (SMTP), ONS Index API, Yardi, Property Manager UI Portal, Tenant UI Portal, Docusign, and GoCardless.

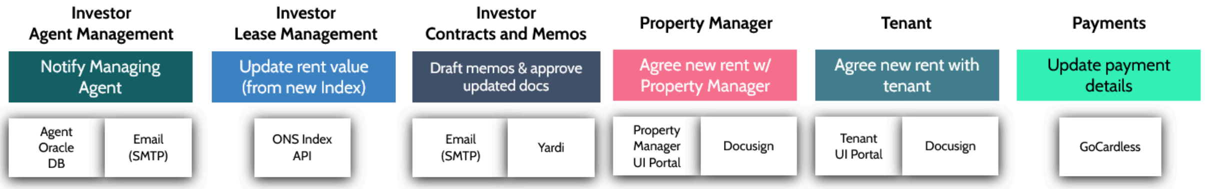

Investor Agent Management
Notify Managing Agent
Agent Oracle DB
Email (SMTP)
Investor Lease Management
Update rent value (from new Index)
ONS Index API
Investor Contracts and Memos
Draft memos & approve updated docs
Email (SMTP)
Yardi
Property Manager
Agree new rent w/ Property Manager
Property Manager UI Portal
Docusign
Tenant
Agree new rent with tenant
Tenant UI Portal
Docusign
Payments
Update payment details
GoCardless

Fig 15. Luther's team identified the software systems involved in the end-to-end process operations.

6.3. Nodes and Connectivity through distributed system for end-to-end team connectivity

Luther's team assigned a dedicated node to each team involved in the process by allocating servers to their respective teams. These servers are cloud-native and can be deployed on either public or private clouds, depending on security requirements. All nodes are interconnected through a distributed system, which facilitates the sharing and validation of operational functions and data among all teams.

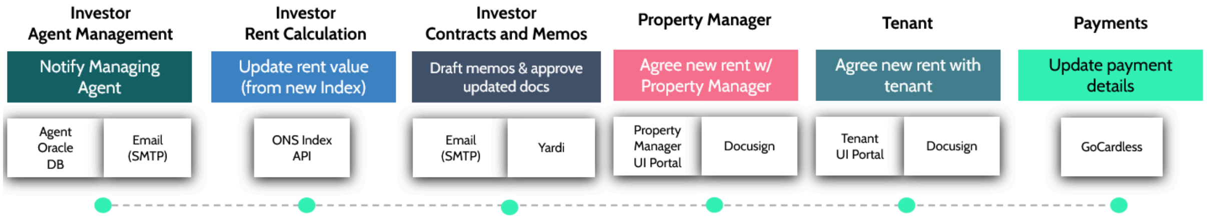


Fig 16. Nodes are connected via a distributed system on the Luther Platform.

6.4. Connectors to software systems

Each team has a number of software systems involved in its operations, as identified in the process map. For each team, Luther's platform connects its node to all software systems involved in its operations. Luther has a set of standard connectors across a wide range of enterprise systems, which the Luther platform deploys to rapidly connect to the systems involved in operating the process. This is done by determining the technology, type and system of the connector to connect to each system in the process.

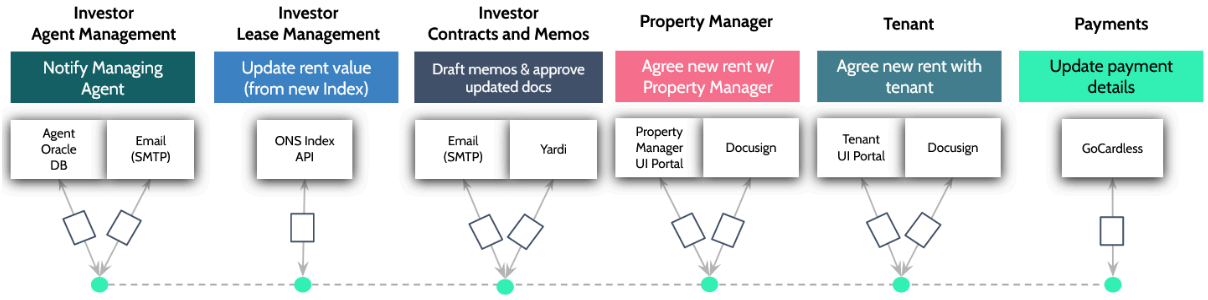


Fig 17. Luther's team set up connectors that link the processes together.

Luther, through numerous enterprise implementations has standard connectors to a majority of enterprise software systems across a range of processes and industries. For a full list of our connectors, please visit: *"Luther Platform Connectors"*.

6.5. Platform set-up

The Investor's team selected a set of configurations for their platform specifications. This selection depends on i) the process complexity (number of tasks), ii) amount of data processed (KB) per process run, iii) number of participants, iv) reliability, availability and security requirements for the application. Based on these selections, Luther's team deployed the platform on all nodes. For more details on platform configuration specs please visit: *"Luther Platform Connectors"*.

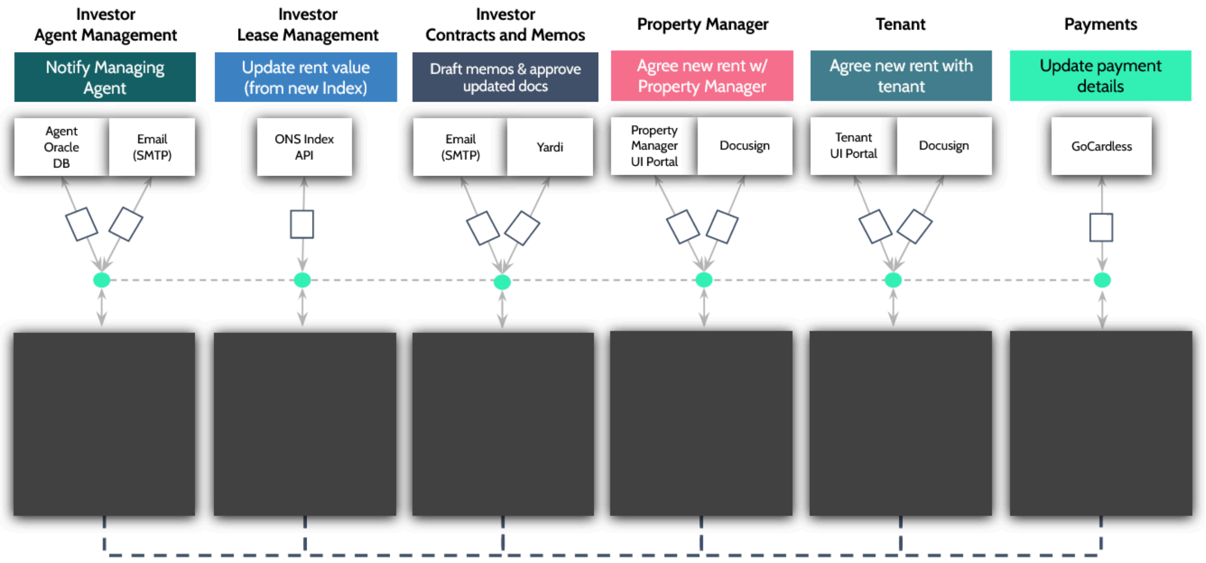


Fig 18. The platform is set up on each of the nodes, ready to reliably operate the end-to-end process at each step.

Luther's platform vertically integrates distributed system technology, optimal resource allocation and management, real-time event ordering and streaming (sharing), and deterministic event processing and execution, to provide a modern technology stack to reliably operate an end-to-end process across multiple software systems, at scale.

6.6. Common Operations Script for process operations

The platform is now fully set up and connected with all systems involved in the operation. The Investor's development team, in collaboration with Luther, developed the Common Operations Script to manage the end-to-end process. Connectors translate data from local systems into a common data model utilized by the Common Operations Script. This script encapsulates the business logic, data, rules, and validations for each process step.

Luther's platform vertically integrates

distributed system technology
optimal resource management
real time event streaming
deterministic event execution

To make reliable end-to-end process operations possible.

The Common Operations Script effectively codes and operates the process map, executing the Operating Cycle for each system across the entire process. To operate the process end-to-end, each function performs the same cycle of steps: i) send data & information to the System, ii) receive response from the System, iii) compute & validate response, iv) share & store execution of step, v) evaluate & initiate next steps.

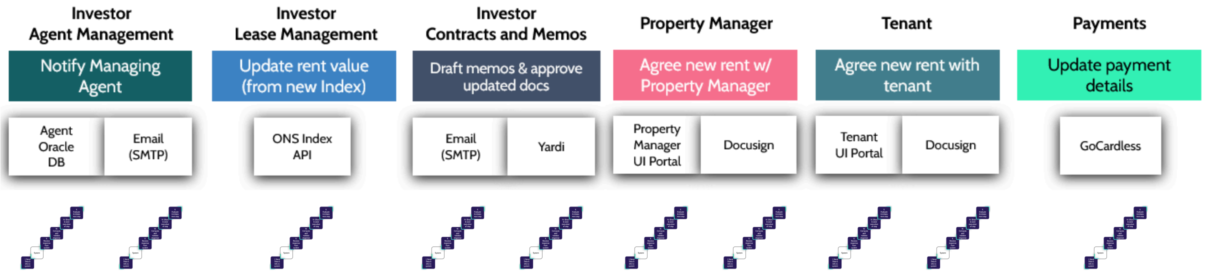

Investor Agent Management
Notify Managing Agent
Agent Oracle DB
Email (SMTP)
Investor Lease Management
Update rent value (from new Index)
ONS Index API
Investor Contracts and Memos
Draft memos & approve updated docs
Email (SMTP)
Yardi
Property Manager
Agree new rent w/ Property Manager
Property Manager UI Portal
Docusign
Tenant
Agree new rent with tenant
Tenant UI Portal
Docusign
Payments
Update payment details
GoCardless

Fig 19. These requirements repeat for all functions across the end-to-end Process Operations.

For a more detailed description of how the Common Operations Script operates the Process please see the Appendix.

This script is shared by all participants and operates on the Luther Platform. Each participant can change the script through suggesting changes, once the changes to the script are approved by all participants the script is updated for all participants.

Teams & Systems can easily change the Common Operations Script by proposing changes but first all other teams & systems have to know and agree to the change upfront.
So, changes are not an afterthought in a memo
All teams & systems involved operate the same end-to-end process all the time!

The enterprise has full autonomy over the process operations to modify and change them, and it also ensures all participants are operating "the same process" at all times. When a team changes their operations, the operations for all participants are updated simultaneously. For a demo of the build process please visit our website.

The enterprise has full autonomy over its Operations code

No need to call someone everytime you want to make a small change!

6.7. Go live (production)

Once the platform is set up and the Common Operations Script is coded, the application is ready to go live. Upon going live, it automates the operations of the end-to-end rent renewal process by providing i) standardized connectivity between teams and systems, ii) the Common Operations Script, shared by all teams, ensures a consistent process operation at all times. For more information about Luther's platform please consult this video.

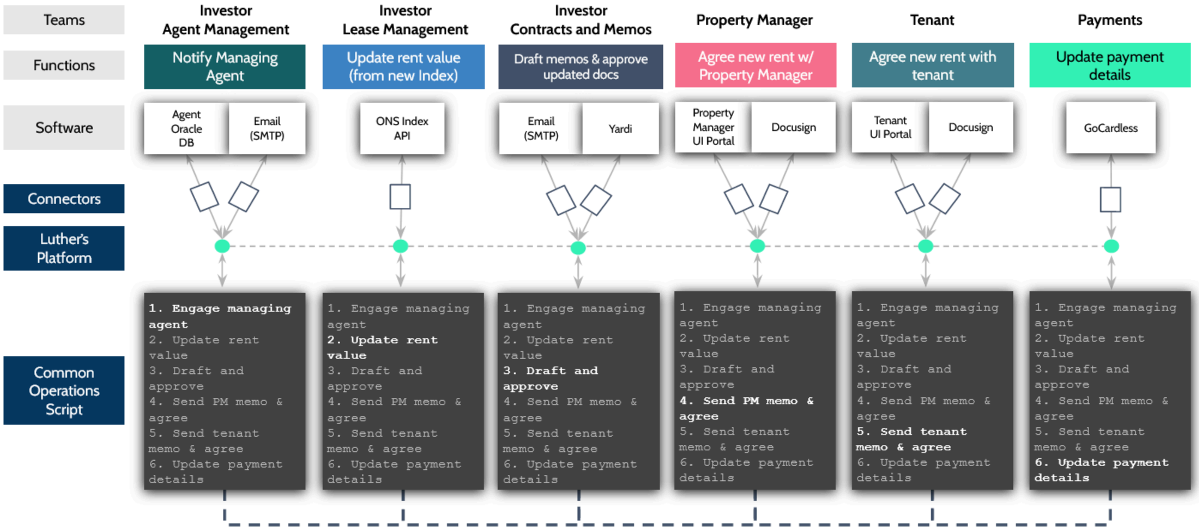


Fig 20. Luther and the Investor's developer teams work together to write the common operations script, converting tasks into an objective workflow that links every step in the process. The common operations script links independent systems into one cohesive process

7. Results

7.1. Commercial results

Using Luther's Deep Process Automation platform, implementing the automated Long-term Commercial Real Estate Contract Rent Renewal Process reduced the cost of processing the long-term lease contracts by 70%. This is primarily due to FTE savings in both ops teams involved in process operations as well ops teams involved in reprocessing faulty lease rent renewal. The percentage of errors is estimated to be reduced to below 2%, due to automating operations and identifying many errors in near real time. The average total time for the rent of a lease to be renewed was reduced from 5 days to just 5 hours, speeding up the average processing time by 9X. This results in a return on investment of 800%.

Specific commercial advantages:

- Average total time for rent renewal of lease reduced from 5 days to 5 hours
- fast turnaround times for contract updates, leading to better relations with tenants for property managers
- Cost savings of 70%, reducing the operational costs from 6.75 million a year to just 2 million a year, resulting in an ROI of 800%

ROI: 8X

Cost Reduction: 70%

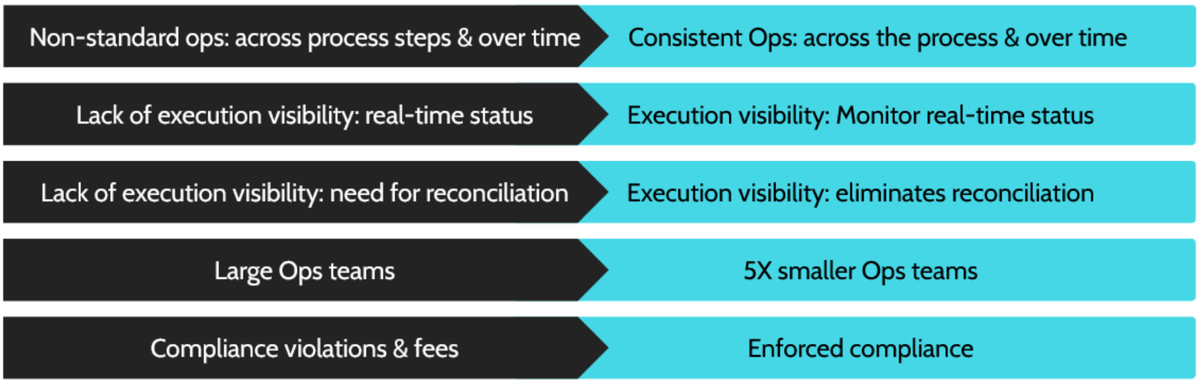
Average Processing Speed Increase: 9X

7.2. Operational benefits

Luther delivered a platform that standardizes long-term lease rent renewal process operations, and demonstrated the potential for other property managers to be integrated into the network, while reducing inefficiencies, improving process transparency, reducing the size of operations teams, and improving compliance, which could not have been achieved without Luther's Deep Process Automation Technology.

General operational advantages

The Luther Platform streamlines operations across enterprise processes, reducing process time and cost while maintaining transparency and flexibility.



Non-standard ops: across process steps & over time	Consistent Ops: across the process & over time
Lack of execution visibility: real-time status	Execution visibility: Monitor real-time status
Lack of execution visibility: need for reconciliation	Execution visibility: eliminates reconciliation
Large Ops teams	5X smaller Ops teams
Compliance violations & fees	Enforced compliance

Fig 21. General results from implementation of the Luther platform

Specific operational advantages

Implementing the automated Long-term Commercial Real Estate Contract Rent Renewal Process has streamlined the operations of the long-term lease contract rent renewal process, making it more efficient, faster, and standardized all while requiring minimal manual intervention, and without sacrificing transparency or integrity of data. The platform is flexible and scalable to future changes to the process or regulations.

Enhanced long-term lease contracts processing operations:

- Provides visibility and transparency for most up-to-date lease details to all participants
- Contracts and memos can be automatically updated with zero downtime

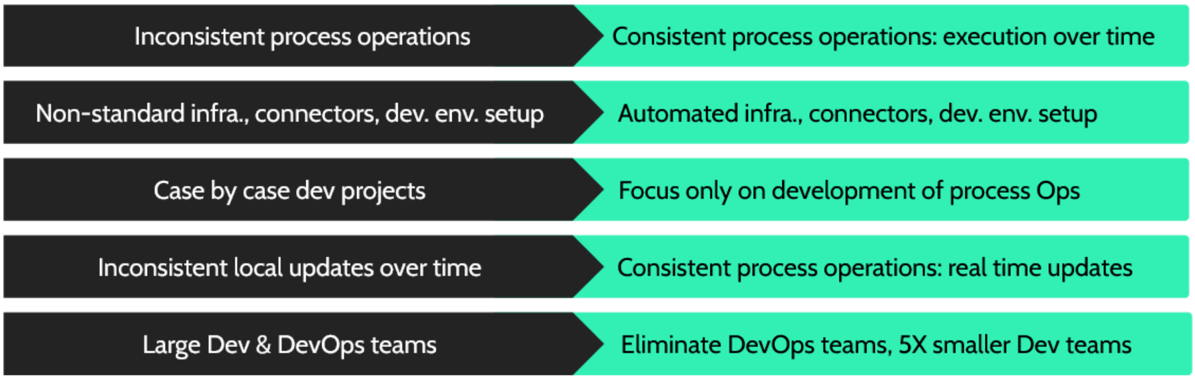
Enhanced rent renewal process operations:

- Elimination of manual intervention means smaller operational teams at the Investor
- Standardization of the rent renewal process, ensuring faster timescales helping to avoid compliance issues
- Increased process reliability and fewer processing errors reduce costs associated with duplicate leases or memos

7.3. Technical benefits

General technical advantages

The Luther Platform makes process operations more consistent as well as standardizing the infrastructure used to operate the long-term lease contract rent renewal process. Real-time updates across the end-to-end process ensure less downtime in the process, improving efficiency. All this means that developer and developer operations teams can be reduced in size and that developers can focus on developing and improving process operations rather than focusing on handling inefficiencies in the process.



Inconsistent process operations	Consistent process operations: execution over time
Non-standard infra., connectors, dev. env. setup	Automated infra., connectors, dev. env. setup
Case by case dev projects	Focus only on development of process Ops
Inconsistent local updates over time	Consistent process operations: real time updates
Large Dev & DevOps teams	Eliminate DevOps teams, 5X smaller Dev teams

Fig 22. General technical results from implementation of the Luther platform.

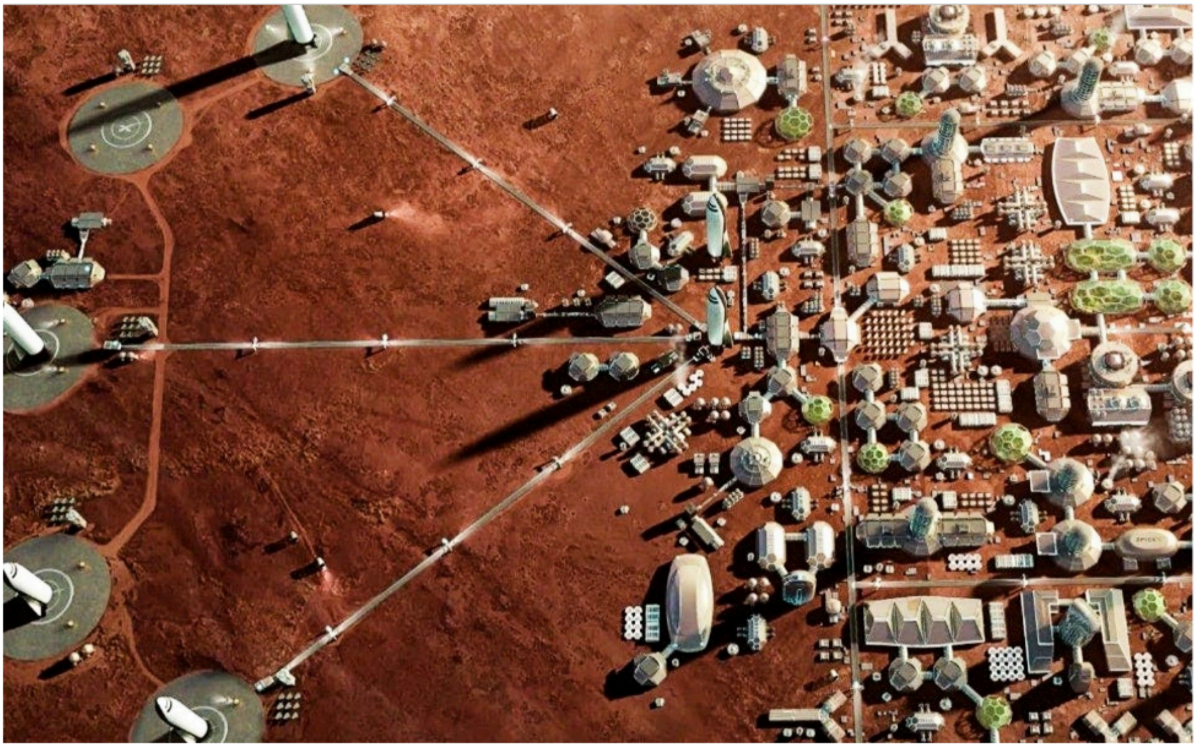
Specific technical advantages

Improved operating efficiency:

- Automatically verifies execution to increase reliability and reduces processing errors
- Common execution visibility to all participants reduces troubleshooting effort
- Automatically supports Common Operations Script updates including new validation rules

Reduced audit trail and reporting costs:

- Entire memo & lease history is securely maintained with timestamps for every event



8. Expansion

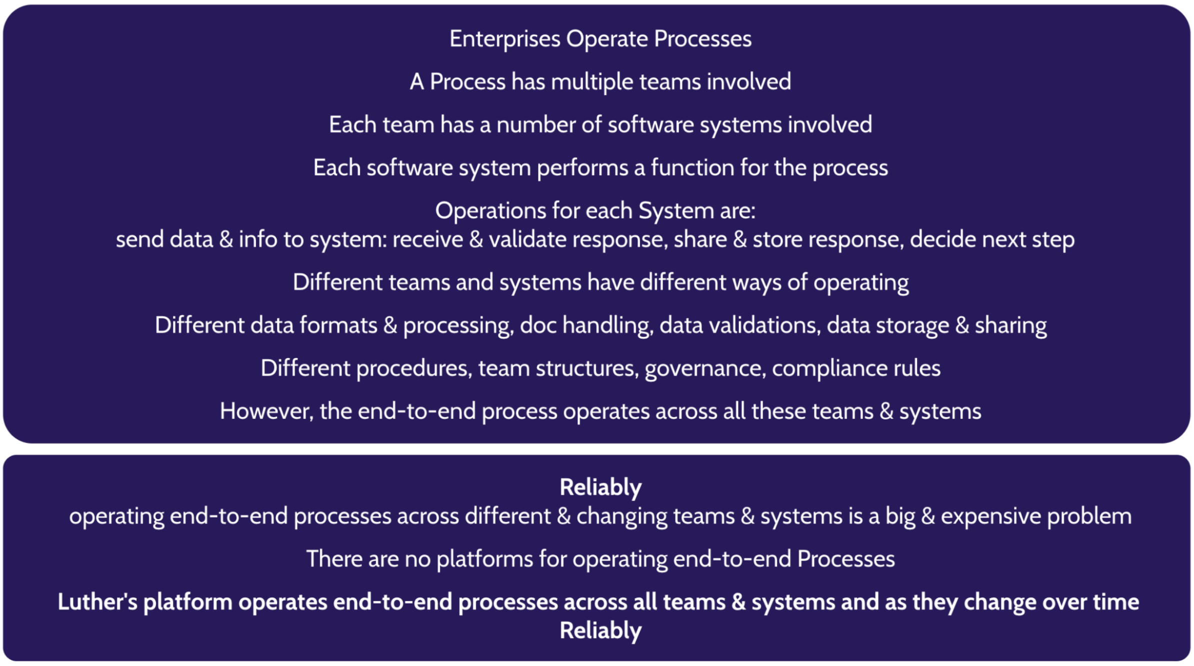
This project demonstrates a streamlined process built on the Luther platform that standardizes and automates the long-term lease contract rent renewal process operations for long-lease contracts. The network can be further expanded to encompass the renewal of other of the Investor's long-lease contracts. It could also expand to include other of the Investor's processes, including the management of other real asset investments such as infrastructure and private debt. Increasing the scope of the network to other investments, holdings and business lines could reduce costs and achieve economies of scale.

Potential areas of expansion to further automate the Investor's operations on the Luther platform include:

- **The Investor's properties:** Expanding the rent-renewal process to encompass other of the Investor's properties
- **Upstream & downstream in the lease contracts value chain:** Creating new products using the Luther Platform to expand to other aspects of managing real estate, such as drawing up initial contracts and property management
- **Creating new products** to automate the management of other of the Investor's assets
- **More automation:** More effectively resolving rent disputes and further eliminating manual intervention

9. Luther Company & Offerings

9.1. What Luther does



Enterprises Operate Processes

A Process has multiple teams involved

Each team has a number of software systems involved

Each software system performs a function for the process

Operations for each System are:
send data & info to system: receive & validate response, share & store response, decide next step

Different teams and systems have different ways of operating

Different data formats & processing, doc handling, data validations, data storage & sharing

Different procedures, team structures, governance, compliance rules

However, the end-to-end process operates across all these teams & systems

Reliably
operating end-to-end processes across different & changing teams & systems is a big & expensive problem

There are no platforms for operating end-to-end Processes

Luther's platform operates end-to-end processes across all teams & systems and as they change over time Reliably

Fig 23. Luther's platform solves the complicated problem of end-to-end enterprise process operations.

For more information about Luther, please visit our website.

9.2. "In a nutshell" - Luther's unique value

Luther's platform vertically integrates

distributed system technology
optimal resource allocation and management
real time event ordering and streaming (sharing)
deterministic event processing and execution

To make reliable end-to-end process operations possible.

Luther's unique value for reliable end-to-end Process Operations is providing

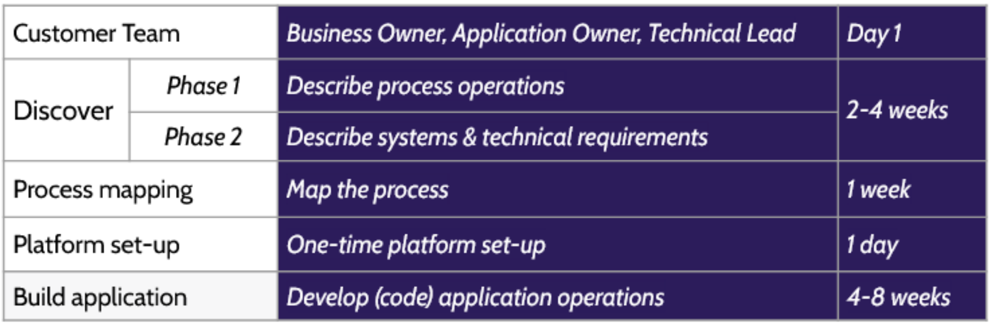
standard connectivity
a common operations script

across all teams and software systems.

Luther automates the operations of the end-to-end process across multiple teams and systems

9.3. Platform implementation

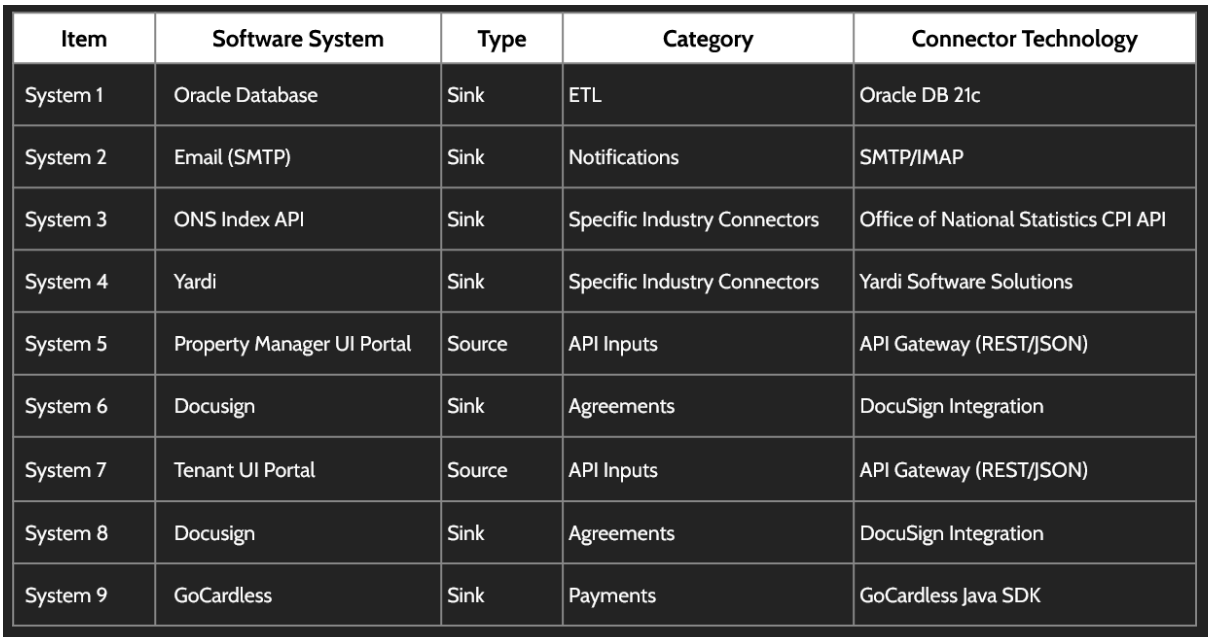
To implement the Luther Platform, organizations work with Luther through an implementation process - laying out objectives and expectations for the project, then mapping the process and setting up infrastructure. After this, enterprise developers build code that will execute the agreed process.



Customer Team		*Business Owner, Application Owner, Technical Lead*	*Day 1*
Discover	*Phase 1*	*Describe process operations*	*2-4 weeks*
	Phase 2	*Describe systems & technical requirements*	
Process mapping		*Map the process*	*1 week*
Platform set-up		*One-time platform set-up*	*1 day*
Build application		*Develop (code) application operations*	*4-8 weeks*

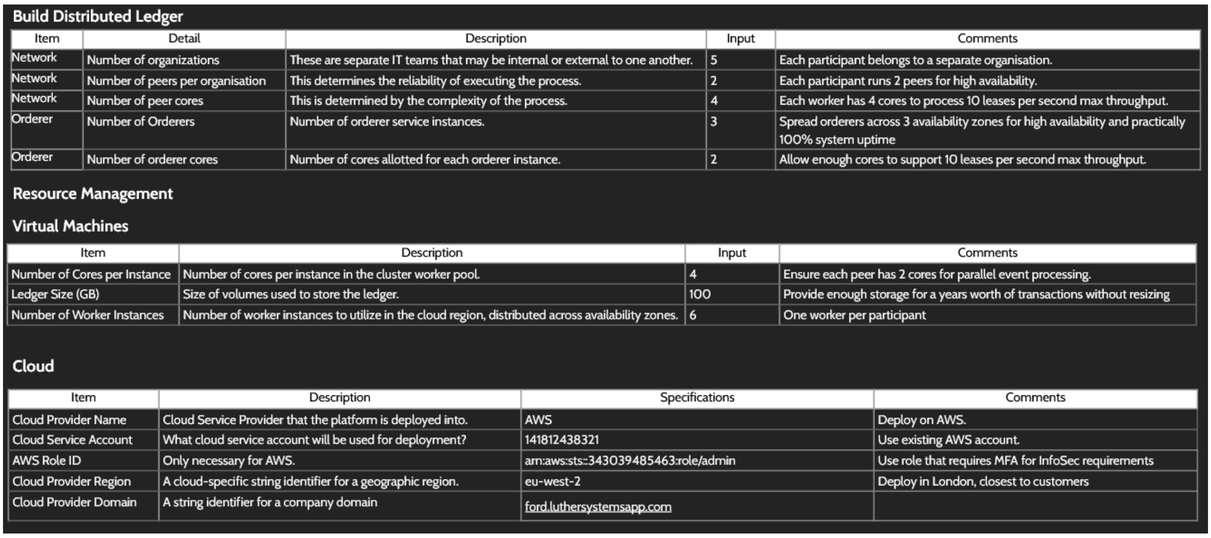
Fig 24. Implementation timeline for an application operated on the Luther Platform..

Enterprises working with Luther fill in the details of all software systems and connectors for their processes. These documents are used to build the process map and subsequently, the application.



Item	Software System	Type	Category	Connector Technology
System 1	Oracle Database	Sink	ETL	Oracle DB 21c
System 2	Email (SMTP)	Sink	Notifications	SMTP/IMAP
System 3	ONS Index API	Sink	Specific Industry Connectors	Office of National Statistics CPI API
System 4	Yardi	Sink	Specific Industry Connectors	Yardi Software Solutions
System 5	Property Manager UI Portal	Source	API Inputs	API Gateway (REST/JSON)
System 6	Docusign	Sink	Agreements	DocuSign Integration
System 7	Tenant UI Portal	Source	API Inputs	API Gateway (REST/JSON)
System 8	Docusign	Sink	Agreements	DocuSign Integration
System 9	GoCardless	Sink	Payments	GoCardless Java SDK

Fig 25. The list of software systems involved in end-to-end Long-term Commercial Real Estate Contract Rent Renewal Process operations



Build Distributed Ledger

Item	Detail	Description	Input	Comments
Network	Number of organizations	These are separate IT teams that may be internal or external to one another.	5	Each participant belongs to a separate organisation.
Network	Number of peers per organisation	This determines the reliability of executing the process.	2	Each participant runs 2 peers for high availability.
Network	Number of peer cores	This is determined by the complexity of the process.	4	Each worker has 4 cores to process 10 leases per second max throughput.
Orderer	Number of Orderers	Number of orderer service instances.	3	Spread orderers across 3 availability zones for high availability and practically 100% system uptime
Orderer	Number of orderer cores	Number of cores allotted for each orderer instance.	2	Allow enough cores to support 10 leases per second max throughput.

Resource Management

Virtual Machines

Item	Description	Input	Comments
Number of Cores per Instance	Number of cores per instance in the cluster worker pool.	4	Ensure each peer has 2 cores for parallel event processing.
Ledger Size (GB)	Size of volumes used to store the ledger.	100	Provide enough storage for a years worth of transactions without resizing
Number of Worker Instances	Number of worker instances to utilize in the cloud region, distributed across availability zones.	6	One worker per participant

Cloud

Item	Description	Specifications	Comments
Cloud Provider Name	Cloud Service Provider that the platform is deployed into.	AWS	Deploy on AWS.
Cloud Service Account	What cloud service account will be used for deployment?	141812438321	Use existing AWS account.
AWS Role ID	Only necessary for AWS.	arn:aws:sts::343039485463:role/admin	Use role that requires MFA for InfoSec requirements
Cloud Provider Region	A cloud-specific string identifier for a geographic region.	eu-west-2	Deploy in London, closest to customers
Cloud Provider Domain	A string identifier for a company domain	ford.luthersystemsapp.com	

Fig 26. A sample list of connectors and infrastructure, similar to one an enterprise building an application on the Luther Platform would fill out.

9.4. Results of the Luther platform for Process Operations Automation

At Luther, we recognize that enterprise processes of today are complex and challenging to automate. We provide a platform for successful process automation.

The results are incredible. Enterprises working with Luther see an average return of 10 times their investment. Time is saved everywhere, with development of process applications and automation technology sped up by 2.5 times, and processing times 7 times faster. Find out more about Luther's core platform features here.

2.5X faster development

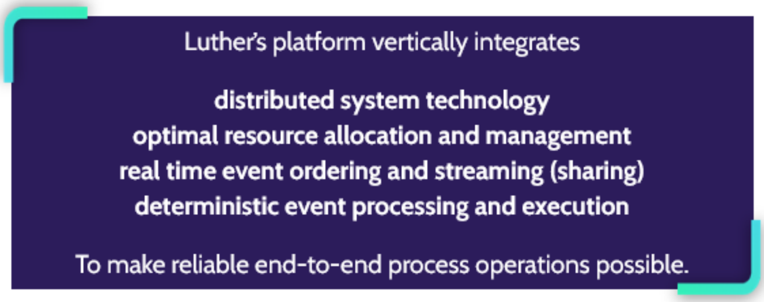
10X less operational costs

7X faster processing time

10X ROI

1000s of compliance rules automated

9.5. Luther's platform architecture



Luther's platform vertically integrates

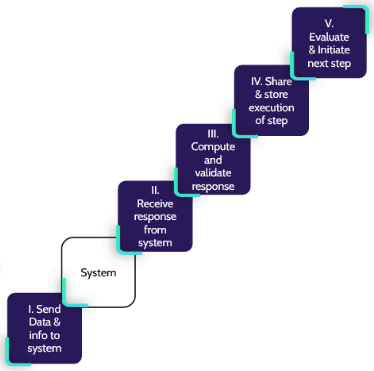
distributed system technology
optimal resource allocation and management
real time event ordering and streaming (sharing)
deterministic event processing and execution

To make reliable end-to-end process operations possible.

For a more detailed introduction on the Luther platform please request access to the "*Luther Deep Process Automation Primer*".
For a detailed introduction and documentation examples please see the Luther Platform site.
For more information about Luther's platform please visit luthersystems.com.

10. Appendix

10.1 How the platform operates an end-to-end process: Application walkthrough



V. Evaluate & Initiate next step
IV. Share & store execution of step
III. Compute and validate response
II. Receive response from system
System
I. Send Data & info to system

Below is a more detailed walkthrough of the process operations, across the teams and software systems. Each step in the process follows the exact same 5 operational substeps which the Platform executes:
i) **Send:** Platform sends data & information to the System,
ii) **Receive:** Platform receives response from the System,
iii) **Check:** Platform computes & validates the response,
iv) **Store:** Platform shares & stores execution of step,
v) **Next:** Platform evaluates & initiates next steps.

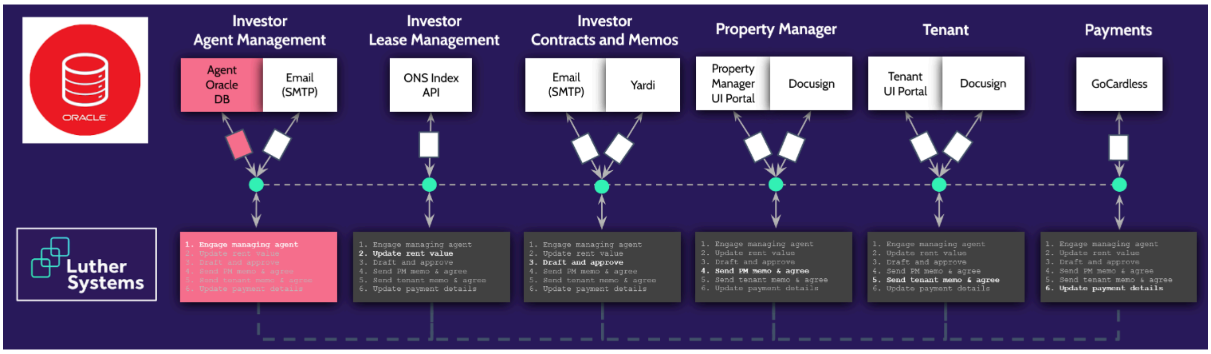
The Common Operations Script ensures that these operations cycle steps are carried out for all systems involved in the process to ensure reliable process operations.

The Platform operates the Process by standardizing the execution of each step in section 5.2. "How it Works on the Luther Platform"

Step 0: Investor Agent Management Team initiates the process by sending the Agent ID to the Platform through a portal.

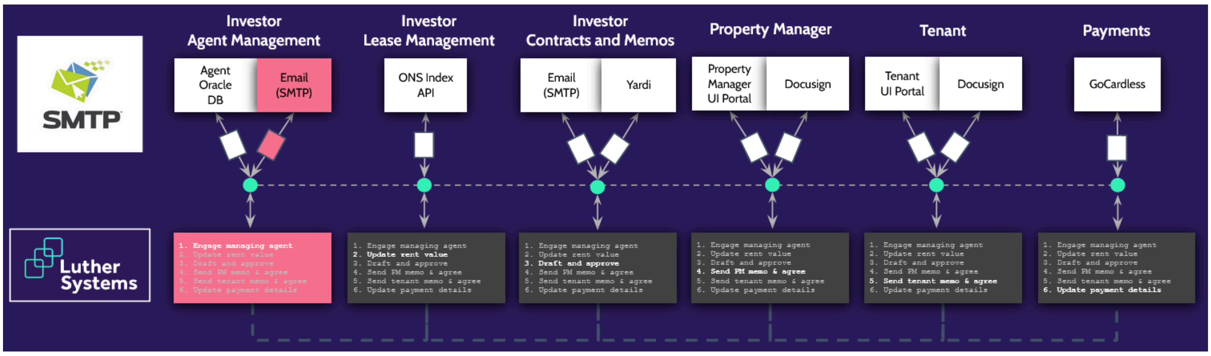
Step 1: Investor Agent Management team executes Notify Managing Agent , specifically Retrieve Agent Information

I. Platform sends *agent ID* (request) to Agent Database
II. Platform receives *agent information* (response) from Agent Database
III. Platform validates *agent information* based on predetermined rules in the Common Operations Script
IV. Platform shares & stores *agent information* from Agent Database
V. Platform evaluates & initiates next step



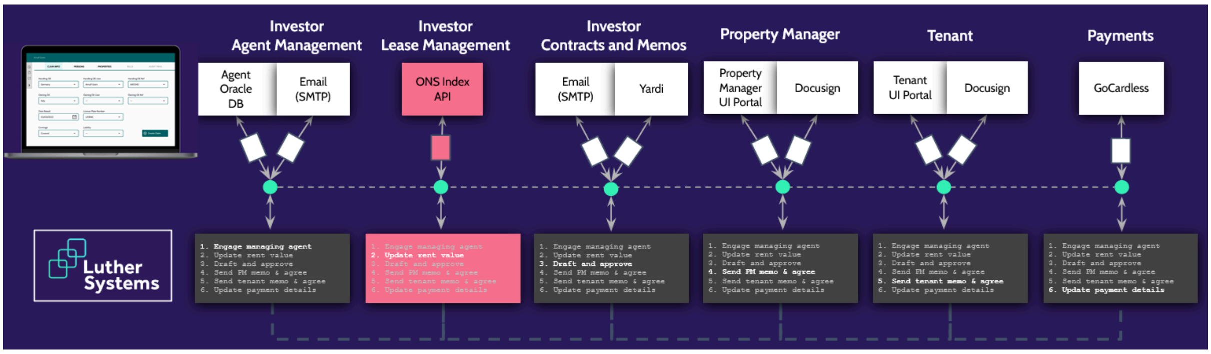
Step 2: Investor Agent Management team executes Notify Managing Agent, specifically Send Notification to Managing Agent

I. Platform sends *agent information* (request) to Email (SMTP)
II. Platform receives *email confirmation* (response) from Email (SMTP)
III. Platform validates *email confirmation* based on predetermined rules in the Common Operations Script
IV. Platform shares & stores *email confirmation* from Email (SMTP)
V. Platform evaluates & initiates next step



Step 3: Insurer Lease Management team executes Update Rent Value, specifically Update Rent Value

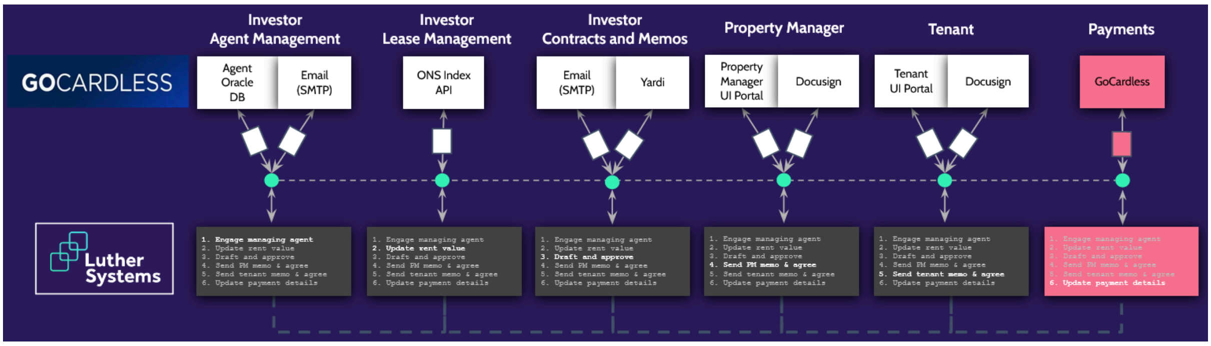
I. Platform sends *index value* (request) to ONS Index API
II. Platform receives *index value* (response) from ONS Index API
III. Platform validates *index value* based on predetermined rules in the Common Operations Script
IV. Platform shares & stores *index value* from ONS Index API
V. Platform evaluates & initiates next step



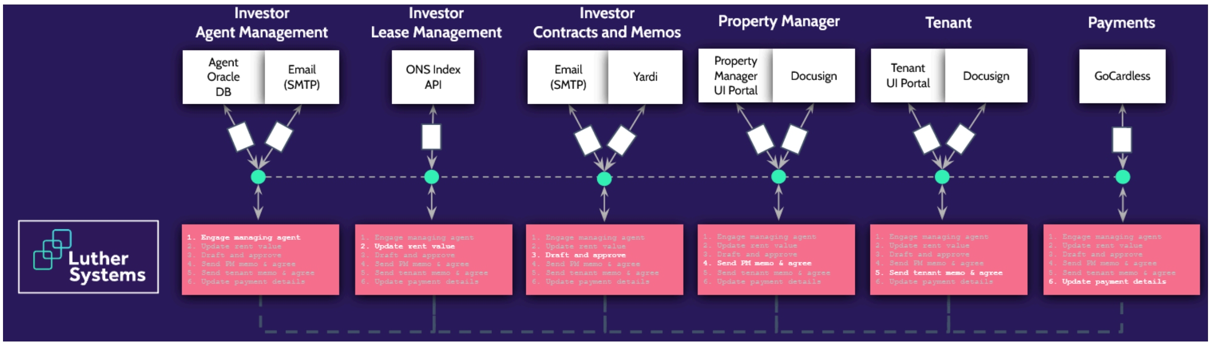
The steps operate in a similar manner until the final step is reached:

Step 10: Investor Payments team executes Update Payment, specifically Update Payment Details

I. Platform sends *payment information* (request) to GoCardless
II. Platform receives *payment information* (response) from GoCardless
III. Platform validates *payment information* based on predetermined rules in the Common Operations Script
IV. Platform shares & stores *payment information* from GoCardless
V. Platform evaluates & initiates next step



Final Step: The Platform completes the process:



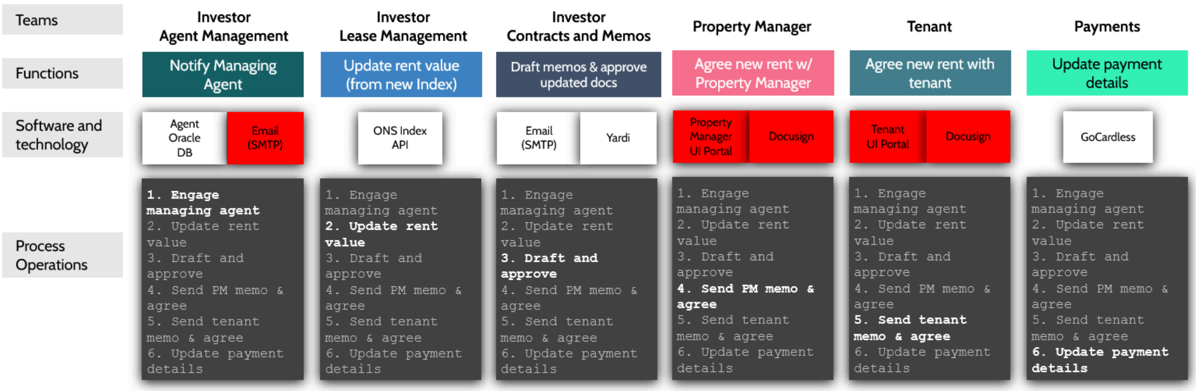
10.2. Definitions

Term	Definition	Examples
Task	Simple events that are localized to one team involving one or two software systems	Copying data between systems, retrieving data from a database, making a payment
Workflow	A series of 10-20 tasks involving 1-2 software systems and 1-2 teams	Collecting related data from several systems, onboarding a new employee
Process	A series of 20+ tasks involving 3+ teams and multiple software systems	Reviewing the rent of leases
Value Stream	A collection of processes delivering a business critical value	Leases and Contract Management, Property Acquisition
Participant	Operationally separate teams that have their own operations, governance and utilization of software systems and can make some autonomous decisions	Insurer participants: Agent Management, Lease Management, Contracts and Memos Other participants: Property Manager, Tenant
Team	As broadly defined by enterprises, otherwise known as departments, groups, units, etc.	All employees who work on Agent Management in the long-term lease rent review process
Function	A unit of operations performed by a single team	Updating rent values in a contract
Process Operations	End-to-end completion of process operations across multiple teams and software systems, to deliver a specific business objective	The long-term lease rent review process

10.3. Process Journey vs. User Journey

The Process Journey involves all the systems and teams including interactions with the users of the process, which usually interact with the process through UI systems and specifically designed Apps, with their own interfaces. However, process operations run through a much larger set of systems and teams, most of which are not visible to the user.

The User Journey is a small subset of the Process Journey. For an optimal User Journey, the whole process must operate reliably, not just the systems involved in the user journey! They must all operate correctly to operate the process end-to-end.



Teams	Investor Agent Management	Investor Lease Management	Investor Contracts and Memos	Property Manager	Tenant	Payments
Functions	Notify Managing Agent	Update rent value (from new Index)	Draft memos & approve updated docs	Agree new rent w/ Property Manager	Agree new rent with tenant	Update payment details
Software and technology	Agent Oracle DB Email (SMTP)	ONS Index API	Email (SMTP) Yardi	Property Manager UI Portal Docusign	Tenant UI Portal Docusign	GoCardless
Process Operations	**1. Engage managing agent** 2. Update rent value 3. Draft and approve 4. Send PM memo & agree 5. Send tenant memo & agree 6. Update payment details	1. Engage managing agent **2. Update rent value** 3. Draft and approve 4. Send PM memo & agree 5. Send tenant memo & agree 6. Update payment details	1. Engage managing agent 2. Update rent value **3. Draft and approve** 4. Send PM memo & agree 5. Send tenant memo & agree 6. Update payment details	1. Engage managing agent 2. Update rent value 3. Draft and approve **4. Send PM memo & agree** 5. Send tenant memo & agree 6. Update payment details	1. Engage managing agent 2. Update rent value 3. Draft and approve 4. Send PM memo & agree **5. Send tenant memo & agree** 6. Update payment details	1. Engage managing agent 2. Update rent value 3. Draft and approve 4. Send PM memo & agree 5. Send tenant memo & agree **6. Update payment details**

Fig 27. The process journey of the automated Long-term Commercial Real Estate Contract Rent Renewal Process. Systems highlighted in red directly interact with or require direct interaction from the property manager or tenant.

10.4. Plaintext Links

6. Implementation

For a walkthrough of the implementation process, view the Luther Systems Sandbox Setup: https://app.platform.luthersystemsapp.com/sandboxSetup

For a full explanation of the implementation process, view the Full Luther Platform Setup: http://app.platform.luthersystemsapp.com

For a more detailed description of the implementation steps please visit: https://www.luthersystems.com/platform/platform-overview

Request access to an example of a more detailed timeline here: https://docs.google.com/spreadsheets/d/1jHSeFRhaWVkUiEtQ_crxGoyGFJ82eGUZ3rxhnYi4cro/edit?gid=1722375828#gid=1722375828

9. Luther's Company and Offerings

For more information about Luther's platform please visit our website: http://luthersystems.com

Find out more about Luther's core platform features here: https://app.platform-test.luthersystemsapp.com/features

For a demo of the build process please visit our website: https://app.platform-test.luthersystemsapp.com/build

For more information about Luther’s platform please consult this video:
https://www.youtube.com/watch?v=78H5m1aZZoU

For a more detailed introduction on the Luther platform and a full explanation of traditional process operations and Luther’s solution please request access to the Deep Automation Primer here:
https://docs.google.com/document/u/1/d/1O3KIQUDuwMV0e5CzjNFMYoYnq7g_7AoU_qIHLOza_Tw/edit

For a detailed introduction and documentation examples please see the Luther Platform site:
https://www.luthersystems.com/platform/platform-overview