genesis
Code Less, Do More™

How a **low-code** platform can deliver you from **legacy**

Financial markets are still over-reliant on outdated systems that are kept on life support by an endless succession of integration projects. But what if you could use integration to shrink the functional footprint of these systems until they become simple enough to remain, or be phased out without disruption?

2020

Executive Summary

Legacy systems are a thorn in the side of financial markets; some have been around for decades which, as unfit for purpose as they often are, makes them difficult to do without.

The regulatory shortcomings of many legacy systems and their inability to support technological innovation (even the most fundamental innovation of cloud computing) forced financial markets to invest heavily in integrations. In one metaphor, APIs became the pieces of string that held everything together. Forrester[1] put it more politely when it heralded the advent of "the API economy".

The "spaghetti architecture" of these integrations (to use another metaphor) does not solve the core legacy problems, because businesses will be writing more and more APIs for as long as the outdated systems remain in place. However, the technology of low-code paves the way for a different integration scenario whereby the functionalities of legacy systems are replaced piece by piece.

An obvious but by no means simple first step is to understand the full functional footprint of the legacy system's architecture. Having identified all data-feeds flowing upstream and downstream through multiple systems, it will be possible to use modern low-code techniques to create the necessary integration protocols for an efficient IT architecture layout that is robust and efficient without the need for immediate legacy replacement. These feed mechanisms become the gate keeper to protect the legacy systems architecture, encapsulating it and obfuscating its existence. With control of the feeds it is also possible to rapidly replace the user interfaces of these legacy systems, while retaining the core functionality in the legacy stack.

The mitigation strategy should be to steadily shrink the functional footprint, prioritizing functions that the legacy architecture finds challenging. Over time, as the footprint shrinks, the technology debt is reduced, costs become more manageable and agility increases. The legacy IT landscape will then either become simple enough for it to remain, or the task to replace it completely will be much more feasible.

1 - https://searchapparchitecture.techtarget.com/blog/Microservices-Matters/Connect-the-Now-and-API-Economies-Must-have-capabilities

Contents

About the author:

Ray Chee

Head of Solutions Delivery
- Genesis Global

Ray has over 25 years' experience as a senior technologist at a number of top-tier financial services firms including Bank of America Merrill Lynch, Nomura, NatWest Markets and Royal Bank of Scotland, creating and leading teams of up to 100 to deliver projects across all the major financial centers. Ray is Head of Solutions Delivery and part of the senior leadership team executing the genesis growth strategy.

If you have any questions about this technical brief or would like to demo the genesis LCAP, please connect with Ray directly at **raymond.chee@genesis.global**

Foreword

Financial markets feel they are in a Catch-22 with legacy: it is a roadblock to innovation, yet any attempt to clear it will be too disruptive. However, businesses know they cannot afford to sit on their hands about this, not least because the rapid acceleration of technological change is also adding to the list of what is legacy. Today's shiny new software is tomorrow's legacy.

Integrations have so far largely been a matter of running to stand still, developing (and maintaining) a network of APIs to drag legacy systems into the 21st century. This keeps the show on the road but does nothing to address the core problem – fragmented IT architectures that are too inflexible and sluggish to remain competitive in a rapidly changing operational and regulatory market environment.

Until now there was a binary choice between replacing legacy systems or yet more API acrobatics. No longer. The genesis Low-Code Application Platform (LCAP) enables a strategic approach where functionality is gradually migrated away from the legacy system until it can operate safely and harmlessly in the background – or become so self-contained it can finally be replaced.

The Genesis LCAP is unique not only in reducing traditional coding effort by 80% but also in having been built to do so for even the most demanding financial markets use cases – a game-changing differentiator compared with other Low-Code tools on the market.

With legacy systems, LCAP plays a long game. Functionalities that pose the biggest problems for the system architecture are prioritized and "satellite solutions" that remove the friction points from legacy are built using the genesis platform.

Legacy mitigation is just one of the challenges that the Genesis LCAP addresses, and over the coming weeks and months, genesis will continue to talk to financial markets about other innovative approaches.

This is only the beginning. Gartner predicts that by 2024, low-code application development will be responsible for more than 65% of application development activity. This is an opportunity that financial markets cannot afford to miss – and do not have to miss, now that genesis has cracked the code to growth and innovation.

Chapter 1: Introduction

The financial services industry has been one of the drivers of innovation in the Age of Computing. The earliest computers – even the 19thC ancestors of what eventually became binary computing – were invented to be good at number-crunching. And money is about numbers. With the exponential increases in computational power, financial markets were able to take "money" to incredible heights of abstraction with complex derivative structures. The financial crisis of 2008 was an existential wake-up call to remind us that no matter how dazzling our mathematical modelling, it can never escape the realities of every-day life, of money.

The pioneering role of financial markets in the development of digital technologies has given rise to the following paradox: no industry is as dependent on legacy systems, and no industry has been disrupted more fundamentally by start-ups that owe a large part of their success (when they are successful) to having no legacy at all.

For fintechs, "agility" is a given. For established financial firms, it is a continuous process of digital transformation.

Legacy is a roadblock to agility, poses compliance risks and makes it virtually impossible to deliver the slick frontend that users now expect – and which they get in spades from fintech apps.

Replacing legacy is not like replacing a battery. The German retail giant Lidl discovered this the hard way when it pulled out of a project to replace its outdated merchandizing system with SAP's S/4 Hana – after wasting seven years and €500m on attempts to customize it. For financial firms, even the prospect of business disruption is problematic, so their preferred route has been to mitigate legacy by integrating it with modern systems and software.

> "**92** out of **100** top banks still rely on IBM mainframes."
>
> -IBM (2017)

The complexity of what is being integrated in financial markets systems architecture is of a different order than is standard in telecommunications, retail, manufacturing, government and healthcare – verticals that also have a lot of legacy.

Financial markets demand a platform that offers "prefab" complex integrations, with sector-specific APIs built in. Such a platform would not only make it viable to develop applications for financial instruments and workflows that connect with existing (legacy) systems ¬– but do so in a short time-frame. The Low Code Application Platform (LCAP) from Genesis Global was built from the ground up to speed up app development for financial markets use cases – typically by as much as 80%.

Standard LCAPs are used in financial markets, but only around simple use cases such as Business Process Management (BPM). The genesis platform casts its net much wider, and takes on complex financial markets use cases, including integrations with legacy systems.

The integrations specific to financial markets for which the genesis platform was conceived make it possible for legacy to keep functioning within a rapidly changing technological environment – but they do nothing to make financial firms less reliant on legacy. This is the argument of the vendors of monolith systems: unless you replace your legacy systems lock, stock and barrel, you are merely accreting more integrations, more APIs, without ever resolving the core problem.

This technical briefing is a riposte to that point of view. It outlines a second integration path: one where the genesis LCAP builds "satellite apps" that migrate functionality away from the legacy systems, gradually neutralizing them with modern, agile capabilities. The legacy systems continue to perform their core functions without slowing down the business or posing a regulatory threat. In certain cases, when the legacy system has had to surrender most of its functionality, it will have withered on the vine, and can finally be retired.

In the opening section, we reiterate some of the problems posed by legacy and why removing it wholesale is almost never an option for financial markets. We then go on to describe how and in which contexts low-code can be used to shrink the functional footprint of the legacy IT architecture. We conclude with three short case studies of this approach.

Chapter 2: Why legacy systems are problematic

Many legacy systems are built using technology that was prevalent in the 20th century and does not take advantage of the latest capabilities. This makes them problematical on many fronts, including:

- **System connectivity:** this is often very bespoke and unsophisticated e.g. flat file.
- **Hardware:** some of the platform operating systems and product versions do not run on the latest hardware.
- **Malware:** aging software versions can prevent the latest security patches being applied.
- **Software deprecation:** often legacy systems will have incorporated other technologies within their core functionality. Over time this type of software deprecates and can cease working altogether on desktops and other devices. This can even affect the most commonly used browsers where legacy platforms cannot handle later versions.
- **Functionality:** modifying these systems in a timely fashion (if at all) can be challenging and the UI usually ends up looking hopelessly out of place in a world where the slick frontends of fintech apps, Über and Netflix are the norm.
- **User interface:** The oldest systems still in use have "green screen" frontends or terminal emulators. According to IBM, an astonishing 92 out of 100[2] top banks still rely on its mainframes as recently as 2017.
- **Single points of capacity:** many legacy systems have single-threaded components that can no longer cope with 21st century performance needs.
- **Regulation and audit:** the latest requirements for security, data lineage, unauthorized access and so on can often be beyond these legacy systems.
- **Regional divergence:** the global legacy system might not do what is needed in a specific regional unit (i.e. Brazil or Singapore) where there may be different requirements for reporting, or compliance, or financial instruments types.
- **Surveillance and monitoring:** many of these systems cannot support business surveillance nor do they have the ability for the level of environment monitoring needed to run resilient systems in a robust fashion.

2 - https://www.fnlondon.com/articles/banks-face-spiraling-costs-from-archaic-it-20170912

- **Legacy infrastructure:** based on system designs and architecture that long predate the advent of cloud services, legacy systems may struggle to offer the elastic capabilities for both compute and storage of current cloud-ready designs, along with the improved range of disaster recovery and business continuity options offered by the cloud.

- **Software Development Lifecycle (SDLC):** the design of the systems comes from an era when the main SDLC was waterfall. This can make it challenging, if not impossible, to move to an Agile or DevOps methodology.

Chapter 3: Why replacing legacy is problematic

Legacy systems are still around for a reason: replacing them is costly, complex and extremely disruptive. No senior business leader relishes the prospect of committing to a project that is highly likely to fail, as happened in the case of Lidl, and is certain to run massively late and over budget.

If cost is not an issue, disruption certainly is. The CIO of a leading bank puts it as follows: "Clearly we make all of our money from the systems we have today, not from systems of the future. Upgrades can't put any of the systems of today at risk[3]."

For many years, financial firms had little choice but to circumvent legacy by connecting layer upon layer of new features to it. This not only kicks the problem into the long grass, but also makes it worse, because with every additional feature, the challenge of migrating off the legacy system grows.

But what if the integration of modern functionalities was part of a strategy to gradually shrink the footprint of core systems? Such an approach would alleviate the need to replace legacy as the reduction of technical debt would bear down on cost and unlock agility. The legacy system would be able to fulfil core tasks, but out of harm's way, so to speak.

In the following section, we describe what some of this "satellite functionality" would look like. We begin at the beginning: with the data.

3 - https://www.fnlondon.com/articles/banks-face-spiraling-costs-from-archaic-it-20170912

Chapter 4: Data In (transforming the user experience)

A common task of the technology team is to improve the usability of their core technology framework applications. Its users are accustomed in their every-day lives to the slick interfaces of Apple Pay, Netflix et al so an ancient "green screen" at the office, or any of those "Super Mario"-like late 20th century graphics, inspire neither confidence nor enthusiasm. A poor system interface is a drag on time and morale. Downstream users get to see a more acceptable UI, but not without extensive deployments to end-user desktops and heavy network traffic, technologies that seldom support the latest business models which include high-frequency trading, low-latency capability and web and mobile software direct to the consumer.

It does not have to be this way. With a Low Code Application Platform (LCAP) that "understands" financial markets, it is possible to encapsulate frontends with the latest user interface. The core functionality continues in the legacy system, but the user now encounters the latest GUIs delivered in an agile fashion. This can often be done rapidly in a cost-efficient manner.

With insight into its possible input capability, the LCAP can typically build an adapter into the legacy system in a number of weeks (against many months using traditional technologies). This enables the use of the latest frontend technology – including rich desktop clients, web and mobile – in a fully controlled and audited framework that inserts data, transactionally, into the legacy system which is operating as the golden source.

A recent deployment of a low-code "satellite" solution for a legacy system had trade capture for complex products integrated to the core legacy system in a matter of weeks. By using an LCAP application, this came with all the necessary security and audit capabilities to guarantee that no audit issues or control weaknesses were introduced.

The ability to put the latest frontend technology in front of a legacy software system transforms not only the user experience but also the pace of change that can be realized going forward; the new software enables rapid change while the legacy software focuses on its core functionality. This is a low-cost and rapid way to improve systems: weeks or months versus the years necessary to replace a large legacy stack.

By coupling this approach with a solution for Data Out of the platform it is possible to build a low-code solution that encapsulates the legacy framework, enabling end-users to experience the latest technology with the core functionality of the legacy system continuing discretely – and discreetly – in the background. This improves controls, accelerates time to market and analytical capability, while maximizing the assets in place in the legacy system.

Chapter 5: Data Out (enabling new business flows)

Using an LCAP platform to build new interfaces can enable rapid changes downstream and enable the removal of “hacks” and tweaks in the feed area of the legacy systems.

For example, a system used throughout the financial industry only supports XML on MQ as a final endpoint or requires end-of-day data feeds into a reporting data structure. To make rapid changes to this is challenging as all coding must be done in the bespoke legacy system language, necessitating extensive regression testing for any change, as well as specialist practitioners to implement the changes.

Deploying an LCAP solution, using the latest technology, enables direct interrogation of the underlying data structures in a way that brings the data (in real-time) into a modern in-memory NoSQL database. Although many organizations have historically touted this as an impossibility, this methodology has now been proven – and within an eight-week period.

The eight-week project created a fully validated, real-time data set in a modern low-code application that is permanently in sync with the core golden-source legacy system. This made it possible to rapidly create the mandated downstream reporting and feeds without the need to continually query the legacy system. With this integration, the legacy system focuses on delivering core business functionality while the low-code application manages the ever-changing downstream data needs in a controlled fashion.

Chapter 6: Reporting

Once a mechanism is in place for accurate data to come out of the legacy system into a low-code platform, it becomes possible to consider replacing the myriad of tactical or End User Computing (EUC) reporting solutions that will inevitably have been built around the legacy framework, providing a much simpler reporting stack.

Global legacy systems often "fall down" at a regional level, with specific financial markets and jurisdictions operating different regulatory and compliance reporting regimes. A low-code platform can give the legacy system a new lease of life by integrating it with reporting solutions compliant with the reporting protocols of each particular market.

The latest front-end technologies have the ability to create visual data tools and graphs in real-time (driven by the user), and also ensure that data is controlled rather than emailed in a spreadsheet or CSV file as is the case with EUC reporting. By moving these reports to an online environment, data becomes controlled, traceable, and accurate and strips away the ability to modify or massage this data (improving data lineage and audit trails).

This type of transformation is sometimes met with pushback from co-workers used to being able to get data into a spreadsheet so they may then summarize and reorder it. By controlling this process and ensuring the right reports are in place online (in real-time), fewer workers will be required and roles where little value was added can easily be removed. In this way, co-workers can focus on making decisions based on the data rather than preparing the data.

"The use of **low-code platforms** instead of legacy software systems provides future-proofing against costly **monolithic legacy** systems."

- Genesis

The power of LCAPs to bring the latest customizable code and applications to the end-user can spark a much-needed change by empowering users to customize the experience, rather than modify the data, enabling data mining and presentation capabilities into the legacy system where the data safely resides. Where end-users may have accepted a sub-optimal workflow or tool given the time, cost, complexity, and inertia to change, the opportunity is now there to ensure the system can optimally support the precise needs of the business users without compromise.

As well as the significant improvements in control, it is probable that legacy reporting solutions can be retired; many of these are expensive to manage because of hardware and licensing costs, and the recurring costs of development and run staff.

The removal of these solutions saves money and frees up significant amounts of technology. It also improves data lineage requirements by removing legacy systems from the ecosystem. The use of low-code platforms instead of legacy software systems provides future-proofing against costly monolithic legacy systems through the use of the latest cloud and Open Source, alongside plug- and-play components.

Chapter 7: Solving vendor system issues

Many legacy vendor systems have inbuilt issues – for example single-threaded areas that are "impossible to fix without an upgrade". However, by complementing old technology with new technology it is possible to resolve these problems and remove the need for an expensive upgrade, or at least delay the need to avoid outages and system delays.

Examples of this include peripheral services, such as document generation. Often these run as autonomous services and are fixed as single-threaded. However, it can be possible to fool the legacy system into running multiple instances in a safe, controlled way. This has been done to great effect in several organizations where the removal of constraints of the legacy system is alleviating critical stress points and paving the way for replacement of the legacy stack.

Positive action and the use of smart new technology allows the legacy technology to work comfortably alongside the latest technology enabling investment to be focused in the right area; not everything needs replacing.

"Positive action and the use of **smart new technology** allows the legacy technology to work comfortably alongside the latest technology."

- Genesis

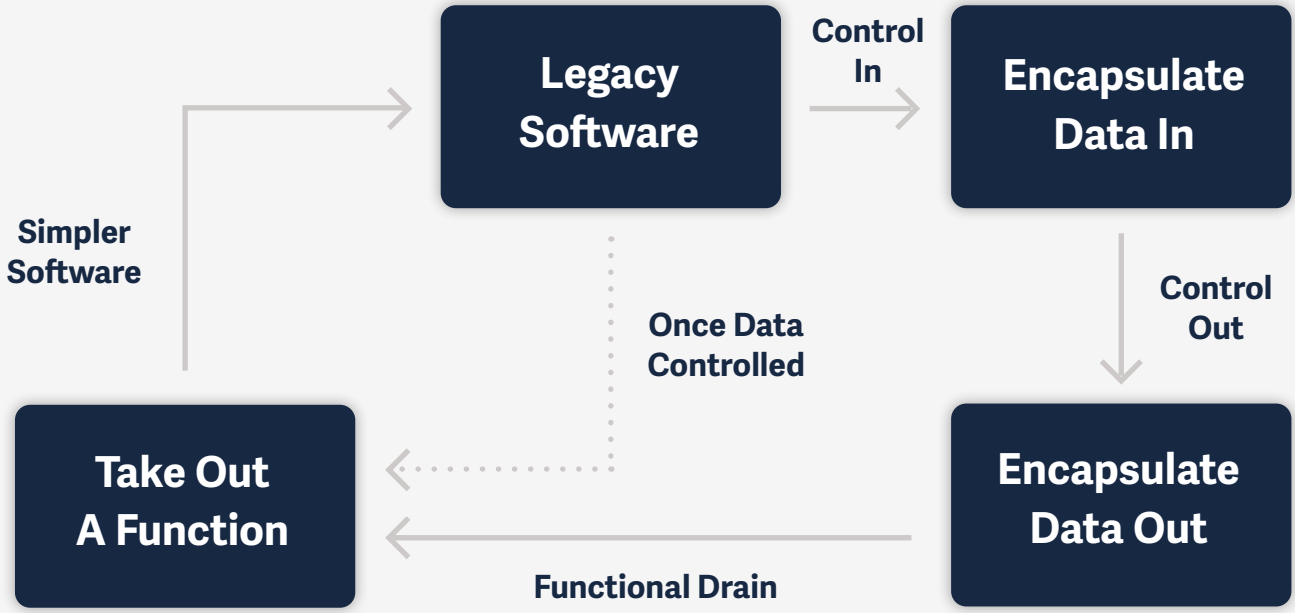
Chapter 8: Encapsulation and functional migration of legacy

Once the legacy system is integrated with (i) a low-code application supplying a data **input** mechanism; (ii) a data **output** mechanism and (iii) the ability to generate **user reports** in a controlled way, it is possible to shrink the legacy software footprint even further.

By moving all data input and output into the low-code application, we are realizing an encapsulation strategy that neutralizes the legacy system through tight control of data entry and egress, and protects the wider organization by forcing all data through a modern low-code application that has all necessary audit and control functionality in place.

Alongside the encapsulation process, the low-code application can initiate functional migrations off the legacy system by focusing on functions that should not reside in the legacy system, and prioritizing areas where the legacy system is experiencing stress or significant control weaknesses.

Over time, the legacy system becomes encapsulated and has a clear functional footprint that fits into the broader architecture. At this point a decision is required as to whether the legacy system in its new state is fit for purpose, or whether to continue to migrate functionality until the process reaches its logical conclusion: the total replacement of the encapsulated legacy system



Case Studies

1: Legacy Database

A low-code application was deployed to manage the data reporting and feeds for a legacy system. The legacy stack was struggling with control points and could not manage to keep up with the pace of change.

The low-code application was deployed to interrogate the underlying data structure of the legacy database. From this it rapidly created an in-memory NoSQL replica running at real-time. This mechanism was then used to generate new data feeds and provide end-user reporting. At all times, the legacy system was able to continue, unaware of the changes being made around it, as the new low-code application took on the burden of the much-needed changes. All downstream feeds and reporting can now be moved off the legacy system in a cost-effective manner, significantly improving the data lineage and control points.

2: Complex Structured Trades

A trading desk needed to book and price complex structured trades, but its golden source of trades was a legacy system unable to manage these new trade types. A low-code application was rapidly deployed to manage the upstream capture of these trades, insert a golden source record into the legacy system (with the more obscure financial data as an XML field). The low-code application then also intercepted the legacy pricing calls to re-inflate the structured trade and provide a full pricing and risk analysis of the trade in a manner that the legacy system could understand. The legacy system therefore remains the golden source of all trades but has been extended to manage complex structured trades rapidly, at a low cost.

3: Vendor System

A complex technology framework with a vendor system at its heart needed to rapidly create the ability to provide external client interactions and negotiations, based upon the golden source of data. The deployment of the Genesis low-code application enabled a sophisticated, modern, frontend to be deployed within weeks, taking data from the legacy vendor system and integrating with the complex interoperating technology framework that had been built up over time. This gave rapid client benefit and provided a segue for the complex framework to start a steady migration of functionality into a low-code application with the associated benefits of using the latest technology both in the frontend and in broader business flows.

genesis
Code Less, Do More™

About Genesis

Genesis is a global software company rewriting the rulebook for financial markets. Our unique Low-Code Application Platform (LCAP) is a robust, real-time, scalable framework and toolset allowing you to rapidly build and deliver solutions.

Our LCAP is built specifically for financial markets firms, which face a challenging and constantly changing environment. We help you adapt, innovate, and transform your business and operating models, making it possible to do more with less code in less time.

Find out more at **genesis.global**