



Invest | Pro™

a Constellation Software Inc. Company




Build vs. Buy


Key considerations for life assurance, pensions firms and wealth managers in deciding to purchase software or build in-house


October 2020


A longstanding question facing the boards of financial services companies has been whether to buy proprietary software or build software in-house to meet their business needs. This whitepaper looks at the issue as it relates to software for investment administration in life assurance and pensions organisations but the general principles will apply to most software verticals.


There is no doubt that in-house IT teams are perfectly capable of building bespoke software. There are a few questions for decision makers to consider in this decision:

 **What is the opportunity cost of in-house build when internal IT teams are diverted from their primary task of growing the firm's business?**

 **Is the question less of "Can we build this?" and more "Should we build this?"**

 **Is software development and maintenance the core competency of the firm?**





 **What are the relevant cost comparisons?**

 **What are the risks of in-house build versus buying proprietary software from operational, reputational and regulatory perspectives?**

Rationales for 'Build' and 'Buy'








Build Arguments

The arguments advanced to support the in-house build approach are as follows:

-  A market developed solution will not meet a firm's specialised needs
-  An in-house solution is cheaper
-  The firm retains control over development and is not at the mercy of an external vendor's development timescales
-  In-house built software will yield a competitive advantage over rival firms

Buy Arguments

The arguments in favour of buying proprietary software are as follows:

-  Buying market tested and validated proprietary software is a much lower risk approach than in-house build
-  Time to market is quicker with proprietary software than in-house build
-  The costs of building and maintaining proprietary software are shared across multiple clients rather than a single client
-  In-house built software cannot afford the resources, time or cost to invest in market leading security, usability, scalability, upgradeability and auditability
-  Proprietary software will generally come with a dedicated support service level agreement which is generally not available in-house
-  Proprietary software avoids inevitable 'keyman' risk associated with internal builds
-  Proprietary software will generally have a much higher 'resilience' level than in-house built software

In this whitepaper we explore the above arguments in more detail.

The Rationale for In-House Build Software

1 A Market Developed Solution will not meet a Firm's Specialised Needs

It may be the case that a firm cannot find a proprietary solution that meets 100% of its requirements and the gap between what can be met by a proprietary solution and what is required is too large. In these circumstances a firm may have no other choice but to build in-house.

2 In-House Built Solutions are More Cost Effective ... or are they?

Where a proprietary solution is available to meet a firm's business requirements it will generally compare the costs of the proprietary solution with the costs of an in-house build. A key problem with this cost comparison is that it generally does not compare like-for-like as explained below.

2.1 Limited Functionality Set

The cost of the in-house build will generally only include the initial cost for what might be called a 'minimum viable product' ("MVP"), i.e. a software package that meets the functionality set requested by the firm's business users based on today's business needs. Market solutions offer broader functionality as a result of a large set of customer requirements driving the software's functional footprint.

The MVP is costed on a 'once and done' basis. The reality with software is that it requires constant maintenance as business requirements change, the regulatory landscape changes and technology itself changes. Technology upgrades are required to ensure compatibility with operational systems and other functional systems. The average support life of a Microsoft SQL Server version for example is just four years. Various other operating systems have even shorter lifespans.

A suitably qualified and experienced team of developers and support staff is needed to not only make all these changes but also to comprehensively test them. This maintenance cost is often overlooked in the 'once and done' approach to costing the MVP.

2.2 Automated Testing

Most in-house built systems do not have a comprehensive automated test framework for the software because building this will at least double the cost and the timeframe for the project. For most in-house builds this doubling of the project size to build automated testing is too daunting and is deemed unnecessary. What happens in reality is that the time taken to manually test a new system often exceeds the time it would have taken to build the automated testing framework but this reality is only understood in hindsight.

A further problem with this approach is that every time a change is required to the software another massive manual test effort is required to test the enhanced software.

The cost of building a comprehensive automated test framework or the alternative cost of large continuous manual test projects is seldom included in the overall cost benefit analysis when comparing in-house build versus buying external proprietary systems.

Because testing of the enhanced software is manual and generally limited in scope some errors in the software will not be discovered until it is in production. The costs of additional errors, be they customer compensation, large scale policy admin system record amendments or reputational damage, are also not included in the in-house versus external analysis.

2.3 Security Considerations

Financial services businesses, subject to high levels of regulation, will insist on minimum security standards for any proprietary software they buy. These security standards will include:

- ✓ Access controls including:
 - Strong password controls, password encryption, users required to change the default password after first logon etc.
 - Special privileges, such as security account set-up and administration restricted to specific administrators
 - Multi-factor authentication used for remote access, administrators/privileged users, web applications, cloud environments
 - Access to resources is based on business need following least privilege principle
- ✓ Regular system vulnerability and penetration testing to mitigate cyber security risk
- ✓ Ability to apply 'four eyes' checks on all input screens
- ✓ Ability to apply segregation of duties on appropriate processes
- ✓ Maintaining records of all changes including the 'changed from' and 'changed to' data, the date/time of the change, person doing the change, machine ID etc.

Most in-house built applications will not have the same industrial level of security that proprietary software needs to have available to compete in the market.

The cost of building these security features is either not included or significantly underestimated when comparing in-house build costs with proprietary software.



2.4 Usability



In-house built software generally does not have the same level of usability and automation as proprietary software which is built to meet a market need. Modern proprietary software will be automated and exception based, i.e. the system will run itself and check for exceptions. If no exceptions are found no further human intervention is required.

On in-house built systems the task of running and validation a process is generally as follows:

- ✓ The user is required to open one or more screens
- ✓ The user selects a set of inputs on each screen (requiring him/her to remember what input dates to choose etc)
- ✓ Press the 'OK' button to run the process
- ✓ Download the results to Excel
- ✓ Run a set of macros to verify the results

There are multiple costs associated with this lack of usability:

- ✓ Staff costs are higher because they spend more time doing their work on in-house built systems compared to automated exception based proprietary systems
- ✓ More highly qualified staff are required to run and verify results
- ✓ Operational risk is generally higher on in-house built systems as staff may inadvertently choose an incorrect input – the costs of correcting these errors can be significant
- ✓ There is no audit trail of any validation in the in-house built system as validation takes place on an Excel spreadsheet

These additional costs are seldom included in the build versus buy cost comparison.

2.5 Auditability



Most modern proprietary software solutions will include a full audit trail of all activity on the system, such as who logged onto the system, what processes they ran, what reports they looked at, what data they imported into the system, what data they changed, what data they exported to other systems, and will also include similar records for system initiated tasks.

System initiated tasks include validations of automated processing results which are stored in the system database unlike the validation of most in-house built system processing which takes place external to the system, generally in Excel.

This audit trail can be reviewed, filtered and sorted in user reports and viewed on dashboard widgets.

This comprehensive audit trail and the reporting capability around it is generally not included as a requirement of an in-house built system and hence not included in the cost comparison between build and buy.

2.6 Scalability



Most in-house built systems are not architected with scalability in mind and are generally written to cope with current business volumes. If business volumes increase significantly it may be necessary to re-architect the system.

Proprietary systems on the other hand are built with scale in mind as the vendor will generally have a range of clients from the small to the very large where scalability is a key requirement.

This point is similar to that in 2.1 re building for the current functionality set but in this case the in-house built system is built to manage current capacity only or marginal variations around it.

This scalability cost is often overlooked in the 'once and done' approach to costing the MVP.



2.7 Upgradeability



In-house built systems will use the latest technology when they are built. Technology has a limited support lifetime from most of the large technology, operating systems and database systems providers, like Microsoft, Oracle etc. This may be between 5-10 years.

So unless the technology underpinning the in-house built system is continually upgraded over time the once 'state of the art system' will become an unsupported legacy system with significant 'technical debt'. The interest on this technical debt will be paid annually, for example by trying to obtain security patches for unsupported operating systems, but eventually the capital of the debt will have to be repaid either in the form of a re-write of the system or the purchase of a proprietary system.

This upgrade cost (or the cost of the technical debt associated with not upgrading) is generally overlooked in the 'once and done' approach to costing the MVP.

A recurring problem with in-house built software is that the developers who have built the software may leave the company or get promoted to other roles. This leaves a knowledge gap where there are no developers expert in the system or there is a dependency on a key person to maintain it. This could be termed a 'people debt' instead of a 'technical debt'.

The level of documentation created for in-house built software is never at the same standard as that required for proprietary software and is not maintained to the same level as internal staff will always have 'higher priorities'.

This creates a significant resilience risk for the firm which can be mitigated by having multiple developers familiar with the system and capable of maintaining it.

The cost of mitigating this resilience risk, that is, having a development team which is capable of maintaining the in-house system, is seldom built into the cost comparison of build versus buy.

2.8 Resilience



Regulators are increasingly focusing on the resilience of the systems used by financial services firms following some large-scale failures of IT systems in recent years, as seen in the FCA's 2019/2020 Building operational resilience consultation paper¹. One of their key priorities is addressing the risks of harm that could result from insufficient operational resilience.

Operational disruptions can have many causes including, for example, technology failures when making changes to systems, cyber-attacks, inability to obtain market prices on time, loss of key people and wider telecommunications or power failure.

Investment administration is a key function for firms managing unit-linked business and as such would be covered by these resilience requirements.

Proprietary software firms will have a service level agreement ("SLA") with their clients which will provide for appropriate response times to any issues that arise in the software. The cost of this SLA support is generally included as part of the annual licence fee.

»» Cost Summary ««

The cost of adding the features outlined above is a multiple of the order of 3-5 times the cost of the MVP functionality. Internal IT development teams may argue that such features are a luxury for an internal build and are not required at the outset but no procurement department would exclude these requirements in an RFP for any proprietary system. This argument fails to recognise the long-term total cost of ownership of the in-house solution.

An in-house built system without these features is a classic case of the 'technical debt' metaphor well recognised in the software engineering industry.

Interest on this technical debt is paid every year in the form of additional staff costs, both higher number and higher quality staff will be required, higher error rates, greater security risks and lower operational resilience.



Eventually the technical debt has to be repaid when the in-house system becomes an unsupported legacy system and requires a full re-write.

3 Development Control

With an in-house build the argument is that the firm retains control over development and is not at the mercy of an external vendor's development timescales.

This is true particularly if an external proprietary system does not meet all the firm's requirements and substantial initial and ongoing development is required as business requirements evolve.

Where an external proprietary system is available which does meet the firm's requirements the opposite may be the case as the functionality required may already be included in the proprietary system and simply needs to be 'turned on' whereas this would require a significant project with the in-house build.

4 Competitive Advantage

Where a firm has some intellectual property in its in-house built software which yields a competitive advantage over rival firms this is clearly something it will wish to retain and not wish to have available in a proprietary system.



The Rationale for Buying Proprietary Software

1 Minimise Project Risk

Buying market tested and validated proprietary software is a much lower risk approach than in-house build.

Documenting the business requirements to the level of detail needed by a software developer is not the core skillset of a fund accountant or a unit pricing analyst (unless they came from a software development environment). Unless the software development team has expertise in the investment administration, unit pricing, investment accounting fields etc. they simply cannot tell whether the requirements specified by the business team are in enough detail to build the appropriate software. This is a key risk and reason why internal builds of complex systems fail outright or fail to meet expectations of the business team.

It's this missing link that also results in many internal IT projects taking significantly longer than expected, and/or ultimately costing multiples of the original estimated budget. The more complex the system the higher the risk of project failure.



Another major risk in this approach is that the new in-house build is a more automated version of the old application rather than a complete re-think of the overall operating model design. Henry Ford put it well when he said 'If I had asked people what they wanted, they would have said faster horses'. Software vendors who work with many clients will have developed best practice in the market and designed the 'car' rather than the faster 'horse'.

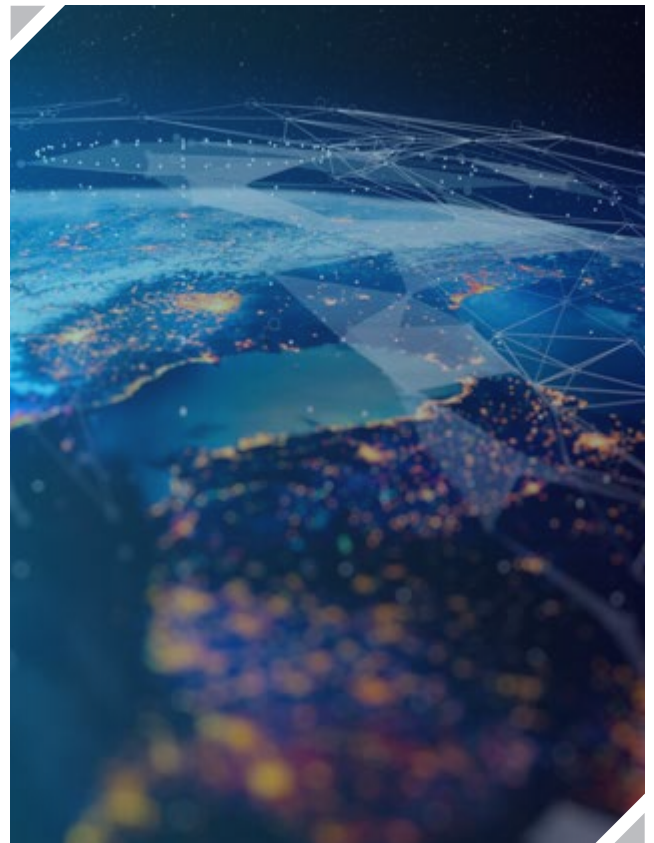
A study published in the Harvard Business Review² in September 2011 estimated that one in six IT projects had an average cost overrun of 200% and a schedule overrun of almost 70%. These regular IT project overruns are easily avoided with market developed software which comes with a licence fee and carefully estimated and managed implementation project fees.

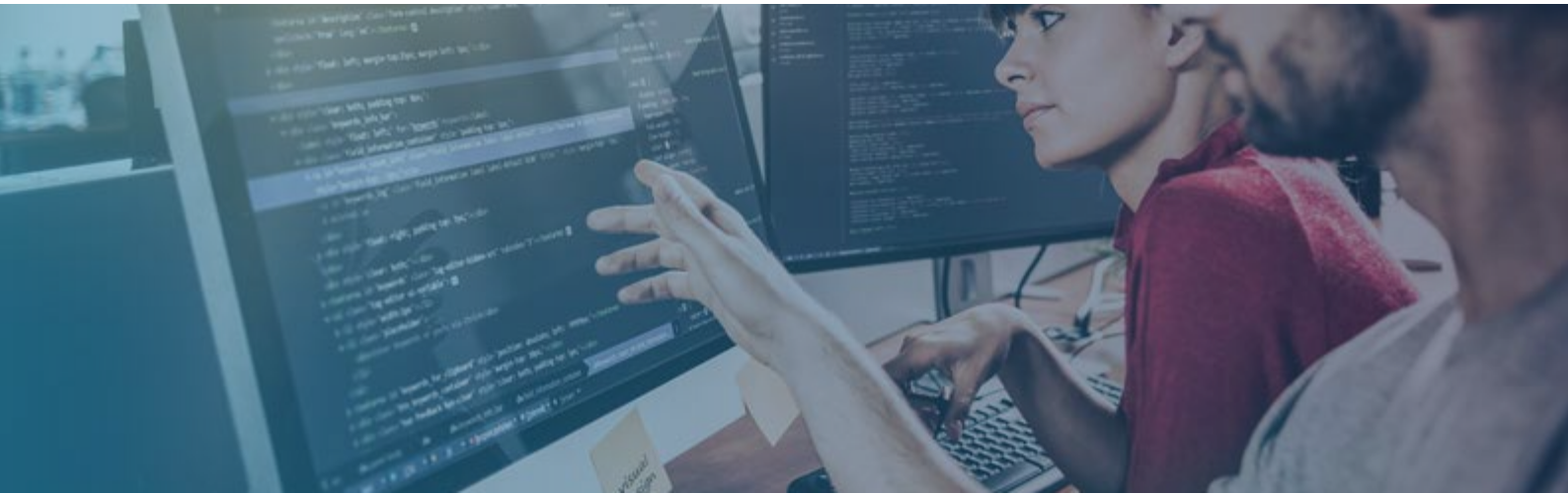
2 Time to Market

Time to market is quicker with proprietary software than in-house build. A fully functional enterprise in-house investment administration system may take 3-5 years to build and test assuming the project does not fail outright. This is a significant opportunity cost for both the business users required for the specification and test effort but also for the development and project management teams.

Implementing a fully functional enterprise proprietary investment administration system should take no more than 9-18 months depending on the size of the organisation.

Time to market is not an initial 'one-off' issue – this will recur every time competitors develop new products or new regulations require changes to the software. This can cause a competitive disadvantage for firms with in-house software as they are often unable to introduce new product features at a pace that matches their rivals using market solutions.





3 Shared Development Costs in Proprietary Software

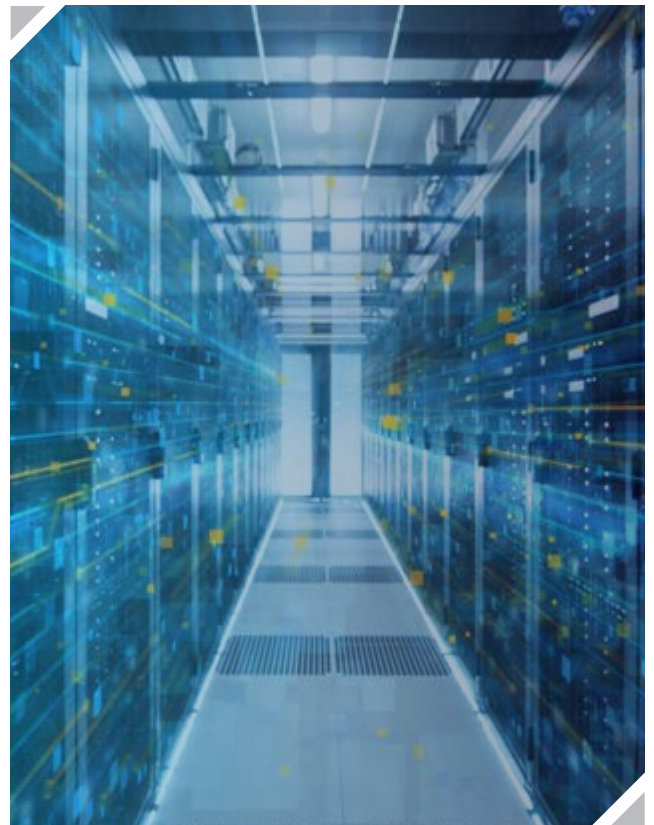
A software vendor which develops a product for multiple clients can spread the cost of this development over the client base. All the costs in section 2 are spread across the client base rather than borne by a single client.

The software vendor will obviously apply a profit margin on its costs but this is small in comparison to dividing development costs across a client base of 50-100 clients.

4 Dedicated Support

Proprietary software vendors will have SLA's to support their software and a dedicated support team to provide assistance when required. The cost of this support team is also spread across a large client base so will in general be much cheaper than providing dedicated internal resources for an in-house built package.

The ability to rely on an external vendor for support eliminates 'key person' risk often associated with in-house built systems. This issue is becoming much more relevant with the current regulatory focus on 'Resilience'.



»» Summary ««

In summary, the decision to build in-house or invest in propriety software deserves careful consideration. Two particular areas to consider in any comparison are whether cost comparisons are on a 'like for like' basis and the significant underlying economic advantage a proprietary vendor has in its ability to spread costs across its client base.

Footnote 1; FCA Building operational resilience consultation paper <https://www.fca.org.uk/publication/consultation/cp19-32.pdf>

Footnote 2; Harvard Business Review 'Why Your IT Project May Be Riskier Than You Think' <https://hbr.org/2011/09/why-your-it-project-may-be-riskier-than-you-think>

About Financial Risk Solutions Ltd (FRS)

With over 20 years delivering Investment Administration software, Financial Risk Solutions Ltd (FRS) is a trusted technology partner to life assurance, wealth and asset management firms worldwide. Led by an expert team of actuaries, compliance and IT specialists, clients license FRS software to help navigate the ever-changing challenges of growth, regulatory pressures and competition in the industry.

The award-winning* InvestPro™ platform is relied on by over 60 blue-chip financial services and BPO clients to reduce operational costs, increase efficiencies and mitigate risk in the manufacture and management of investment products. More than 150,000 funds are managed on the Invest|Pro™ platform today.

Delivered on-premise or cloud-hosted, Invest|Pro™ securely automates multiple complex fund administration processes including unit-pricing, cash allocation and rebalancing; oversight and validation of operational activity performed by outsourced partners; and in Europe monitoring and reporting for PRIIPs, KID requirements, and Pillar III asset reporting for Solvency II.

FRS is part of the Constellation Software Inc. group and headquartered in Dublin, Ireland, with offices in Hong Kong and Sydney.

For more information visit frsltd.com or follow FRS on LinkedIn at www.linkedin.com/company/frs-ltd



Peter Caslin is the CEO and a founder of FRS, Peter is a Fellow of the Society of Actuaries in Ireland.



*2020 - GRC Product of the Year - Asia Risk.Net Awards, 2019 - Best Solvency II Tech Solution - Insurance Asset Management Awards, Pensions Technology Provider of the Year - Irish Pensions Award, 2017 & 2018 - Tech Firm of the Year - Insurance Asset Management Awards, 2016 - Tech Provider of the Year, Governance Risk and Compliance - Risk.Net Awards.



Peter Caslin

Chief Executive Officer
Financial Risk Solutions

✉ peter.caslin@frsltd.com